JCSA 2021

12 y 13 de noviembre de 2021

IV JORNADAS DE CALIDAD DE SOFTWARE Y AGILIDAD

IV

LIBRO DE ACTAS

2021

UNIVERSIDAD NACIONAL DE MISIONES

UTN

Libro de Actas de las Cuartas Jornadas de Calidad de Software y Agilidad / Gladys Noemí Dapozo, Emanuel Irrazábal, María de los Ángeles Ferraro, Horacio D. Kuna, Eduardo Zamudio, Alice Rambo, César Acuña, Verónica Bollati y Noelia Pinto ; compilación de Gladys Noemí Dapozo ; Emanuel Agustín Irrazábal. - 1a ed compendiada. - Corrientes : Universidad Nacional del Nordeste. Facultad de Ciencias Exactas, 2021. Libro digital, PDF

Editorial de la Universidad Nacional del Nordeste (EUDENE)

ISBN 978-987-3619-72-4

1. Software. 2. Jornadas. 3. Argentina. I. Dapozo, Gladys Noemí, comp. II. Irrazábal, Emanuel Agustín, comp.

CDD 004.0711

Fecha de catalogación: 03/01/2022

i-QuAGI: Un enfoque inteligente para la gestión de calidad en proyectos de software ágiles

Jazmín Teng, Lucas Maidana, Nicolás Tortosa, Noelia Pinto, César J. Acuña

[1] Centro de Investigación Aplicada a TIC, CInApTIC
French 414, Resistencia, Chaco, Argentina

{jazminteng, lucasmaidanasoy, nicotortosa, ns.pinto, csr.acn}@gmail.com

Abstract. En la actualidad, el conocimiento constituye un recurso realmente competitivo de las empresas. De hecho, muchas de ellas evalúan la calidad de sus procesos teniendo en cuenta factores como la eficiencia en la gestión del conocimiento. Por lo tanto, una manera de contribuir con las organizaciones en esto, es diseñar sistemas que faciliten el seguimiento de calidad de sus proyectos desde la gestión del conocimiento generado. Así, y a partir de experiencias anteriores de vinculación con diversas organizaciones, se presenta en este trabajo, una primera aproximación al diseño de i-QuAGI, un enfoque inteligente que se incorpora al framework AQF y que permite liberar del trabajo manual a los equipos, ofreciendo recomendaciones automáticas que permitan favorecer la calidad del proceso de desarrollo de software, cuando es guiado por enfoques ágiles.

Palabras clave: Calidad de Software, Enfoques ágiles de desarrollo de software, Sistemas Multi-agente, Proyectos ágiles de software.

1 Introducción

La Ingeniería de Software Inteligente (ISE) se refiere a la aplicación de técnicas inteligentes a la ingeniería de software [1]. Se define una "técnica inteligente" como una técnica que explora datos (de artefactos digitales o expertos en el dominio) para el descubrimiento de conocimientos, el razonamiento, el aprendizaje, la planificación, el procesamiento del lenguaje natural, la percepción o la toma de decisiones de apoyo.

Es posible utilizar técnicas de inteligencia artificial (IA) para ayudar a construir productos de software, sin embargo, el objetivo de la ISE no es ese, sino que esta disciplina se ocupa principalmente de la definición de procesos que permitan incorporar técnicas de IA en los sistemas resultantes [2].

En el escenario del desarrollo de productos de software, una variable fundamental es la gestión del conocimiento generado. De hecho, los equipos responsables deben tener, no solo la experiencia técnica en el campo, sino el conocimiento pertinente para aplicar cualquiera de las metodologías que la Ingeniería de Software demanda para ser aplicada de la mejor manera. La falta de procedimientos y herramientas formales en la

gestión de conocimiento, en el desarrollo de software, impiden su acertada ejecución generando diversos problemas que redundan en la calidad del producto entregado [3]. Asociado a esto, se destaca la popularidad ganada por el uso de enfoques ágiles [4], pues ofrece a los equipos el control de requerimientos variables, la gestión efectiva y eficaz de los grupos de trabajo y el involucramiento y empoderamiento del cliente dentro del proyecto.

Desde hace unos años, resulta imperante la necesidad de la gestión de calidad de los procesos en equipos dedicados al desarrollo de proyectos ágiles[1], por lo que, producto de investigaciones anteriores, se ha presentado el framework AQF [5][6], una propuesta que integra un modelo de calidad (QuAM) junto a una herramienta de software (QuAGI) que permite la automatización de dicho modelo y que ayuda a la gestión de este tipo de proyectos.

En función a resultados obtenidos luego de diversas experiencias de validación llevadas a cabo en ambientes reales, y si bien los equipos participantes han manifestado adaptarse fácilmente a la herramienta de software QuAGI, se ha observado que resulta necesario enriquecer el framework de forma tal de ofrecer una nueva herramienta que permita liberar de trabajo de monitorización manual de los proyectos

Derivado de esta necesidad detectada, en este artículo se presenta una evolución del framework AQF a partir de la incorporación de una primera aproximación al diseño de i-QuAGI, un enfoque inteligente que permita recomendar acciones al equipo de forma tal de mejorar los niveles de calidad del ciclo de desarrollo ajustando los factores que sean necesarios. Se busca, entonces, incorporar al framework AQF una herramienta que dé soporte al equipo de desarrollo, a partir de recomendaciones automáticas que surjan del seguimiento del proyecto ágil y sus actividades, las cuales muchas veces son afectadas por acciones en segundo plano que pasan desapercibidas e impactan negativamente en los niveles de calidad del proceso de desarrollo asociado.

El resto del artículo se estructura de la siguiente forma: la sección 2 describe los trabajos relacionados a la temática abordada; la sección 3 presenta brevemente una primera aproximación al diseño de i-QuAGI; por último, la sección 4 resume las conclusiones a las que se llegó durante la realización de este trabajo y se presentan futuras líneas de investigación.

2 Trabajos Relacionados

En el campo de la ingeniería de software son muchas las investigaciones que se centran en presentar propuestas sobre sistemas inteligentes para el seguimiento de proyectos de software. Por un lado, una mayoría importante de contribuciones se corresponden con “Aplicaciones prácticas”, propuestas que describen soluciones de aplicación práctica dentro del contexto de una organización en particular. Enfocadas en la estimación del esfuerzo cuando el ciclo es guiado por enfoques ágiles. Algunas de estas propuestas se presentan, brevemente, a continuación.

[1] Se utiliza el término “proyectos ágiles” para referenciar proyectos de software cuyos ciclos de desarrollo estén guiados por enfoques ágiles

En referencia a esto último, el trabajo presentado en [7] propone un enfoque que busca mejorar la estimación del esfuerzo del software y la gestión del conocimiento de los proyectos de software al centrarse en el proceso y las prácticas de Scrum utilizando el modelo de ontología en un sistema de estimación de agentes múltiples. La aplicación motiva, además, a guardar regularmente un conocimiento tácito significativo de situaciones únicas en forma de lecciones aprendidas durante el desarrollo del proyecto. En esta propuesta, los agentes del sistema acceden a la base de conocimientos existente y realizan de manera autónoma sus actividades de inferencia utilizando la lógica de descripción según los requisitos especificados por el Scrum Master y le responden con una estimación adecuada en forma de tiempo, recursos y lecciones aprendidas para el éxito de proyectos futuros.

Otro ejemplo de aplicación práctica se expone en [8], donde a partir del uso de técnicas de aprendizaje automático para modelar los tipos de visualización existentes con las historias de usuario correspondientes, se diseña e implementa un prototipo de sistema de recomendación ReVizy para predecir el tipo de visualización recomendado para nuevas historias de usuario.

En el caso del trabajo referenciado en [9], se presenta el desarrollo de un framework que describe la estimación de un desarrollo de software en las etapas iniciales y consta de enfoques de retroalimentación y retroalimentación en todo el ciclo de desarrollo, utilizando para ello técnicas basadas en Redes Bayesianas. El artículo concluye con la implementación de la técnica de estimación mediante la implementación de un pequeño proyecto.

Una propuesta diferente se presenta en [10], donde sus autores comparten un sistema inteligente basado en algoritmos genéticos para resolver el problema de estimación del esfuerzo de desarrollo de software en proyectos ágiles. El trabajo se basa en la morfología matemática, que consiste en una neurona híbrida-artificial (perceptrón Dilation Erosion) extendido desde el concepto de teoría de la red completa, sus autores modifican los modelos existentes realizando adaptaciones al enfoque ágil para la estimación en proyectos de software.

En [11], se propone un sistema inteligente de recomendación y apoyo a la toma de decisiones que da soporte al Scrum Master para estimar mejor un próximo proyecto de software en términos de costo, tiempo y recomendaciones de recursos humanos. La aplicación toma como base las mejores prácticas recolectadas de casos exitosos y datos históricos guardados.

Otro trabajo relacionado que se destaca es el caso de la propuesta presentada en [12], donde se describe un bot de refactorización de software inteligente, llamado RefBot. Integrado en el sistema de control de versiones (por ejemplo, GitHub), el bot supervisa continuamente el repositorio de software y reacciona frente a determinados eventos del ambiente.

Por otro lado, se diferencian también las propuestas que se enfocan en hacer de soporte o gestionar diferentes prácticas ágiles independientemente del enfoque que caracterice al proyecto. Así, un caso, se representa en el estudio referenciado en [13], el cual describe una solución llamada Sponto, una aplicación para automatizar la producción de historias de usuario en base a ontologías, ofreciendo plantillas tanto a desarrolladores como a la comunidad que desee reutilizarlas. Si bien no se focaliza en un único enfoque

ágil, ni en todo el ciclo de desarrollo, esta propuesta se destaca por la posibilidad de reducir el tiempo empleado en recrear artefactos ya utilizados en proyectos de similares características. La herramienta no solo favorece la producción de historias de usuario, sino que las utiliza como entradas para obtener otros artefactos útiles para el desarrollo de software (por ejemplo, scripts de base de datos, diagramas de Business Process Model (BPM), fragmentos de código JAVA, entre otros).

Otra aplicación que considera también la elicitación de requerimientos en proyectos ágiles de software, es la presentada en [14], donde sus autores proponen una ontología de dominio para dar soporte, en contextos de Scrum, al Product Owner durante el proceso de ingeniería de requisitos en la gestión ágil de proyectos de software. La fortaleza de la propuesta radica en que integra la ontología de soporte de información para el proceso de ingeniería de requisitos en la gestión ágil de proyectos y la ontología de dominio de aplicación del producto software que se está desarrollando.

Sin embargo, la mayoría de estas herramientas fallan a la hora de dar soporte a los procesos de toma de decisiones y no ofrecen asistencia en cuanto a identificar qué prácticas o acciones llevadas adelante en la gestión de proyectos de software están afectando negativamente a la calidad del proceso, repercutiendo directamente en la calidad del producto final de software. Esto último se debe a que la mayoría se abocan al seguimiento de un único proyecto y no se detienen en recopilar experiencias de proyectos anteriores, desaprovechando de esta manera todo el conocimiento previamente adquirido. Normalmente, esta experiencia es sólo adquirida por el equipo de desarrollo y cuando éste ya no existe, la información que no esté asociada a la herramienta de seguimiento de proyectos y que se relacione al equipo en sí mismo, se pierde. Esto termina perjudicando a la organización [15] y derivando en lo que se conoce como Amnesia Organizacional (AO).

El artículo que aquí se presenta, expone la primera aproximación al diseño de un enfoque inteligente, que se constituye como una nueva herramienta integrada al framework AQF, con el objetivo de ofrecer una alternativa que permita la gestión del conocimiento y la toma de decisiones en el equipo de modo de evitar realizar acciones que repercutan negativamente en el proyecto de desarrollo basado en enfoques ágiles.

3 Primera aproximación al diseño de i-QuAGI

La herramienta i-QuAGI, representa un sistema multiagente (SMA), compuesto por agentes de software (AS)[2] donde el control del sistema es distribuido, y en el que los agentes comparten un modelo de comunicación, caracterizado por reacciones ante eventos generados a partir del uso de QuAGI por el equipo del proyecto de software y que puedan afectar a la calidad final del proceso ágil subyacente.

Para modelar el SMA, desde su diseño y hasta su implementación, el equipo de trabajo se abocó a obtener los diagramas establecidos en la metodología Agile PASSI [16]. Esta es una versión que, según expresan sus autores, incorpora la filosofía de enfoques

[2] Se corresponde con la descripción de una entidad de software capaz de actuar con cierto grado de autonomía para completar tareas complejas

ágiles de desarrollo de software a la metodología de Proceso para la Especificación e Implementación de Sociedades de Agentes (PASSI por sus siglas en inglés) [17].

La selección de la metodología como herramienta para el diseño de i-QuAGI se basó fundamentalmente en 3 aspectos:

- Agile PASSI se caracteriza por un proceso iterativo y, además, dado que la implementación se encuentra explícitamente dentro del proceso de la metodología, se dan actividades de testing entre iteraciones que permite validar las versiones que se obtienen de cada modelo.
- Agile PASSI es una metodología paso a paso que abarca desde los requerimientos hasta el código para el diseño y el desarrollo de los SMA.
- Agile PASSI logra integrar modelos y conceptos que provienen de la ingeniería orientada a objetos y de la inteligencia artificial basados en la notación de UML, con la cual el equipo de investigación abocado a este proyecto tiene familiaridad.

La metodología Agile PASSI distingue 4 fases:

- Requerimientos del Dominio (Requirenments, en inglés): un modelo de los requisitos del sistema que se compone de dos pasos, Planificación y Descripción de requisitos del dominio.
- Sociedad de Agentes (Agent Society, en inglés): visión de los agentes implicados en la solución, sus interacciones y sus conocimientos sobre el mundo.
- Implementación de código (Code, en inglés): un modelo de dominio de solución a nivel de código.
- Pruebas (Testing, en inglés): descompuesto en el plan de prueba antes de la implementación y la ejecución de pruebas a continuación de ello.

La interacción entre fases y los artefactos que caracterizan a cada una de ellas se pueden observar en la Figura 1, que se expone seguidamente.

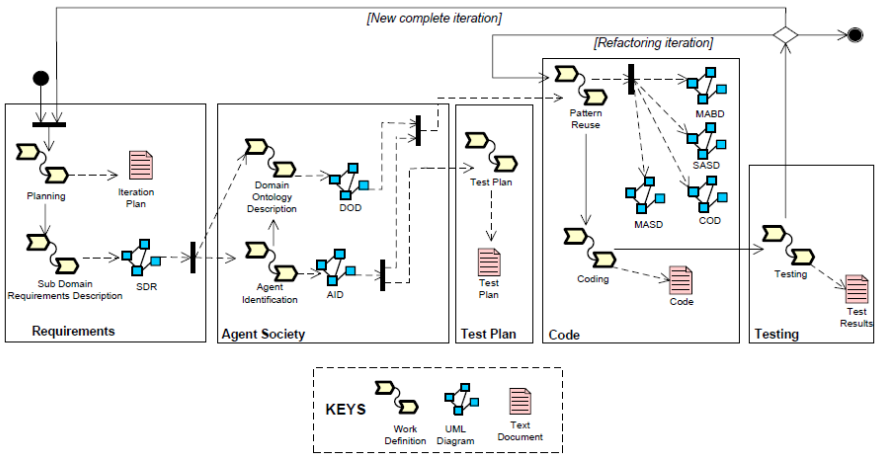

Fig. 1. Proceso Agile PASSI [17]

6

En base a lo anteriormente referenciado, a continuación, se presenta el avance del proyecto i-QuAGI en sus etapas iniciales de modelado.

3.1 Modelo de Requerimientos de Dominio

El Modelo de Requerimientos de Dominio, representa el primero de los 4 propuestos por la metodología PASSI y está compuesto por 2 etapas: Planificación, a partir de la *Descripción del dominio*, y Descripción de Requerimientos a través de los modelos de *Identificación de agentes*, *Identificación de rol* y *Especificación de tarea*.

Haciendo foco en el modelo correspondiente a la *Descripción del Dominio*, éste consiste en expresar los requerimientos funcionales del sistema que se diseña, en términos de diagrama de casos de uso. Actualmente, el equipo se encuentra trabajando en la especificación de los requerimientos del agente Product Backlog, identificando las relaciones con los eventos disparados por el actor Administrador en un contexto de seguimiento de proyecto ágil. De acuerdo a los eventos y las acciones identificadas, se logró expresar dichos requerimientos en el diagrama de caso de uso que se muestra en la Figura 2.

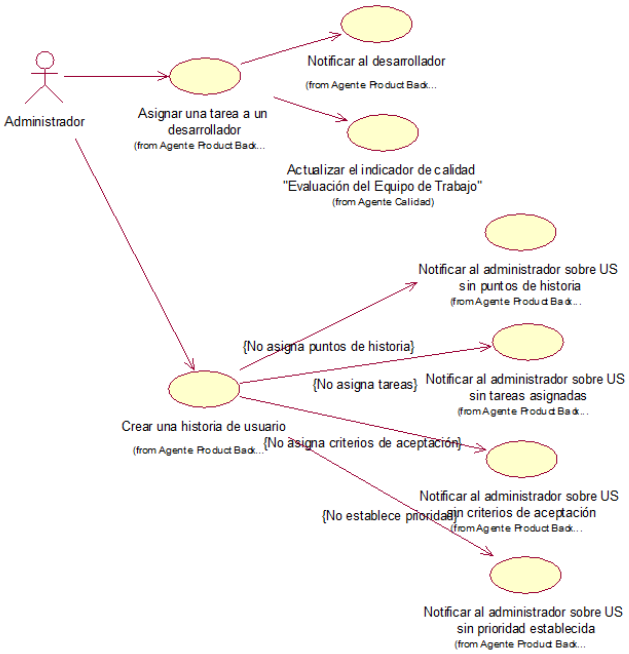

Fig. 2. Esquema inicial de un Diagrama de Descripción del Dominio

A partir de este primer diagrama, y con la ayuda de PASSI Toolkit, se inició con la fase de Requerimientos del dominio, modelando el componente de: *Identificación del Agente*. En este caso, se agruparon las funcionalidades de un agente en un paquete de

caso de uso[3]. Como resultado de este proceso se obtuvo un Diagrama de Identificación de Agentes que se presenta en la Figura 3, donde es posible identificar las funcionalidades y responsabilidades de cada agente. Las entidades externas que interactúan con el sistema se representan como *actores*, que, en este caso, corresponde al rol Administrador.

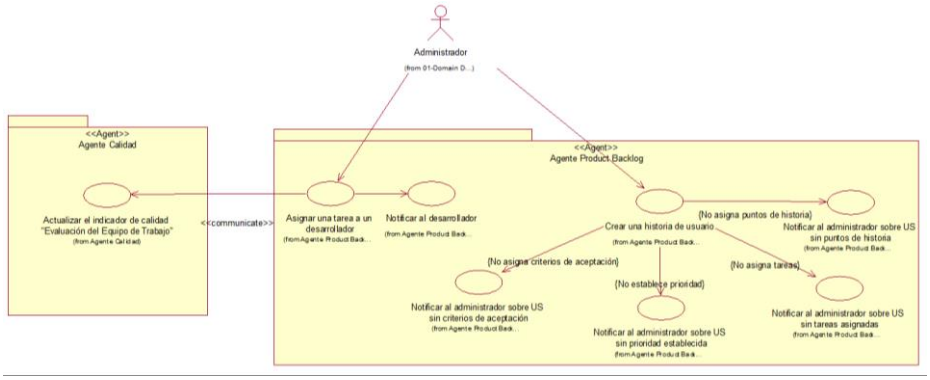

Fig. 3. Esquema inicial de un Diagrama de Identificación de Agentes

Con el modelo anterior obtenido, el equipo se abocó a la obtención del modelo correspondiente a la *Identificación de Roles*. Según la metodología Agile PASSI, el rol es una función que asume temporalmente el agente en su contexto mientras persigue un subobjetivo. Esto significa que un agente puede ocuparse de varios roles funcionales para cumplir con su objetivo.

En este trabajo, se presenta en la Figura 4, una primera instancia ejemplo de comunicación entre los roles identificados en i-QuAGI, a través de un diagrama de secuencia: Agente Product Backlog y Agente Calidad.

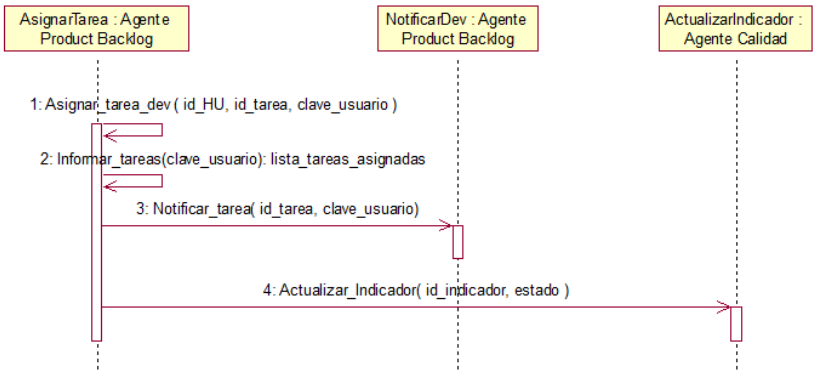

Fig. 4. Esquema inicial de un Diagrama de Identificación de Roles

[3] Para lograr mejor comprensión, se subdivide el diagrama de casos de uso en paquetes. Los paquetes de casos de uso son la forma de agrupar a estos últimos según algún criterio [18]

8

Para completar el modelo de Requerimientos del Dominio, se ha trabajado en el componente de *Especificación de Tarea*, actividad que consiste en la especificación de las capacidades de cada agente mediante diagramas de actividad.

En cada diagrama de actividad correspondiente al componente Especificación de Tarea, se dividen, por un lado, las tareas de un agente en específico y, por otro lado, las tareas de los agentes que interactúan con él. Por ejemplo, en la Figura 5 se incluye el caso que modela la interacción entre el Agente Calidad que, al recibir datos y detectar el evento de asignación de tareas, debe actualizar indicadores del informe de calidad correspondiente.

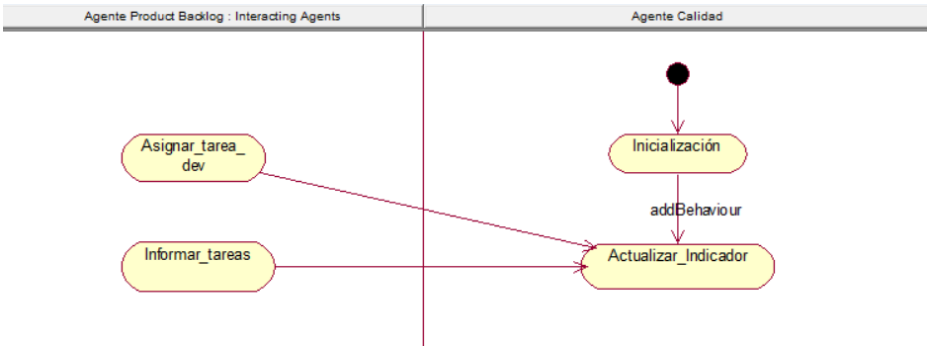

Fig. 5. Esquema inicial de un Diagrama de Especificación de Tarea para el agente Calidad

Si bien aún el Modelo de Requerimientos del Dominio del SMA i-QuAGI se encuentra en revisión, el equipo ha comenzado a organizar y planificar el trabajo correspondiente a la próxima fase del diseño del sistema: Sociedad de Agentes. El objetivo principal de esta etapa se centra en definir una solución orientada a agentes del problema, no solo identificando a cada uno de ellos, sino que estableciendo las interacciones y las dependencias existentes entre los dos.

4 Conclusiones y Trabajos Futuros

En este artículo se ha presentado una aproximación al diseño de un enfoque inteligente que permita recomendar acciones a un equipo de desarrollo de software ágil, de forma tal de mejorar los niveles de calidad del proceso ajustando los factores que sean necesarios. Para ello se ha descripto el avance logrado para la obtención del sistema multiagente, usando la metodología Agile PASSI y exponiendo cada uno de los modelos obtenidos en las fases de diseño del sistema.

El sistema i-QuAGI, se pretende incorporar a una nueva versión del framework AQF, como una herramienta que dé soporte al equipo de desarrollo, a partir de recomendaciones automáticas que surjan del seguimiento del proyecto ágil y sus actividades, las cuales muchas veces son afectadas por acciones en segundo plano que pasan

desapercibidas e impactan negativamente en los niveles de calidad del proceso de desarrollo asociado. Por esto, resulta fundamental el modelado que permita el diseño y revisión del sistema multi-agente desde sus requerimientos hasta su validación.

Como trabajos futuros se prevé continuar desarrollando las fases siguientes, según lo establecido por la metodología Agile PASSI, e iniciar el diseño de estrategias de validación de los modelos que se vayan logrando en cada etapa, a fin de ajustar y mejorar los aspectos que sean necesarios en cada instancia. Esto, a su vez, seguirá enriqueciendo el modelo QuAM y la plataforma QuAGI, de forma tal de añadir modificaciones que contribuyan a una mejor versión del framework AQF.

5 Agradecimientos

Este trabajo se enmarca en las actividades preliminares relacionadas con el proyecto de investigación y desarrollo “I-QuAGI: Un enfoque inteligente para la evaluación de calidad de procesos de software ágiles” (PID SIPPBRE0008092), correspondiente al Centro de Investigación Aplicada a TIC (CInApTIC) de la Universidad Tecnológica Nacional, Facultad Regional Resistencia, de la provincia del Chaco, Argentina.

Referencias

1. Xie, T. (2018, September). Intelligent software engineering: Synergy between AI and software engineering. In International Symposium on Dependable Software Engineering: Theories, Tools, and Applications (pp. 3-7). Springer, Cham.
2. Deugo, D., Oppacher, F., Kuester, J., & von Otte, I. (1999). Patterns as a Means for Intelligent Software Engineering. In IC-AI (Vol. 99, pp. 605-611).
3. Jurado-Muñoz, J. L., & Pardo-Calvache, C. J. (2013). La gestión de proyectos Software, una prospectiva en la aplicación de estrategias en la Ingeniería colaborativa. Lámpsakos, (9), 26-33..
4. PINTO, Noelia, ACUÑA, César et al. Validación de la reingeniería aplicada sobre la primera versión de Agile Quality Framework. XIX Simposio Argentino de Ingeniería de Software (ASSE)-JAIIO 47 (CABA, 2018). 2018.
5. J. Y. González, C. P. Calvache y O. S. Gómez, “Estado del Arte de la Utilización de Metodologías Ágiles y Otros Modelos en Pymes de Software”, Informática-XVI Convensión y Feria Internacional, 2016
6. PINTO, Noelia Soledad. Framework para la evaluación de calidad de proyectos ágiles de software. 2020. Tesis Doctoral. Universidad Nacional de La Plata.
7. M. Adnan y M. Afzal, “Ontology based multiagent effort estimation system for scrum agile method”, IEEE Access, vol. 5, 2017.
8. L. I. U. Xu, “User Story based Information Visualization Type Recommendation System”, International Journal of Information Engineering & Electronic Business, 2019.
9. S. Dhir, D. Kumar y V. B. Singh, “Feedforward and Feedbackward Approach-Based Estimation Model for Agile Software Development. In Advances in Computer and Computational Sciences”, Springer, pp. 73-80, 2017.
10. S. Bilgaiyan, P. K. Panigrahi, y S. Mishra, “Chaos-Based Modified Morphological Genetic Algorithm for Effort Estimation in Agile Software Development. In A Journey Towards Bio-inspired Techniques in Software Engineering”, Springer, pp. 89-102, 2020.

10

11. M. Hamid, F. Zeshan, A. Ahmad, F. Ahmad, M. A. Hamza, Z. A. Khan y H. Aljuaid, “An Intelligent Recommender and Decision Support System (IRDSS) for Effective Management of Software Projects”, IEEE, 2020.
12. V. Alizadeh, M. A. Ouali, M. Kessentini y M. Chater, “RefBot: Intelligent Software Refactoring Bot”, 34th IEEE/ACM International Conference on Automated Software Engineering (ASE)”, 2019
13. K. Athiththan, S. Rovinsan, S. Sathveegan, , N.Gunasekaran, K. S. A. W. Gunawardena y D. Kasthurirathna, “An Ontology-based Approach to Automate the Software Development Process”, IEEE International Conference on Information and Automation for Sustainability (ICIAfS)., 2018
14. M. Murtazina y T. Avdeenko, “The ontology-driven approach to intelligent support of requirements engineering in agile software development”, International Conference on Information Technology and Nanotechnology (ITNT), (pp. 1-6). May. 2020.
15. Kransdorff, Arnold. “Corporate amnesia: Keeping know-how in the company”. Elsevier, 1998.
16. Chella, Antonio, et al. "Agile passi: An agile process for designing agents." International Journal of Computer Systems Science & Engineering 21.2 (2006): 133-144.
17. Cossentino, Massimo, and Colin Potts. "A CASE tool supported methodology for the design of multi-agent systems." International Conference on Software Engineering Research and Practice (SERP'02). 2002.
18. Ivar Jacobson, El Proceso Unificado de Desarrollo de Software, Addison Wesley, 2000.