Espirales revista multidisciplinaria de investigación
ISSN: 2550-6862

Metodologías ágiles frente a las tradicionales en el proceso de desarrollo de software

Agile methodologies against traditional methods in the software development process.

Enviado marzo 2018 – Revisado abril 2018 – Publicado junio 2018

Bryan Molina Montero [1]
Harry Vite Cevallos[2]
Jefferson Dávila Cuesta[3]

[1] Ingeniero en Sistemas. Maestrante en Sistemas de Información Gerencial. bryanmolinamontero@gmail.com
[2] Ingeniero en Sistemas. Maestrante en Sistemas de Información Gerencial. hvite@utmachala.edu.ec
[3] Ingeniero en Sistemas. Maestrante en Sistemas de Información Gerencial. jeffersondavila@outlook.es

ISSN: 2550-6862

Resumen

El proceso de desarrollo de software a través de los años se ha venido implementando una serie de metodologías que facilitan a programación. La presente investigación realiza una revisión de publicaciones sobre las metodologías llamadas ágiles, sus principios y fundamentos, estableciendo definiciones y explicaciones detalladas de las más relevantes en la actualidad (Scrum y XP), convirtiéndose en la más acertada para el desarrollo de los procesos de ingeniería de software.

Palabras clave
Ingeniería de software, metodologías de desarrollo, metodologías ágiles.

Abstract
The process of software development over the years has been implementing a series of methodologies that facilitate programming. The present investigation makes a review of publications on the so-called agile methodologies, their principles and foundations, establishing definitions and detailed explanations of the most relevant at present (Scrum and XP), becoming the most successful for the development of engineering processes of software.
Key words
Software engineering, development methodologies, agile methodologies.

1. Introducción

En la década de los 50 el desarrollo de sistemas estaba a cargo de programadores más enfocados en la tarea de codificar, que en la de comprender y recoger las necesidades de los usuarios, que muy a menudo no quedaban satisfechos con el resultado final, es decir no era un software de calidad. La calidad no solo se refiere a la satisfacción del cliente, sino que también puede referirse a su velocidad, estabilidad, flexibilidad, seguridad, usabilidad, escalabilidad, entre muchos otros atributos.

Pero para poder hablar de calidad, se tuvo que generar un gran proceso histórico que sigue en constante evolución, como son las Metodologías de Desarrollo de Software. Estas proponen como objetivo principal presentar un conjunto de técnicas tradicionales, modernas y ágiles de modelado de sistemas que permitirían desarrollar software con calidad, incluyendo heurísticas de construcción y criterios de comparación de modelos de sistemas. (Egas & Játiva, 2008)

La Ingeniería de software es la aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento de software, y el estudio de estos enfoques, es decir, la aplicación de la ingeniería al software.(Egas & Játiva, 2008)

El principal problema es que de las múltiples metodologías de desarrollo de software existentes no se selecciona la adecuada o la que más convenga, y en el peor de los casos no se selecciona ninguna (al menos de manera explícita), para desarrollar el software que se requiere.

Para dar solución a lo anterior, en este trabajo se pretende brindar una descripción general de las metodologías de desarrollo tradicionales, así como también una descripción de las metodologías de desarrollo ágiles presentando algunas como: Scrum y XP. Además, un análisis comparativo que permitirán tomar decisiones en la elección de una metodología de desarrollo de software.

2. Metodología

En la presente investigación se realiza un recorrido bibliográfico de las principales corrientes sobre las metodologías de desarrollo tradicional frente a las contemporáneas y sus diferencias, presentando sus características y aspectos relevantes que se han ido desarrollando en el tiempo de la sociedad del conocimiento.

Las metodologías desarrolladas se han ido modificando a las innovaciones requerida en cada momento, evidenciando cambios pequeños en algunos casos y giros importantes en otros, siendo importante su estudio.

3. Resultados

La revisión bibliográfica, facilito la establecer los cambios presentados y las consideraciones que permitieron su implementación en el desarrollo de software, para lo cual se plantea el recorrido de los principales autores en esta temática a fin de ir evaluando sus criterios de trabajo, relacionados a desarrollo de las metodologías.

En primer lugar, es sustancial conocer la definición de la palabra metodología, siendo una palabra formada por tres vocablos griegos: metá que significa más allá, odós que significa camino y logos que significa estudio; considerando lo anterior como ese estudio más allá del camino. (Rivas, Corona, Gutiérrez José Fructuoso, & Henandez Lizeth, 2015) .

3.1 Metodologías de desarrollo tradicional

Según (Pressman, 2013), las metodologías de desarrollo tradicionales o clásicas son también llamados modelos de proceso prescriptivo, y fueron planteadas originalmente para poner orden en el caos del desarrollo de software que existía cuando se empezó a generar masivamente. La historia revela que estos modelos tradicionales que fueron presentados en la década de los 60, dieron cierta estructura útil al trabajo de la Ingeniería de software y constituyen un mapa razonablemente eficaz para los equipos de desarrollo.

En las metodologías tradicionales se concibe al proyecto como uno solo de grandes dimensiones y estructura definida; el proceso es de manera secuencial, en una sola dirección y sin marcha atrás; el proceso es rígido y no cambia; los requerimientos son acordados de una vez y para todo el proyecto, demandando grandes plazos de planeación previa y poca comunicación con el cliente una vez ha terminado ésta.

En la década de los 70 aparece algo denominado el ciclo de vida de desarrollo de software, como un consenso para la construcción centralizada de software, y daría las pautas de manera general que logran establecer los estados por los que pasa el producto software desde que nace a partir de una necesidad, hasta que muere. (Egas & Játiva, 2008)

El primer modelo publicado acerca del proceso de desarrollo de software, se originó por procesos más generales de la ingeniería, debido al paso de una fase en cascada a otra, Winston Royce define al modelo como modelo en cascada, que empezó a diseñarlo en el año 1966 y fue terminado alrededor de 1970. Este modelo propone un enfoque secuencial y

ISSN: 2550-6862

sistemático para el desarrollo de software, conlleva más disciplina y se basa principalmente en las etapas de análisis de requisitos, diseño, codificación, pruebas y mantenimiento. (Sommervjlle, 2006)

A continuación, se presenta gráficamente (Figura 1) las fases del modelo en cascada.

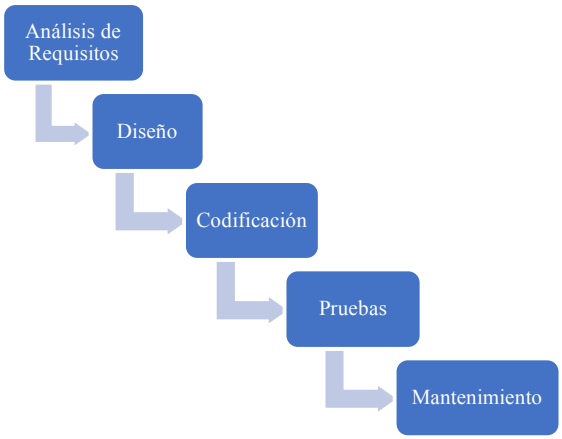

Figura 1. Fases del modelo en cascada.

El modelo en cascada se define como una secuencia de fases, que al final de cada etapa reúne toda la documentación para garantizar que cumple con los requerimientos y especificaciones.

El modelo en cascada para la época se convirtió en un pilar fundamental de ejemplo de proceso dirigido, donde se planificaría todas las actividades antes de comenzar a trabajar en ellas.

Al pasar el tiempo se empieza a detectar los principales problemas tales como la dificultad de responder a los requerimientos cambiantes del cliente.

3.2 Metodologías de desarrollo ágil

Hoy en día, el mundo empresarial opera en un entorno global que cambia rápidamente; por ende, se debe responder a las nuevas necesidades y oportunidades del mercado, teniendo en cuenta que el software es partícipe de casi todas las operaciones empresariales, se debe desarrollar soluciones informáticas de manera ágil para poder dar una respuesta de calidad a todo lo necesario. (Rivas et al., 2015)

Las metodologías ágiles presentan como principal particularidad la flexibilidad, los proyectos en desarrollo son subdivididos en proyectos más pequeños, incluye una comunicación constante con el usuario, son altamente colaborativos y es mucho más adaptable a los cambios. De hecho, el cambio de requerimientos por parte del cliente es una característica especial, así como también las entregas, revisión y retroalimentación constante. (Cadavid, Fernández Martínez, & Morales Vélez, 2013)

Entre las más notables metodologías de desarrollo ágil, se encuentran:

Scrum (Muy popular en emprendimientos)

Programación extrema (XP)

Crystal Clear

Mobile-D (ágil y extrema para móviles)

Adaptive Software Development (ASD)
Lean Development
Entre las más populares actualmente se encuentran Scrum y XP, las cuales se detallas a continuación:

3.3 Metodología Scrum

Scrum no corresponde a ningún acrónimo, su nombre proviene del deporte rugby, que es una formación requerida para la recuperación rápida del juego ante una infracción menor.(Cadavid et al., 2013)
Scrum es un marco de trabajo diseñado de tal forma que logra la colaboración eficaz del equipo de trabajo, emplea un conjunto de reglas y se definen roles para generar una estructura de correcto funcionamiento.
Scrum define tres roles, los cuales son: El Scrum master, el dueño del producto o Product owner y el equipo de desarrollo o team. El scrum master es la persona que lidera el equipo asegurándose que el equipo cumpla las reglas y procesos de la metodología. El dueño del producto es el representante de los accionistas y clientes que usan el software. El equipo de desarrollo es el grupo de profesionales encargados de convertir la lista de requerimientos o también llamado Product Backlog en funcionalidades del software (Cadavid et al., 2013)
Scrum utiliza un elemento representativo llamado Sprint (figura 2) que corresponde a una etapa de trabajo donde se crea una versión utilizable del producto. Cada sprint es considerado como un proyecto individual. Un Sprint está compuesto por los siguientes elementos: reunión de planeación del Sprint, Daily Scrum o reunión diaria, trabajo de desarrollo, revisión y retrospectiva del Sprint.

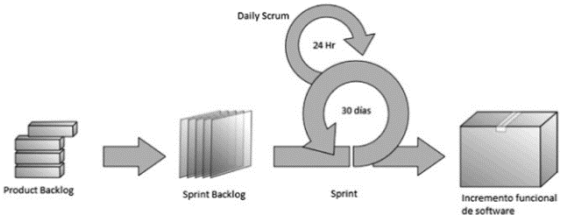

Figura 2. Fases del Sprint.
Fuente: Tomado de (Cohn, 2004)

3.4 Metodología XP

La metodología extreme programming o XP, es la metodología ágil más conocida (Sato, Bassi, Bravo, Goldman, & Kon, 2006). Fue desarrollada por Kent Beck en la búsqueda por guiar equipos de trabajo pequeños o medianos, entre dos y diez programadores, en ambientes de requerimientos imprecisos o cambiantes
La principal particularidad de esta metodología son las historias de usuario, las cuales corresponden a una técnica de especificación de requisitos; se trata de formatos en los cuales el cliente describe las características y funcionalidades que el sistema debe poseer.(Beck, 1991)
En esta metodología se realiza el proceso denominado Planning game, que define la fecha de cumplimiento y el alcance de una entrega funcional, el cliente define las historias de usuario y el desarrollador con base en ellas establece las características de la entrega, costos de implementación y número de interacciones para terminarla. Para cada iteración el cliente

estipula cuales son las historias de usuario que componen una entrega funcional.(Cadavid et al., 2013)

Se realizan entregas pequeñas que son el uso de ciclos cortos de desarrollo, llamado iteraciones, que muestra al cliente una funcionalidad del software terminado y se obtiene una retroalimentación de él.

Algo muy característico de esta metodología es la programación en parejas, indica que cada funcionalidad debe de ser desarrollada por dos programadores, las parejas deben cambiar con cierta frecuencia, para que el conocimiento no sea solo de una persona sino de todo el equipo. (Cadavid et al., 2013)

Para terminar con lo relevante de esta metodología, se presenta una etapa muy importante las cuales son las pruebas de aceptación, una vez que se ha desarrollado una funcionalidad, entra a pruebas por parte del cliente, dando su aprobación.

3.5 Metodologías tradicionales versus metodologías ágiles

Las metodologías de desarrollo tradicionales imponen una disciplina de trabajo fundamentada en la documentación sobre el proceso de desarrollo de software, se realiza un hincapié en la planificación global y total de todo el trabajo a realizar, y una vez que esté detallado, comienza el ciclo de desarrollo de software; caso contrario a lo que respecta a las metodologías de desarrollo ágiles que muchas veces obvia la documentación y se centra en el trabajo, busca el equilibrio entre proceso/esfuerzo. (Cáceres, Marcos, & Kybele, 2001)

La tabla 1 muestra los aspectos relevantes entre las metodologías de desarrollo tradicionales y las metodologías ágiles.

Tabla 1. Metodologías tradicionales vs metodologías ágiles.

Metodologías Tradicionales	Metodologías Ágiles
Predictivos Orientado a procesos	Adaptativos Orientado a personas
Proceso rígido Se concibe como un proyecto	Proceso flexible Un proyecto es subdividido en varios proyectos más pequeños.
Poca comunicación con el cliente	Comunicación constante con el cliente.
Entrega de software al finalizar el desarrollo	Entregas constantes de software
Documentación extensa	Poca documentación

Fuente: Tomada de (Cadavid et al., 2013)

De igual manera otros autores concuerdan y agregan más diferencias en esta comparación, como se puede apreciar en la tabla 2.

Tabla 2. Comparación de metodologías

Metodologías Ágiles	Metodologías Tradicionales
Se basan en heurísticas provenientes de prácticas de producción de código	Se basan en normas provenientes de estándares seguidos por el entorno de desarrollo
Preparados para cambios durante el proyecto	Cierta resistencia a los cambios
Impuestas internamente por el equipo	Impuestas externamente
Proceso menos controlado, con pocos principios	Proceso muy controlado, numerosas normas
Contrato flexible e incluso inexistente El cliente es parte del desarrollo	Contrato prefijado Cliente interactúa con el equipo de desarrollo mediante reuniones
Grupos pequeños (<10)	Grupos grandes
Pocos artefactos Menor énfasis en la arquitectura del software	Más artefactos La arquitectura del software es esencial

Fuente: Tomada de (Canós, Letelier, & Penadés, 2014)

ISSN: 2550-6862

4. Conclusiones

Se puede determinar que no existe una metodología universal para hacer frente a cualquier proyecto de desarrollo de software; toda metodología debe aplicada de acuerdo con el contexto tales como recurso humano, documentación necesaria, tiempo, disponibilidad del usuario.

Todas las metodologías tienen ventajas que se pueden aprovechar, así como también tienen desventajas, lo importante es saber escoger una metodología apropiada en el desarrollo de software.

Las metodologías de desarrollo ágil van destinadas para equipos de trabajo donde sus integrantes sean menor a diez.

Para usar una metodología de desarrollo ágil es necesario la disponibilidad del cliente, ya que se necesita su la retroalimentación de manera continua.

Las metodologías tradicionales presentan cierta resistencia a los cambios.

Muy independiente de la metodología que se emplee, hay que tener en cuenta que el producto final de un desarrollo debe de ser un software de calidad.

Referencias bibliográficas

Beck, K. (1991). *Extreme Programming Explained* (2da ed.). Retrieved from http://ptgmedia.pearsoncmg.com/images/9780321278654/samplepages/9780321278654.pdf

Cáceres, P., Marcos, E., & Kybele, G. (2001). Procesos ágiles para el desarrollo de aplicaciones Web. *Taller de Web Engineering de Las Jornadas de Ingeniería Del Software Y Bases de Datos*. Retrieved from http://dlsi.ua.es/webe01/articulos/s112.pdf

Cadavid, A. N., Fernández Martínez, J. D., & Morales Vélez, J. (2013). Revisión de metodologías ágiles para el desarrollo de software. Retrieved from https://uac.edu.co/images/stories/publicaciones/revistas_cientificas/prospectiva/volumen-11-no-2/4_articulo_vol_11_2.pdf

Canós, J., Letelier, P., & Penadés, M. C. (n.d.). Metodologías Ágiles en el Desarrollo de Software. *Universidad Politécnica de Valencia*. Retrieved from http://issi.dsic.upv.es/archives/f-1069167248521/actas.pdf

Cohn, M. (2004). *User stories applied : for agile software development*. Addison-Wesley.

Egas, L., & Játiva, J. (2008). Evolución de las Metodologías de Desarrollo de la Ingeniería de Software en el Proceso la Ingeniería de Sistemas Software, Creación:2008;Recuperado:9 mayo 2015. Retrieved from http://repositorio.espe.edu.ec/bitstream/21000/8771/1/AC-ESPEL-SOF-0004.pdf

Pressman, R. S. (2013). *Ingeniería de Software un enfoque práctico*. *Journal of Chemical Information and Modeling* (7ma ed., Vol. 53). Mc Graw Hill. https://doi.org/10.1017/CBO9781107415324.004

Rivas, C. I., Corona, V. P., Gutiérrez José Fructuoso, & Henandez Lizeth. (2015). Metodologías actuales de desarrollo de software. *Artículo Revista Tecnología E Innovación Diciembre*, *2*(5), 980–986. Retrieved from http://www.ecorfan.org/bolivia/researchjournals/Tecnologia_e_innovacion/vol2num5/Tecnologia_e_Innovacion_Vol2_Num5_6.pdf

Sato, D., Bassi, D., Bravo, M., Goldman, A., & Kon, F. (2006). Experiences tracking agile projects: an empirical study. *Journal of the Brazilian Computer Society*, *12*(3), 45–64. https://doi.org/10.1007/BF03194495