Revista Ingenio
UFPS Ocaña
"Journal of Engineering Sciences"

Original Article DOI: https://doi.org/10.22463/2011642X.2779

Mic-agile: metodología ágil para micro-equipos de desarrollo de software

Mic-agile: agile methodology for software development micro-teams

Ing. Alberto Ramos-Blanco[1], Ph.D. Hector G. Pérez-González[2], Ph.D. Sandra E. Nava-Muñoz[3], Ph.D. Francisco E. Martínez-Pérez[4],

[1] Ciencias de la Computación, Universidad Autónoma de San Luis Potosí, México, Orcid: https://orcid.org/0000-0002-4771-3337, Email: beto@uaslp.mx
[2] Ciencias de la Computación, Universidad Autónoma de San Luis Potosí, México, Orcid: https://orcid.org/0000-0003-3331-2230, Email: hectorgerardo@uaslp.mx
[3] Ciencias de la Computación, Universidad Autónoma de San Luis Potosí, México, Orcid: https://orcid.org/0000-0002-3133-9045, Email: eduardo.perez@uaslp.mx
[4] Ciencias de la Computación, Universidad Autónoma de San Luis Potosí, México, Orcid: https://orcid.org/0000-0001-9345-4391, Email: senavam@uaslp.mx

Cómo citar: A. Ramos-Blanco, H. G. Pérez-González, S. E. Nava-Muñoz y F. E. Martínez-Pérez, "Mic-agile: metodología ágil para micro-equipos de desarrollo de software", *Revista Ingenio*, vol. 19, n°1, pp. 1-8, 2022 doi: https://doi.org/10.22463/2011642X.2779

Fecha de recibido: 05 de mayo de 2021
Fecha aprobación:04 de noviembre de 2021

RESUMEN

Debido al creciente interés en las metodologías ágiles aplicadas en el proceso de software, se propone en este trabajo la denominada MIC-AGILE. Esta se orienta a equipos reducidos (micro-equipos), de desarrollo de software, y se ha utilizado con éxito por más de cinco años en una Facultad de Ingeniería de una universidad pública en México. Se compone por cuatro procesos (solicitud, definición, desarrollo y evolución de software). Se proponen artefactos de software que registran la trazabilidad del proceso y se describen los roles que toman los integrantes del equipo. La metodología MIC-AGILE ha reducido el tiempo de desarrollo y ha incrementado la productividad, cumpliendo restricciones de tiempo y recursos, sin sacrificar la calidad de los productos de software. Como resultado adicional, se presenta un caso de estudio exitoso que surge de una necesidad de la Facultad y se ha extendido a todas las facultades de la universidad.

Palabras claves:

Metodologías Ágiles, Micro-Equipos de Desarrollo de software, Modelos de Proceso de Software, Proceso de Desarrollo de Software.

ABSTRACT

Due to the growing interest in agile methodologies applied in the software process, the so-called MIC-AGILE is proposed in this work. This is aimed at small teams (micro-teams), software development, and has been used successfully for more than five years in a Faculty of Engineering of a public university in Mexico. It consists of four processes (application, definition, development and evolution of software). Software artifacts are proposed that record the traceability of the process and describe the roles that team members play. The MIC-AGILE methodology has reduced development time and increased productivity, meeting time and resource constraints, without sacrificing the quality of software products. As an additional result, a successful case study is presented that arises from a need of the Faculty and has been extended to all faculties of the university.

Key words:

Agile Methodologies, Software Micro-teams, Software Process, Software Process Model.

1. Introducción

En el manifiesto del año 2001 enfocado en el Desarrollo de Software Ágil [1] se muestran principios que deben ser tomados en cuenta para el desarrollo de software de una forma ágil, con una mayor prioridad en satisfacer al cliente. En este sentido, se han analizado diversas metodologías que persiguen estos principios tales como scrum, programación extrema (XP), metodología de cristal, entre otras [2]. Una gran parte de la industria del software está utilizando Scrum y programación extrema (XP), a menudo en combinación [3]. Scrum es un marco de gestión de proyectos, mientras que XP es un conjunto de prácticas destinadas a mejorar la calidad del software y responder a los cambios en los requisitos del cliente [3].

Sin embargo, el lograr satisfacer al cliente tiene muchas implicaciones, entre la más común es el desconocimiento de lo que el cliente desea. Adicionalmente el escalar hacia la utilización de una metodología ágil no es proceso sencillo, ya que los proyectos grandes se distribuyen globalmente teniendo que considerar equipos de trabajo que necesitan coordinarse y colaborar [4]. Los arquitectos de software generalmente enfrentan un dilema: determinar el tiempo a invertir en la arquitectura de un software antes de comenzar el desarrollo. Para ello es necesario considerar si los requerimientos son estables, la cantidad de riesgos técnicos, y las fechas de inicio de uso del cliente [5].

Adicionalmente, las metodologías ágiles han cobrado especial interés en el sector académico y profesional,

Autor para correspondencia
Correo electrónico: beto@uaslp.mx (Alberto Ramos Blanco)

La revisión por pares es responsabilidad de la Universidad Francisco de Paula Santander Ocaña
Artículo bajo la licencia CC BY-NC (https://creativecommons.org/licenses/by-nc/4.0/)

permitiendo principalmente dotar a las micro, pequeñas y medianas empresas de prácticas que favorecen su quehacer y la gestión de sus recursos desde enfoques ágiles y sin mayores formalismos que las metodologías tradicionales [6].

En este artículo se presenta una metodología que satisface los requerimientos del trabajo realizado en un departamento de desarrollo de software de una universidad pública en México. Para alcanzar este objetivo, el trabajo de investigación incluye las siguientes fases: i) revisión del estado del arte de las metodologías ágiles, ii) análisis y diseño de una propuesta de una metodología ágil y iii) desarrollo e implementación de la metodología en un caso de estudio. Se consideró un micro-equipo de desarrollo, por ser un grupo reducido de personal, como máximo cinco, cada uno con más de un rol de trabajo asignado y más de un proyecto asignado. Se adopta esta metodología para asegurar la calidad del producto de software, y disminuir su tiempo de desarrollo. Con la metodología en un tiempo corto, se desarrollan varios proyectos de software de manera paralela, el personal colabora y trabaja en equipo en los proyectos planificados, cumpliendo funciones y roles diferentes en cada proyecto.

2. Metodología de Desarrollo de Software

Esta metodología surge por la carga de trabajo que tiene el departamento y los tiempos cortos para implementar un proyecto. La metodología ágil responde mejor a las necesidades del cliente que los métodos tradicionales, como el modelo en cascada, el cual tiene un flujo de valor amplio y de movimiento lento [7].

El propósito de la metodología propuesta es desarrollar dos o más proyectos de software de manera paralela y que cumplan los requerimientos del cliente. Por lo que los entregables que se proporcionan a los clientes tengan la calidad esperada, con base en el estándar ISO/IEC/IEEE 12207 [8]. En la Figura 1 se describe como está compuesta a través de un diagrama de bloques.

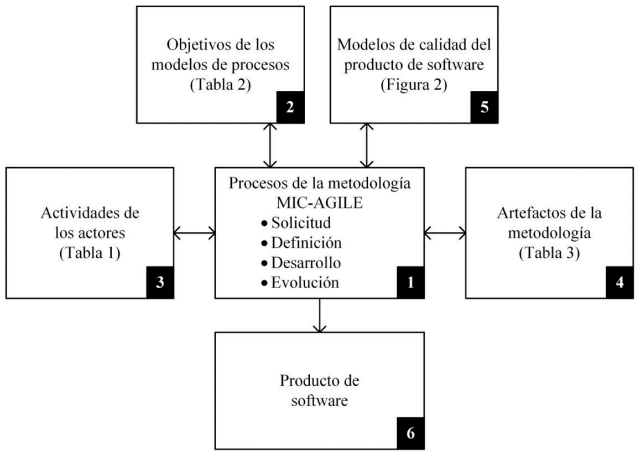

Figura 1. Diagrama de bloques de la metodología MIC-AGILE.

Con el objetivo de cumplir con las expectativas del cliente, y cubrir el 100% de los requerimientos del proyecto, mitigar errores y asegurar la calidad del software, éste debe reunir ciertas características que garanticen un excelente funcionamiento, productividad y que cumpla con su objetivo.

El modelo de calidad del producto de software ISO/IEC 9126-1 [9], mostrado en la Figura 2, sirve como marco de referencia para evaluar los productos de software que elabora el micro-equipo de desarrollo. Además, mantiene como objetivo las características que se deben cumplir en cada proyecto sin sacrificar y mantener la calidad del software. Las seis características de calidad que contiene el modelo ISO/IEC 9126, tomadas de Esaki [9], son: funcionalidad, fiabilidad, usabilidad, eficiencia, mantenibilidad y portabilidad, para medir la calidad del software (Figura 2).

2.1 Descripción del Micro-Equipo

El departamento de desarrollo (micro-equipo), esta embebido en el organigrama de la Facultad de Ingeniería, y lo conforman cinco ingenieros en ciencias de la computación, con más de diez años de experiencia en desarrollo de software, con conocimientos y especialidades en: programación y administración de bases de datos, sistemas operativos de servidor, servicios web, lenguajes de programación web, frameworks como marcos de trabajo para desarrollo de software, arquitectura de software y administración de proyectos.

2.2 Descripción de la Metodología MIC-AGILE

En esta sub-sección se describen los diferentes procesos que integran la metodología MIC-AGILE.

La Tabla 1 muestra los cuatro actores de la metodología MIC-AGILE que son: cliente, responsable de departamento, responsable de proyecto y responsable de proyecto N, este último debe entenderse como otro integrante del equipo, también responsable de otros proyectos. Esto se representa en la Figura 1, bloque 3.

En la Tabla 2 se presentan los objetivos de los procesos que componen la metodología MIC-AGILE, representado en la Figura 1 bloque 2.

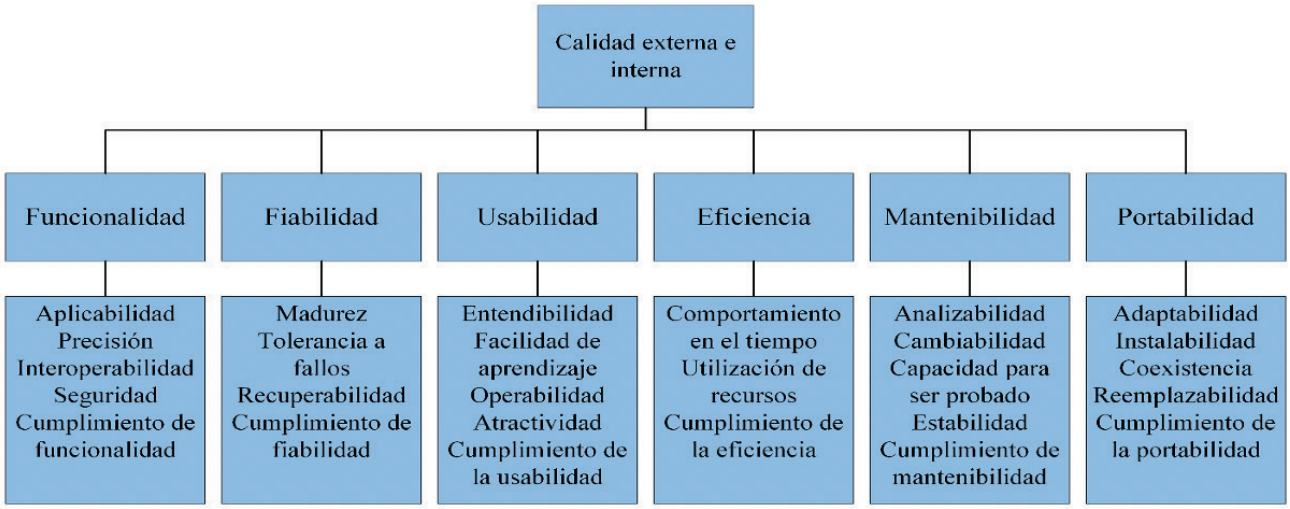

Figura 2. Modelo de calidad del producto de software. Fuente. [9]

Tabla 1. Actividades de los actores de la metodología MIC-AGILE

Actor	Actividad
Cliente	• Solicitar proyecto • Solicitar requerimientos • Aprobar requerimientos • Reconocer plan de trabajo • Aprobar avance • Recibir capacitación
Responsable de departamento	• Analizar solicitud de software • Priorizar proyecto • Asignar proyecto • Organizar y priorizar requerimientos • Evaluar arq. del proyecto. • Evaluar riesgos • Evaluar plan de trabajo • Evaluar avance
Responsable de proyecto *N*	• Reconocer proyecto • Definir arq. del proyecto • Definir y priorizar tareas. • Identificar riesgos • Generar plan de mitigación de riesgos • Generar plan de trabajo • Desarrollar de avance • Implantar avance • Capacitar al cliente
Responsable de proyecto	• Reconocer proyecto • Reconocer avance • Generar, realizar y evaluar pruebas del avance

Tabla 2. Objetivos de los modelos de proceso

Modelo de proceso	Objetivo
Proceso de solicitud	Evaluar la factibilidad del proyecto de software.
Proceso de definición	Definir la arquitectura del proyecto, identificar riesgos, generar un plan de riesgos y elaborar un plan de trabajo.
Proceso de desarrollo	Desarrollar, evaluar e implementar el proyecto.
Proceso de evolución	Actualizar, mejorar un producto de software.

En este trabajo se toma el término artefacto como se describe en Gazhi [10] en donde un artefacto es como cualquier tipo de documento textual o gráfico con la excepción del código fuente. Los artefactos pueden ser, por ejemplo, documentos de requisitos textuales, modelos gráficos (incluidos diagramas de Lenguaje de modelado unificado (UML)), glosarios, cuadros o bocetos. Los artefactos pueden tener cualquier tamaño y granularidad.

En la Tabla 3 se describe la funcionalidad de los artefactos de la metodología MIC-AGILE, son herramientas para realizar la trazabilidad del trabajo realizado durante el proceso del software, representados en la Figura 1 bloque 4.

Tabla 3. Artefactos de la metodología MIC-AGILE

Artefacto	Descripción
1. Especificación general del proyecto.	Se registra un resumen del proyecto, la necesidad principal de la solución de software y se propone una solución.
2. Definición y especificación de requerimientos técnicos.	En él se registran las especificaciones técnicas y los requerimientos tecnológicos del proyecto.
3. Definición requerimientos específicos.	Especificación de operación, funciones, acciones, diseño, interfaces y alcances que debe de tener el producto de software.
4. Definición de la arquitectura del proyecto.	Se registra el andamiaje del sistema, desde la infraestructura de software, hasta la interconexión que hay con otros sistemas.
5. Identificación de riesgos.	Se registran riesgos identificados en los 4 primeros artefactos, se elabora un plan para mitigarlos, calculado el porcentaje de ocurrencia.
6. Definición de tareas de ingeniería.	Se específica individualmente cada tarea. Se identifican y registran el tiempo estimado, tiempo de codificación y la historia de usuario a la que pertenece.
7. Generación de plan de trabajo.	Se define un calendario de actividades de los objetivos que se van alcanzar. En él se relacionan las tareas de ingeniería con la identificación de riesgos.
8. Registrar avance del proyecto	Se registran las pruebas realizadas al software, la evaluación y el resultado, la entrega del módulo al usuario y la capacitación. Y al finalizar la fecha de término.

La IEEE [11] define una actividad como i) un conjunto de tareas cohesivas de un proceso, ii) un componente del trabajo realizado durante el curso de un proyecto, iii) un cuerpo de trabajo definido a realizar, incluida la información de entrada requerida y la información de salida, iv) colección de tareas relacionadas, v). elemento de trabajo realizado durante la implementación de un proceso. Con base en las definiciones anteriores la Figura 1 ilustra la representación de la metodología MIC-AGILE.

La Figura 3, muestra el diagrama de actividad del proceso de solicitud del proyecto, (Figura 1 bloque 1). Se inicia con la solicitud del cliente, luego el responsable de departamento recibe y analiza la solicitud. Si es rechazada se le informa al cliente, existe la probabilidad de ser replanteada por el cliente. Cuando la solicitud es aceptada se le asigna una prioridad y un responsable de proyecto, se informa al cliente y se utilizan los artefactos 1 y 2 de la Tabla 3. Cuando el responsable es asignado al proyecto, se continúa con la definición de requerimientos específicos, se utiliza el artefacto 3 y se finaliza con la aprobación de los requerimientos por parte del cliente. En la definición de requerimientos, es común encontrar requisitos funcionales que especifican operaciones para el procesamiento, mantenimiento y consulta de datos, como crear, leer, actualizar y eliminar información, en la mayoría de los casos, requieren la definición específica de las reglas del negocio [12].

La Figura 4 muestra el proceso de definición del proyecto, (Figura 1 bloque 1). Inicia con el responsable de proyecto, los requerimientos aceptados son ordenados y priorizados entre el responsable del departamento y del proyecto. Se define y evalúa la arquitectura, utilizando el artefacto 4 de la Tabla 3. Después continua el responsable del proyecto con la definición y priorización de tareas de ingeniería en el artefacto 6. Al término de esta tarea, el responsable de departamento y de proyecto identifican y generan un plan de mitigación de riesgos utilizando el artefacto 5. Se genera un plan de trabajo utilizando el artefacto 7; este plan lo realiza el responsable de proyecto utilizando los artefactos 4, 5, y 6, lo evalúa el responsable de departamento, se informa al cliente del plan de trabajo y finalmente a los integrantes del micro-equipo de desarrollo se les da a conocer el nuevo proyecto.

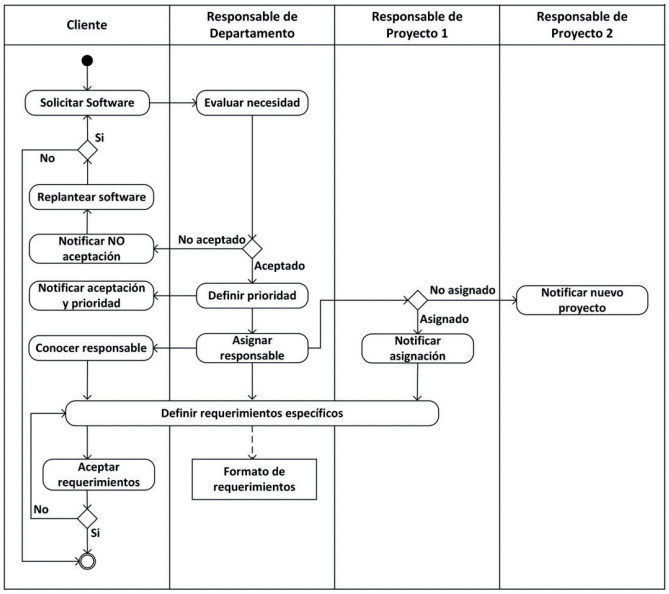

Figura 3. Diagrama de actividad del proceso de solicitud del proyecto

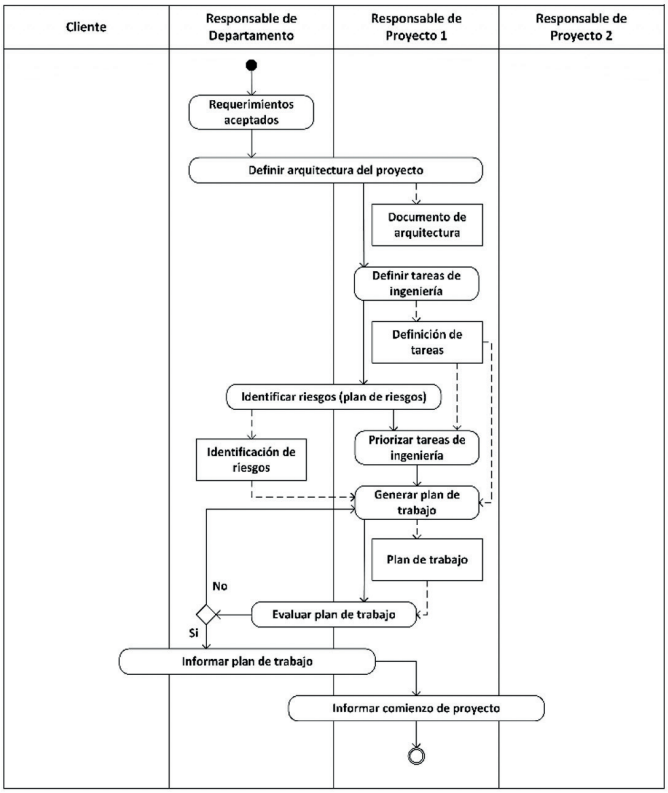

Figura 4. Diagrama de Actividades del proceso de definición del proyecto

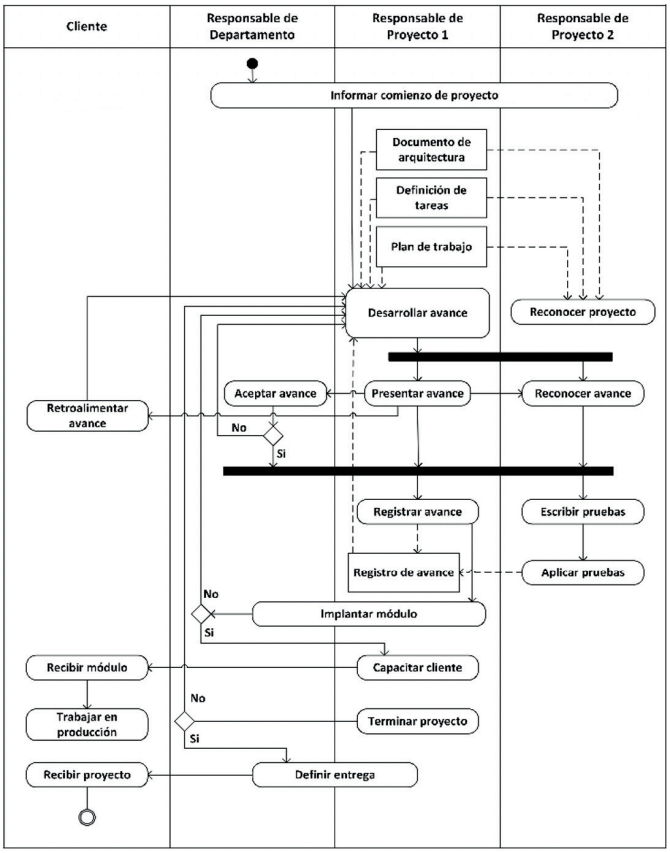

Figura 5. Diagrama de actividad del proceso de desarrollo del proyecto.

La Figura 5 muestra el proceso de desarrollo del proyecto, (Figura 1 bloque 1). Este proceso inicia con la presentación del proyecto a los integrantes del micro-equipo de desarrollo, el responsable de proyecto codifica y programa cada avance siguiendo el plan de trabajo, tomando como referencia los artefactos 1-7 de la Tabla 3, (Figura 1 bloque 4). Al concluir un avance, otro responsable de proyecto evalúa la unidad de software, considerando el modelo de calidad del producto de software ISO/IEC 9126-1 [9], (Figura 1 bloque 5), el avance se le presenta al cliente para recibir realimentación. Se registra el avance en el artefacto 8; al completar la unidad de software, el módulo del sistema se coloca en producción, se informa al cliente y el responsable de proyecto proporciona capacitación al usuario. El ciclo del proyecto continua con cada una de las tareas de ingeniería, hasta completar el plan de trabajo. Terminando el plan de trabajo se define la entrega final del producto de software, representado en la Figura 1 bloque 6.

La Figura 6, muestra el proceso de evolución del proyecto, (Figura 1 bloque 1), lo anterior se relaciona con varios factores externos como son: evolución de los sistemas operativos y frameworks que soportan el producto de software, proyectos de mejora continua en los procedimientos de trabajo de la organización.

El modelo del proceso de evolución (Figura 6), muestra la actualización del producto de software, está se promueve principalmente por dos factores: agregar funcionalidad al software o actualizar algún componente de la base tecnológica que soporta el producto de software (lenguaje de programación, sistema gestor de base de datos, sistema operativo, etc.). Este proceso comienza con una solicitud del cliente.

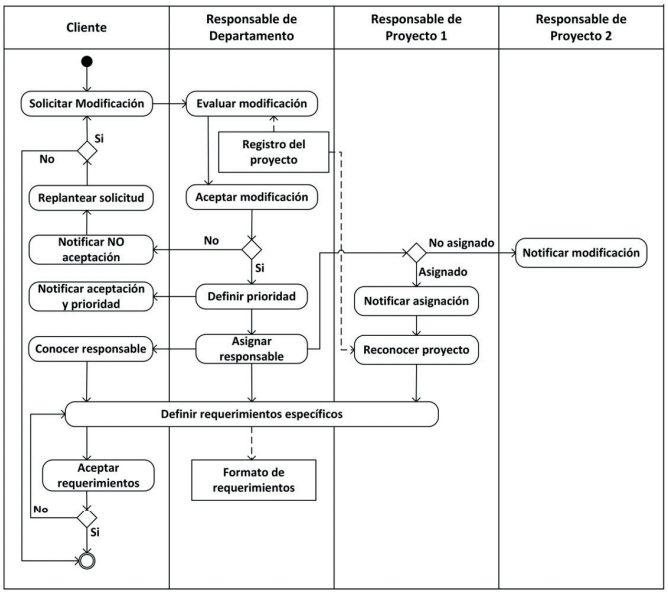

Figura 6. Diagrama de actividad del proceso de evolución del proyecto.

El responsable del departamento evalúa la actualización, si es a solicitud del cliente se le informa que su solicitud ha sido aceptada, en caso contrario hay probabilidad de replantear la solicitud por parte del cliente. Continuando con la actualización o evolución del producto de software, el responsable del departamento define la prioridad y asigna un responsable de proyecto, se procura en lo posible que sea el mismo responsable de proyecto que lo construyó inicialmente. Se retoman los artefactos del proyecto y se definen los requerimientos de la actualización o evolución. Después de acreditar la lista de requerimientos por parte del cliente, se sigue con el proceso de definición del proyecto y luego con el proceso de desarrollo del proyecto.

3. Implementación de la metodología: caso de estudio

Se presenta un caso de estudio como resultado de la aplicación de la metodología MIC-AGILE.

3.1 Caso de estudio

Se diseñó un producto de software denominado "Sistema para la administración de convocatorias de profesor asignatura". Como concepto, una convocatoria se define como una plaza presupuestada que es parte de una entidad académica y tiene asignada una materia y horario. Para ocupar la plaza puede participar cualquier integrante de la sociedad que cumpla con el perfil académico definido. Las plazas presupuestadas las administra la universidad. Este producto de software se compone de tres módulos principales:

- Administración de convocatorias: En este módulo se registran las plazas disponibles y presupuestadas, se define por parte de la entidad académica el perfil académico de cada convocatoria y se asignan los comités evaluadores de las convocatorias.
- Captura de currículum vitae de los aspirantes: el aspirante participa integrando sus documentos comprobatorios.
- Evaluación de los aspirantes: Los miembros de los comités evaluadores asignados por la entidad académica, registran la evaluación y generan las actas de asignación.

Los usuarios del producto de software son:

- Administrador de las convocatorias de la administración central.
- Administrador de las convocatorias en una entidad académica
- Aspirante a una convocatoria.
- Comité evaluador de la convocatoria en la entidad académica.

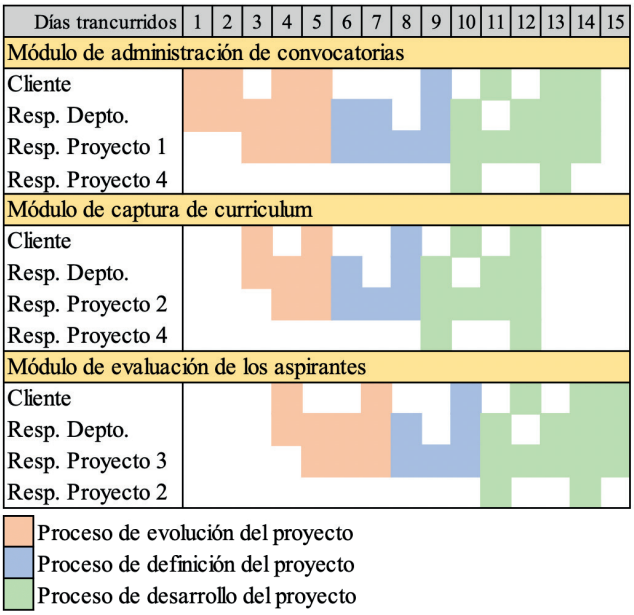

Figura 7. Interacción de los actores en los procesos que integran la metodología MIC-AGILE.

Este producto evolucionó de ser un producto de software embebido en los procesos de la facultad, a ser parte del proceso institucional. Debido a ello no interviene el proceso de solicitud del proyecto. Por el corto tiempo para evolucionar este producto, se utilizó la metodología aplicándola como si fueran tres productos de software independientes, pero al mismo tiempo interconectados entre sí. Se logró la meta en un tiempo de 15 días, atendiendo primero el módulo de administración de convocatorias y en paralelo evolucionar y desarrollar los módulos de captura de currículum y el de evaluación de expedientes. La Figura 7 muestra la interacción en el tiempo, entre los actores y los procesos que integran la metodología MIC-AGILE, aplicándola a los tres módulos que componen el producto de software. Con una

metodología tradicional el proceso del software se enfocaría en la documentación en lugar de al proceso del desarrollo, y tomaría más tiempo poner en producción el producto de software completo y, en consecuencia, se incumple la meta establecida.

4. Discusión

Es importante que los integrantes del micro-equipo tengan una estrecha comunicación entre ellos, antes de realizar una extensa documentación de código, para sea funcional en el proyecto lo antes posible. El cliente es parte fundamental del equipo de trabajo. Debe de ser un cliente embebido en el equipo de desarrollo, para cumplir con los requerimientos, funcionalidad y expectativas del cliente, y que los cambios o ajustes que surgen durante el desarrollo del proyecto sean inmediatamente evaluados y formen parte de la solución.

Es importante que los responsables de proyecto tengan un amplio dominio del lenguaje de programación en el que se realiza el producto de software y conocimiento de programación en los manejadores de bases de datos. Esto disminuye el tiempo de aprendizaje en el proceso del software. Asimismo, contar con actitud propositiva para resolver un problema, poseer conocimiento de varias tecnologías que están en el entorno del producto de software, ya que proporciona habilidades para identificar riesgos potenciales durante el desarrollo del producto. Adicionalmente, conocer a fondo el flujo de información, las reglas del negocio, esté tipo de personal es valioso para la metodología, porque ayudan a resolver dudas en el diseño, y a desarrollar el conjunto de pruebas necesarias para evaluar los resultados.

5. Conclusiones

La metodología MIC-AGILE, se consolida como una herramienta para organizar el micro-equipo de desarrollo utilizando cuatro actores o roles principales, simplifica el proceso de software subdividiéndolo en cuatro procesos (solicitud, definición, desarrollo y evolución), se desarrollan ocho artefactos para llevar la trazabilidad del proyecto de software.

Se logró aumentar la calidad de los sistemas, desarrollar y evolucionar paralelamente más de un producto de software, mejorar los tiempos de entrega, utilizar y especificar en el diseño diferentes frameworks en un proyecto.

Es posible aplicar esta metodología en equipos de desarrollo pequeños, en donde hay que cumplir con tiempos cortos de entrega y no hay presupuesto para contratar personal. La adopción de esta metodología por más de cinco años ha contribuido a lograr madurez y estabilidad en el equipo de desarrollo de software. No fue posible utilizar otra metodología ágil existente por no contar con suficiente personal para asumir los roles que proponen en ellas. Como trabajo futuro se considera la estandarización de los mecanismos de pruebas del software y generar métricas que proporcionen estadísticamente la trazabilidad del proceso. Así mismo se propone iniciar la inclusión de técnicas de generación automática de modelos de software para agilizar el proceso y reducir los plazos de entrega de productos de software. Construir una herramienta de software para administrar el proceso del software.

6. Agradecimientos

A la Facultad de Ingeniería de la Universidad Autónoma de San Luis Potosí, México.

7. Referencias

[1] K. Beck et al., “Manifesto for Agile Software Development,” 2001. [Online]. Available: http://agilemanifesto.org/.

[2] R. Sriram and S. K. Mathew, “Global software development using agile methodologies: A review of literature,” in 2012 IEEE International Conference on Management of Innovation & Technology (ICMIT), Jun. 2012, pp. 389–393. Doi: https://doi.org/10.1109/ICMIT.2012.6225837

[3] G. K. Hanssen, T. Stålhane, and T. Myklebust, “What Is Agile Software Development: A Short Introduction,” in SafeScrum® – Agile Development of Safety-Critical Software, Cham: Springer International Publishing, pp. 11–15, 2018.

[4] C. Ebert and M. Paasivaara, “Scaling Agile,” IEEE Softw., vol. 34(6), pp. 98–103, Nov. 2017. Doi:https://doi.org/10.1109/MS.2017.4121226

[5] M. Waterman, “Agility, Risk, and Uncertainty, Part 1: Designing an Agile Architecture,” IEEE Softw.,vol. 35(2), pp. 99–101, Mar. 2018. Doi: https://doi.org/10.1109/MS.2018.1661335

[6] J. D. Y. González, C. J. P. Calvache, and O. S. Gómez,“Estado del arte de la utilización de metodologías ágiles y otros modelos en pymes de software,” 2016

[7] G. O’Regan, “Agile Methodology,” in The Innovation in Computing Companion, Cham: Springer International Publishing, pp. 15–18, 2018

[8] “ISO/IEC/IEEE International Standard - Systems and software engineering--Life cycle management--Part 3: Guidelines for the application of ISO/IEC/IEEE 12207 (software life cycle processes),” ISO/IEC/IEEE 24748-3:2020(E), pp. 1–76, 2020. Doi:https://doi.org/10.1109/IEEESTD.2020.9238526

[9] K. Esaki, “Verification of Requirement Analysis Method for System Based on ISO/IEC 9126 Six Quality Characteristics,” 2013, pp. 60–68.

[10] P. Ghazi and M. Glinz, "Challenges of working with artifacts in requirements engineering and software engineering," Requir. Eng., vol. 22(3), pp. 359–385, Sep. 2017. Doi: https://doi.org/10.1007/s00766-017-0272-z

[11] "ISO/IEC/IEEE International Standard - Systems and software engineering -- Vocabulary," ISO/IEC/IEEE 24765:2010(E), pp. 1–418, 2010.

[12] L. V. Barcelos and R. D. Penteado, "Elaboration of software requirements documents by means of patterns instantiation," J. Softw. Eng. Res. Dev., vol.5(1), p. 3, Dec. 2017. Doi: https://doi.org/10.1186/s40411-017-0038-9