Asociación Colombiana de Facultades de Ingeniería

Revista Educación en Ingeniería
http://www.educacioneningenieria.org

ÁgilUC: Proceso de desarrollo de software para equipos pequeños y una estrategia para su enseñanza

Sandra Victoria Hurtado-Gil

Facultad de Ingenierías, Universidad de Caldas, Manizales, Colombia. sandra.hurtado@ucaldas.edu.co

Resumen— La norma ISO/IEC 29110 se creó como un modelo para el desarrollo de software para pequeñas entidades, dado que este tipo de equipos cuentan con recursos limitados para mejorar sus procesos. Por otra parte, estos equipos por lo general usan métodos ágiles de desarrollo de software, por sus características de flexibilidad y entrega de valor a los clientes. Buscando relacionar estas dos tendencias, se definió un proceso basado en la norma ISO/IEC 29110, que además incluye prácticas de procesos ágiles, para ser usado en equipos pequeños de desarrollo. Además del proceso, también se definió una estrategia para enseñarlo en los cursos de Ingeniería de Software de la Universidad de Caldas, buscando que los estudiantes puedan aplicarlo en sus proyectos y en sus futuros trabajos. Este artículo presenta la definición general del proceso (ÁgilUC), la estrategia de enseñanza y algunos resultados de su aplicación parcial en un curso.

Palabras Clave— ISO/IEC 29110; procesos de desarrollo software; métodos ágiles; enseñanza de ingeniería de software.

Recibido: 4 de junio de 2019. Revisado: 28 de noviembre de 2019. Aceptado: 4 de diciembre de 2019.

AgilUC: Software development process for small teams and a strategy for its teaching

Abstract— The ISO/IEC 29110 standard was created as a software development model for small entities, given that this kind of teams have limited resources to improve their processes. On the other hand, this teams use, generally, agile software development methods, due to their flexibility and value delivery to clients. Looking to relate these two trends, a process based on ISO/IEC 29110 standard, which also includes agile processes practices, in order to be used by small development teams, was defined. Besides the process, also a strategy to teach it in Software Engineering courses at Universidad de Caldas was defined, seeking that students could apply it in their projects and in their future jobs. This paper presents the general definition of the process (AgilUC), the teaching strategy and some results of its partial application in one course.

Keywords— ISO/IEC 29110; software development process; agile methods; software engineering teaching.

1. Introducción

Los modelos de referencia, de madurez o de calidad de procesos de software, son un conjunto de buenas prácticas que ayudan a las empresas relacionadas con el desarrollo de software a mejorar sus procesos. Existen varios estudios que muestran como los modelos de procesos son de gran utilidad para las empresas, ya que contribuyen al logro de objetivos como: reducción de defectos, aumento de la productividad, mejora en la estimación, satisfacción de los interesados, entre otros [1-4].

A pesar de las ventajas que representan estos modelos, las pequeñas empresas tienen una percepción negativa de ellos, basada en la idea de que son costosos, con mucha documentación y burocracia [5]. Por esta razón se creó la serie de normas ISO/IEC 29110: Ingeniería de software - Perfiles del ciclo de vida de entidades muy pequeñas, que ofrece las ventajas de los modelos de calidad internacionales, pero adaptada especialmente a las características de las pequeñas empresas [6].

En Colombia, un estudio de caracterización del sector de software y tecnologías de la información del año 2015 [7] señala que el 60 % de las empresas encuestadas se consideran Pymes (Pequeñas y medianas empresas), razón por la cual la norma ISO/IEC 29110 es totalmente adecuada para nuestro país.

Como se menciona en [8], las Pymes tienen un gran interés por métodos ágiles, pero también "el interés por integrar practicas ágiles con modelos tradicionales, convencionales, de facto y estándares internacionales ha aumentado considerablemente".

Cabe anotar que la denominación de "métodos ágiles", que se empezó a utilizar desde finales de los años 90, hace referencia a un grupo de procesos y métodos de desarrollo de software que buscan disminuir la cantidad de documentación y burocracia que se encuentra en muchos proyectos de software [9]. Estos métodos ágiles se caracterizan por ser adaptativos más que predictivos y estar orientados hacia las personas más que hacia los procesos. En los últimos años se ha notado una tendencia creciente en el uso de métodos ágiles en la industria [10,11].

Con este panorama, es de gran importancia que la academia se apropie de los conocimientos y sea el canalizador que permita llevarlos a la industria a través de sus egresados. En el caso de la ingeniería de software, tanto los métodos ágiles como la normas ISO/IEC 29110 corresponden a un conocimiento importante que los estudiantes universitarios deben poder conocer y aplicar [12,13]. En los cursos de Ingeniería de Software de la Universidad de Caldas estos temas ya se encuentran presentes en el contenido, pero se trabajan de forma independiente y muchas veces quedan desarticulados con otros temas y proyectos trabajados.

Como citar este artículo: Hurtado-Gil, S.V., ÁgilUC: Proceso de desarrollo de software para equipos pequeños y una estrategia para su enseñanza. Educación en Ingeniería, 15(29), pp. 21-27, Agosto 2019 - Febrero 2020.

© Los autores; licenciado a ACOFI.

Analizando varias propuestas existentes en la literatura, donde se presentan procesos que incluyen elementos de métodos ágiles y de modelos o estándares de calidad [14-19], se encuentra que es pertinente definir un proceso de desarrollo de software basado en elementos de la norma ISO/IEC 29110 y que contemple prácticas de métodos ágiles, el cual pueda ser aplicado en contextos académicos y empresariales. Este proceso, el cual se denomina "ÁgilUC", será aplicado inicialmente en los cursos de Ingeniería de Software de la Universidad de Caldas.

El proceso ÁgilUC servirá como elemento integrador de otras temáticas de los cursos, y podrá ser aplicado por los estudiantes en proyectos de carácter práctico. Por lo tanto, además de la definición del proceso como tal, se contempla una estrategia de integración del proceso en los tres cursos de Ingeniería de Software, y la elaboración de material complementario, como formatos, presentaciones y guías.

En el resto de este artículo se presentarán la descripción general del proceso ÁgilUC, el cual se modeló usando notación BPMN (*Bussines Process Model and Notation*) para la secuencia de las actividades y subprocesos, y una plantilla de texto para el detalle de cada actividad. Posteriormente se incluirá un ejemplo de uno de los materiales complementarios desarrollados, y la visión general de la estrategia de integración del proceso en los cursos. Por último, se presentarán las conclusiones y el trabajo futuro propuesto para aplicar este proceso.

2. Proceso ÁgilUC

2.1. Generalidades

El proceso ÁgilUC comprende un conjunto de actividades, roles y guías para orientar el desarrollo de software para equipos de desarrollo pequeños (de tres a ocho personas) en proyectos de riesgo bajo o medio. Este proceso busca el desarrollo de software de calidad que genere valor a los usuarios y que a la vez permita el crecimiento profesional del equipo.

ÁgilUC tiene como referente principal el grupo de ingeniería de software genérico, perfil de entrada, es decir, la parte 5-1-1 de la norma ISO/IEC 29110 [20]. Este perfil está dirigido a entidades que desarrollan un solo proyecto a la vez, por lo general con duración menor a seis meses; y consta de dos procesos, los cuales representan el núcleo operativo del desarrollo de software:

- Gestión del proyecto (*Project Management* – PM)
- Implementación del software (*Software Implementation* - SI)

Otro referente importante para ÁgilUC es Scrum, uno de los métodos ágiles más conocidos y aplicados [11,21]. Scrum es un marco de trabajo para el desarrollo de productos, que está basado en un conjunto de roles, eventos, artefactos y reglas [22]. En este marco el producto se desarrolla en varias iteraciones (*sprints*) de 2 a 4 semanas de duración, cada una terminando en un incremento funcional, permitiendo así hacer entregas frecuentes de valor a los clientes. Este marco, sin embargo, no establece los aspectos de ingeniería del desarrollo del producto, por lo que es necesario combinarlo con otras prácticas o métodos (tanto ágiles como prescriptivos). En el caso particular de ÁgilUC, se consideran técnicas para el inicio o concepción del proyecto, para la definición de la arquitectura y para las fases "técnicas" de cada iteración (requisitos, modelado, construcción, integración y pruebas).

Las actividades que definen la generalidad del proceso se muestran en la Fig. 1.

A partir de una problemática o de una oportunidad identificada, se contempla una reunión inicial entre todos los interesados, para establecer de manera colaborativa la visión del proyecto, incluyendo un plan preliminar de iteraciones. Esta reunión tiene, internamente, una serie de actividades, basadas en la práctica ágil "Inception Deck" [23].

A continuación, se procede a definir la arquitectura del software y a configurar el entorno, es decir, establecer toda la infraestructura necesaria para el proyecto.

Seguidamente se procede con el desarrollo propiamente dicho, el cual se realiza siguiendo un proceso iterativo e incremental. Las actividades de cada iteración incluyen aspectos técnicos

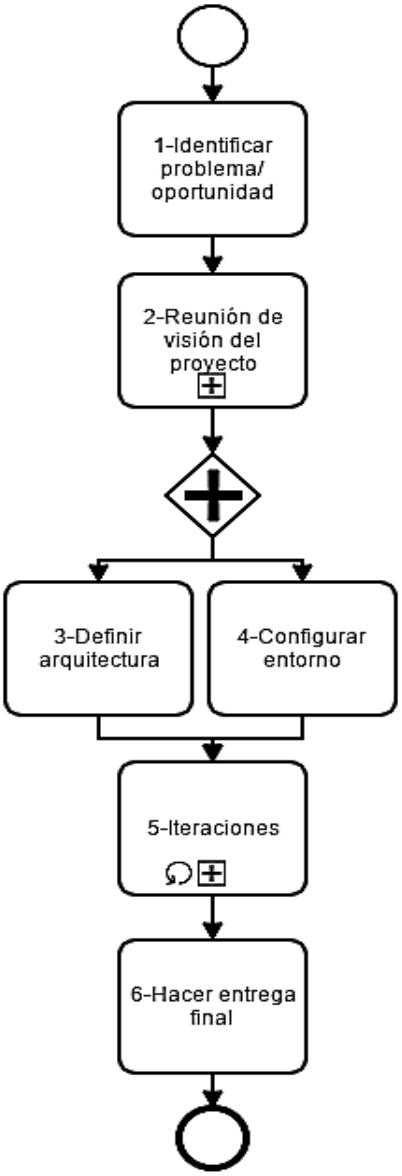


Figura 1. Actividades generales del proceso ÁgilUC.
Fuente: Elaboración propia.

y de gestión, como se observa en la Fig. 2.

Al finalizar cada iteración hay una revisión con el cliente de la funcionalidad desarrollada, y una retrospectiva del equipo para considerar aspectos de mejora continua; y al finalizar el proyecto se hace una entrega formal al cliente.

Siguiendo el esquema que tiene la norma ISO/IEC 29110, para definir el proceso ÁgilUC se consideran: roles, actividades y productos. Cada uno de estos elementos se explican en las siguientes secciones.

2.2. Roles

En ÁgilUC se proponen tres roles principales, los cuales se basan en los roles propuestos en ISO/IEC 29110-5-1-1, que son:

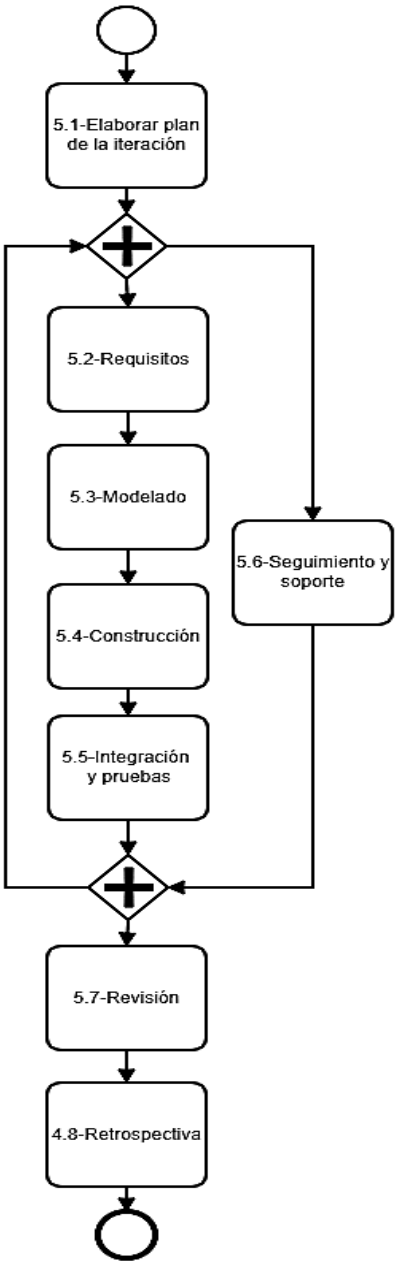


Figura 2. Actividades de una iteración en el proceso ÁgilUC.
Fuente: Elaboración propia.

Tabla 1.
Especificación del rol "Equipo de desarrollo".

Equipo de desarrollo	Este rol lo tiene un grupo de personas, quienes son los encargados de desarrollar el software, realizando para ello actividades de ingeniería (o primarias) y actividades de soporte y gestión. Quienes tengan este rol deben: • Estimar la duración y los recursos necesarios para desarrollar cada uno de los incrementos del software. • Conocer y aplicar prácticas, técnicas y métodos de ingeniería de software, especialmente los relacionados con requisitos, análisis, diseño, implementación, pruebas, implantación, calidad y gestión de la configuración. • Definir, en conjunto con el gestor del proyecto, el proceso que se seguirá, y proponer mejoras al mismo cuando sea pertinente. • Recolectar y analizar las métricas que se requieran para cada proyecto. *Relación con ISO/IEC 29110-5-1-1:* Corresponde al rol de equipo de trabajo (*Work Team* - WT).

Fuente: Elaboración propia.

- Representante del cliente
- Gestor del proyecto
- Equipo de desarrollo

También hay un rol opcional, especialmente para equipos que no han trabajado con este tipo de procesos, que es similar a rol de "coach" que tienen otros procesos como Scrum o TSP. Este rol sería desempeñado por el docente en el caso de contextos académicos, o por un experto (por lo general externo) en el caso de empresas: el Asesor.

Cada rol tiene una especificación, que contempla una descripción general y su relación con la norma ISO/IEC 29110. Por ejemplo, la Tabla 1 muestra la especificación de uno de los roles.

2.3. Actividades

Cada actividad en ÁgilUC tiene los siguientes elementos: objetivo, relación con ISO/IEC 29110-5-1-1, entradas, salidas, roles participantes, descripción y tareas. Además, de manera opcional, puede incluir referencias adicionales.

A manera de ejemplo se muestra (parcialmente) la especificación de la actividad "3-Definir arquitectura", en la Tabla 2

Tabla 2.
Especificación (parcial) de la actividad 3-Definir arquitectura.

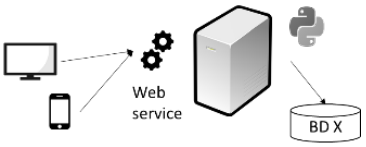
Objetivo:
Elaborar la arquitectura del software, como elemento de diseño de alto nivel que facilite el desarrollo iterativo posterior de los diferentes incrementos.
Relación con ISO/IEC 29110-5-1-1:
Permite cumplir el objetivo SI.O3 y corresponde a la actividad SI.3.
Entradas:
Documento de visión, plan de iteraciones y listado de riesgos.
Salidas:
Documento de arquitectura.
Roles participantes:
Equipo de desarrollo.
Descripción:
A partir del conjunto de funcionalidades, restricciones, límites y riesgos identificados, se define cuál puede ser la mejor forma de organizar o estructurar el sistema, en términos de componentes macro, sus responsabilidades y las relaciones entre ellos, diseñando de esta manera la arquitectura del software.
Tareas:
Revisar los requisitos:
1. Se consulta a los *stakeholders* identificados durante la reunión de visión (pero que no participaron en ella), para complementar la

información de los requisitos, ya sea en términos de ajustes en los casos de uso, restricciones o atributos de calidad.

2. Las restricciones y atributos de calidad refinados se incluyen en el documento de arquitectura. También es posible incluir algún riesgo técnico que esté relacionado con la arquitectura del software.
3. Se revisa el diagrama de casos de uso elaborado durante la reunión de visión, para identificar aquellos casos de uso más importantes, por su complejidad técnica y su prioridad para el negocio, pues estos orientarán la definición de la arquitectura.
   Aquí se recomienda consultar al representante del cliente sobre los casos de uso prioritarios, ampliando su descripción para tener más clara la funcionalidad deseada.
4. El diagrama con los casos de uso seleccionados se incluye en el documento de arquitectura. Este diagrama corresponde a la vista de casos de uso.

Elaborar la visión general de la infraestructura:

5. Tomando como base el entorno de desarrollo definido (lo cual se hace en una actividad paralela), se elabora un diagrama libre que represente los componentes tecnológicos involucrados -de hardware y software- y sus relaciones. Esta vista sirve como un referente rápido de la infraestructura para el equipo de desarrollo.
6. Este diagrama se complementa con el listado de herramientas que se utilizarán, cada una con su número de versión y un enlace a donde se encuentra.



(Continúa hasta el paso 20. No se muestran todos estos pasos por simplicidad).

Referencias adicionales

- A Technique for Architecture and Design. Microsoft Patterns & Practices Team, 2009. https://docs.microsoft.com/en-us/previous-versions/msp-n-p/ee658084%28v%3dpandp.10%29
- Why Do We Need Architectural Diagrams? Ionut Balosin, 2019. https://www.infoq.com/articles/why-architectural-diagrams
- Architecture Review in Agile Development. Sofia Sherman, Irit Hadar, Ethan Hadar y Jay Harrison, 2012. https://resources.sei.cmu.edu/asset_files/Presentation/2012_017_001_24174.pdf

Fuente: Elaboración propia.

2.4. Productos

Los productos corresponden a las entradas y salidas de las diferentes actividades del proceso, empezando por la descripción del problema/oportunidad, hasta la solución de software final. Cada producto tiene una especificación que incluye una breve descripción, la relación que tiene con la norma ISO/IEC 29110 y la información del formato que puede usarse (cuando aplique). En la Tabla 3 se puede observar el ejemplo de la especificación de un producto.

Tabla 3.
Especificación del producto "Documento de arquitectura".

Arquitectura	Documento que presenta un esquema general del software, junto con sus principales vistas arquitectónicas, especialmente la conceptual, la lógica y la dinámica. Estas vistas permiten establecer la estructura macro de los conceptos que se manejarán, la organización que tendrán los compontes del sistema y la forma en la cual se comunicarán entre sí. *Formato*: AgilUC-Formato-Arquitectura *Relación con ISO/IEC 29110-5-1-1:* Corresponde a la Identificación de componentes de software (*Software Component Identification*).

Fuente: Elaboración propia.

1. Descripción del software

Describir el software que se desea desarrollar, haciendo un resumen de los objetivos que se buscan y las características que tendrá.

2. Restricciones y atributos de calidad

Restricciones y riesgos técnicos:

- Las restricciones o riesgos técnicos que afectan la arquitectura, por ejemplo: "El software debe desarrollarse usando el motor de base de datos MySQL, por solicitud del departamento de gestión tecnológica de la institución".

Atributos de calidad:

- Los atributos de calidad establecidos para el software, en lo posible especificados de forma cuantitativa. Por ejemplo: "El sistema debe poder soportar al tiempo entre 150 y 300 usuarios simultáneos ingresando evaluaciones docentes".

3. Visión general de la infraestructura

Incluir un diagrama (en formato libre) que muestre los principales elementos de hardware y software involucrados en el proyecto, como servidores, *frameworks* usados, etc.

Figura 3. Vista parcial del formato del documento de arquitectura.
Fuente: Elaboración propia.

Como se puede observar en el ejemplo, este producto tiene un formato asociado, el cual incluye algunos lineamientos para guiar a los estudiantes en su diligenciamiento (ver Fig. 3).

En las actividades y en los productos, aunque se tiene un buen nivel de detalle, no se presentan los conceptos teóricos asociados. Por ejemplo, aunque el proceso indique que se deben identificar actores y casos de uso, no explica qué es un actor ni un caso de uso, ni tampoco presenta los lineamientos para su identificación. En algunos casos se presentan técnicas concretas, especialmente en aspectos de gestión, pero principalmente como recomendaciones. Esto responde a la flexibilidad promovida por los métodos ágiles, lo cual facilita la adaptación del proceso a diferentes proyectos, y su actualización cuando sea requerido.

3. El proceso ÁgilUC en los cursos de Ingeniería de Software

La definición del proceso consiste en el primer paso, pero también es necesario presentarlo a los estudiantes y relacionarlo con otras temáticas de los cursos de Ingeniería de Software para, por último, poder aplicarlo con éxito en proyectos -tanto académicos como empresariales. Por este motivo, se trabaja en dos aspectos adicionales: hacer un análisis de los objetivos y los contenidos de los cursos de Ingeniería de Software de la institución (son tres cursos), para definir una estrategia que permita integrar ÁgilUC en dichos cursos; y elaborar material de apoyo educativo para complementar esta estrategia.

3.1. Estrategia de integración en los cursos

La estrategia definida se construyó comparando los objetivos definidos para cada curso con los objetivos de las actividades de ÁgilUC (que, indirectamente, corresponden a objetivos de ISO/IEC 29110). Posteriormente se revisan los conceptos relacionados con la actividad, para cruzar con los temas que se deben ver en el curso. Este cruce puede resultar en ajustes en el oren de los temas (en un curso o entre ellos), o en complementar con algún tema no cubierto hasta el momento.

Por ejemplo, para el objetivo "Diseñar aplicaciones computacionales aplicando principios de reutilización y que den respuesta a los requisitos funcionales y no funcionales del sistema", que corresponde al segundo curso de Ingeniería de

Software, se relacionan dos de las actividades de ÁgilUC: 3-Definir arquitectura y 5.3-Modelado. Para la primera actividad se identifican como conceptos asociados: definición de arquitectura de software, vistas de arquitectura, diagramas de casos de uso, diagramas conceptuales, diagramas de secuencia, diagramas de componentes, requisitos funcionales y no funcionales. Los temas de requisitos y diagrama de casos de uso se tratan en el primer curso de Ingeniería de Software, y los diagramas conceptuales se trabajan en el curso de Bases de Datos. Los demás temas, por lo tanto, se deben tratar en el curso que se está analizando. Se encuentra que efectivamente se tratan todos estos temas, excepto el de vistas de arquitectura, por lo que se hace el ajuste correspondiente (cabe anotar que todas las modificaciones quedan pendientes de revisión y aprobación por parte del comité de currículo del programa).

Una visión general de la estrategia, sin contemplar el detalle de los conceptos y temas, se presenta en la Tabla 4.

3.2. Material de apoyo educativo

Los materiales de apoyo educativo son un complemento importante en la práctica docente, pues no solo presentan conceptos, ejemplos y actividades, sino que dan soporte a la diversidad de estudiantes que se pueda encontrar en un salón de clase [24,25]. Por este motivo es importante incluir material docente en la estrategia para la enseñanza (y el aprendizaje) del proceso ÁgilUC.

Se elaboraron dos tipos de materiales: listas de chequeo de lectura y presentaciones. Las listas de chequeo de lectura se crean con el objetivo de orientar la preparación previa del tema por parte de los estudiantes. Antes de cada clase los estudiantes deben estudiar la parte del proceso ÁgilUC que se trabajará, y diligenciar la lista de chequeo para comprobar que han identificado los principales conceptos. Al comenzar la clase se verificarán las respuestas y se aclararán conceptos en caso de ser necesario. Un ejemplo de lista de chequeo de lectura, para el tema de definición de la arquitectura en ÁgilUC, puede verse en la Fig. 4.

Por otra parte, se elaboraron presentaciones con los puntos más importantes de la actividad que se está trabajando, junto con los elementos teóricos que dan soporte a dicha actividad.

Las presentaciones pueden ser usadas durante las clases para reforzar algunos de los conceptos, o también pueden servir como material para preparar los temas o para que los estudiantes repasen los principales puntos de éstos. La Fig. 5 muestra una diapositiva correspondiente a una presentación sobre el tema de arquitectura.

Para verificar la utilidad de estos materiales se realizó una prueba de uso (informal) con un grupo de estudiantes de Ingeniería de Software III, donde se trabajó el tema de "Reunión de visión del proyecto". En ÁgilUC, la reunión de visión es un subproceso que está formado por seis actividades: planeación, determinar objetivos y características, determinar restricciones y alcance, identificar *stakeholders*, identificar y priorizar riesgos y elaborar el plan de iteraciones.

Lista de Chequeo de Lectura – Actividad "Definir arquitectura" de ÁgilUC

Para trabajar adecuadamente el tema de la actividad "Definir arquitectura" del proceso ÁgilUC, se deben tener claros los siguientes aspectos:

Pregunta	S/N	Observaciones
Cuál es el objetivo de la actividad		
Cuáles son las entradas y cómo se obtienen estas entradas		
Cuáles son los resultados o salidas		
Qué relación tiene con ISO/IEC 29110-5-1-1		
Qué pasos deben realizarse en cada una de las siguientes tareas: - Revisar los requisitos - Elaborar la visión general de la infraestructura - Elaborar vista lógica - Refinar vista lógica - Elaborar vista conceptual - Elaborar vista dinámica - Presentar y revisar la arquitectura		
Cómo se relacionan las entradas con las diferentes tareas de esta actividad		
Cuál es el formato para documentar la arquitectura y cuáles secciones tiene		
Investigación: Busque alguna herramienta que permita elaborar los diagramas que se crean en esta actividad		

Lecturas obligatorias:

- Especificación de la actividad "Definir arquitectura" de ÁgilUC, que se encuentra en el campus virtual.

Figura 4. Lista de chequeo de lectura para la actividad "Definir arquitectura" de ÁgilUC.
Fuente: Elaboración propia.

Tabla 4.
Visión general de la estrategia de integración del proceso ÁgilUC en los cursos de Ingeniería de Software.

Curso	Actividad ÁgilUC
Ingeniería de Software I	2-Reunión de visión del proyecto 5.2-Requisitos 5.4-Construcción 5.7-Revisión 5.8-Retrospectiva
Ingeniería de Software II	3-Definir arquitectura 5.1-Elaborar plan de la iteración 5.3-Modelado 5.5-Integración y pruebas 5.6-Seguimiento y soporte
Ingeniería de Software III	1-Identificar problema/oportunidad 4-Configurar entorno 6-Hacer entrega final Otros: ISO/IEC 291100, métodos ágiles y proceso ÁgilUC como un todo

Fuente: Elaboración propia.

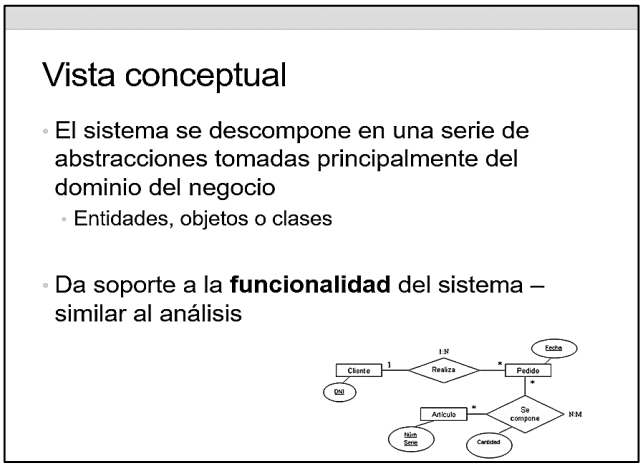
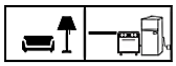


Figura 5. Diapositiva que forma parte de una presentación sobre el tema de arquitectura.
Fuente: Elaboración propia.

Lanzamiento:



Para facilitar la construcción de la solución de software, ésta se puede dividir en lanzamientos (en inglés *release*), que pueden entenderse como avances, módulos o grupos de funciones. Cada lanzamiento debe tener funcionalidad suficiente para que los usuarios puedan trabajar con él en un entorno de producción. El primer lanzamiento por lo general corresponde al "Mínimo Producto Viable" o primera versión de la solución, que sirve para "vender" el producto o motivar a los usuarios, y que, por lo tanto, debe incluir las funciones más prioritarias. Los posteriores lanzamientos van adicionando funcionalidades en orden de prioridad, hasta completar todo el producto.

Figura 6. Ejemplo de un término incluido en el glosario de ÁgilUC.
Fuente: Elaboración propia.

Inicialmente a los estudiantes se les entregó la especificación del subproceso y la lista de chequeo para que las estudiaran antes de clase. Durante la clase se hizo uso de la presentación para mostrar el contexto y las generalidades del subproceso; también se resolvieron algunas dudas. Posteriormente se realizó un taller práctico (que tomó dos sesiones de clase), donde en grupos de cuatro o cinco estudiantes elaboraron la visión para un proyecto que debía tener alguna aplicación en el mundo real.

Al finalizar este ejercicio se aplicó un cuestionario a los estudiantes donde se les realiza una serie de preguntas enmarcadas en dos aspectos: el conocimiento que tenían sobre las actividades del subproceso y los conceptos que no habían quedado claros. De 18 estudiantes que respondieron el cuestionario, tres no respondieron correctamente las preguntas relacionadas con las actividades del subproceso. De estos tres estudiantes, dos mostraron conocer algunos aspectos generales, pero no el detalle del subproceso. Los demás estudiantes respondieron acertadamente las preguntas relacionadas con las actividades de la reunión de visión de ÁgilUC. En cuanto a los aspectos que no quedaron claros, ocho estudiantes no señalaron ningún aspecto (es decir, todo lo consideraron claro), tres mencionaron el tema de objetivos cuantificables, cuatro la distribución de tiempos y los tres restantes mencionaron algunas dudas con respecto a términos como iteración y lanzamiento.

Aunque no se trataba de una prueba formal contralada, esta primera aproximación al uso de los materiales resultó en una percepción muy positiva por parte de los estudiantes (con un 83 % de estudiantes que apropiaron el tema), y también permitió identificar posibles mejoras en la definición del proceso. Para este ejercicio en particular, se complementó el proceso incluyendo algunos ejemplos y adicionando un glosario de términos. Un ejemplo de uno de los términos del glosario puede verse en la Fig. 6.

4. Conclusiones y trabajos futuros

4.1. Conclusiones

El proceso ÁgilUC combina dos temáticas de gran importancia para el desarrollo de software, como lo son los métodos ágiles y los estándares de calidad, lo cual no solo permite apropiarse de los avances en la Ingeniería de Software, sino que además responde a una necesidad sentida de las pequeñas empresas que buscan mejorar sus resultados.

Tratando específicamente de la norma ISO/IEC 29110, existen varias iniciativas para articular esta norma con diferentes métodos ágiles, especialmente para su aplicación en contextos empresariales, lo cual muestra la pertinencia de este tipo de iniciativas. Este proyecto presenta un aporte valioso dado que tiene todo el detalle de un proceso completo (incluyendo formatos y material complementario), facilitando su aplicación en equipos de desarrollo pequeño y contribuyendo al cumplimiento de referentes internacionales que no son fáciles de apropiar por este tipo de equipos.

El proceso tiene un importante nivel de detalle, pero también permite que se cambien las técnicas usadas en los pasos y actividades que lo conforman, como promulgan los métodos ágiles. Esto permite que los equipos de desarrollo puedan seleccionar técnicas diferentes para adaptarse a las particularidades de un proyecto.

A partir del ejercicio donde se aplicaba la actividad de "Reunión de Visión del Proyecto" en un curso de Ingeniería de Software fue posible confirmar, de manera preliminar, su aplicabilidad, al igual que la pertinencia del proceso para diferentes contextos, pues los análisis que realizaron los estudiantes eran totalmente aplicables en entornos empresariales.

ÁgilUC facilita la integración de muchas temáticas relacionadas con desarrollo de software, especialmente mediante su aplicación en proyectos, lo cual es uno de los aspectos más difíciles de lograr en los cursos de Ingeniería de Software. Los comentarios preliminares de los estudiantes son positivos al respecto, pues valoran no solo los materiales y talleres, sino también el poder contar con una guía detallada para llevar a cabo actividades de desarrollo de software.

4.2. Trabajos futuros

Se espera poner en práctica la estrategia definida en los cursos de Ingeniería de Software, no solo para complementar las temáticas que ya se trabajan en la actualidad, sino también para integrar muchas de estas temáticas mediante la aplicación de un proceso disciplinado y ágil en proyectos de desarrollo.

Después de aplicar el proceso en los cursos de Ingeniería de Software se espera extender la aplicación del proceso a otros contextos como prácticas empresariales y cursos de posgrado. A mediano plazo también se espera llevar el proceso a micro y pequeñas empresas de software de la región.

Otro aspecto que se debe continuar trabajando es la elaboración de diferentes tipos de materiales educativos que soporten la estrategia de integración de ÁgilUC en los cursos. Es importante considerar que tanto los materiales como el proceso mismo deben estar en constante actualización, para responder a los cambios en el mercado y en la disciplina.

Referencias

[1] Simoes C. and Montoni, M., Applying statistical process control in small sized evolutionary projects: results and lessons learned in the implementation of CMMI-DEV maturity level 5 in Synapsis Brazil, Journal of Software Engineering Research and Development, 2(2), 2014. DOI: 10.1186/2195-1721-2-2

[2] Kalinowski, M., Weber, K., Franco, N., Barroso, E., Duarte, V., Zanetti, D. and Santos, G., Results of 10 years of software process improvement in Brazil Based on the MPS-SW Model, in: 2014 9th International

Conference on the Quality of Information and Communications Technology, IEEE Computer Society, 2014. DOI: 10.1109/QUATIC.2014.11

[3] CMMI Institute, "Who uses CMMI", [En línea]. 2017. Disponible en: http://cmmiinstitute.com/who-uses-cmmi.

[4] Serrador, P. and Pinto, J., Does Agile work? — A quantitative analysis of agile project success, International Journal of Project Management, 33, pp. 1040-1051, 2015. DOI: 10.1016/j.ijproman.2015.01.006

[5] O'Connor, R. and Laporte, C., The evolution of the ISO/IEC 29110 set of standards and guides, International Journal of Information Technologies and Systems Approach, 10(1), pp. 1- 25, 2017.

[6] Laporte, C., Séguin, N., Villas, G. y Buasung, S., Pequeñas empresas de tecnología: aprovechando las ventajas de las normas de ingeniería de software y sistemas, *ISO Focus+*, [en línea]. pp. 32-36, febrero 2013. Disponible en: http://www.iso.org/sites/edumaterials/focus/iso-pequenas-empresas-de-tecnologia.pdf

[7] Fedsoft, Informe de caracterización del sector de software y tecnologías de la información en Colombia, [en línea]. 2015. Disponible en: https://fedesoft.org/noticias-fedesoft/disponible-estudio-de-caracterizacion-de-la-industria-del-software-colombiano/

[8] Yepes, J.D., Pardo, C.J. y Gómez, O.S., Revisión sistemática acerca de la implementación de metodologías ágiles y otros modelos en micro, pequeñas y medianas empresas de software, Revista Tecnológica ESPOL, [en línea]. pp. 464-479, diciembre 2015. Disponible en: http://www.rte.espol.edu.ec/index.php/tecnologica/article/view/454

[9] Fowler, M., The New Methodology, [En línea]. 2015. Disponible en: https://www.martinfowler.com/articles/newMethodology.html.

[10] Martin, A., Anslow, C. and Johnson, D., Teaching agile methods to software engineering professionals: 10 years, 1000 release plans. de International Conference on Agile Software Development, 2017. DOI: 10.1007/978-3-319-57633-6_10

[11] Peláez, L.E., Toro, A., López, J.F. y Ramírez, A., Caracterización del proceso de desarrollo de software en Colombia: una mirada desde las PYMES productoras, Revista Páginas, [en línea]. (9)2, pp. 89-98, 2012. Disponible en: http://biblioteca.ucp.edu.co/ojs/index.php/paginas/article/view/275

[12] Kropp, M. and Meier, A., Teaching agile software development at university level: values, management, and craftsmanship, de 26th International Conference on Software Engineering Education and Training (CSEE&T), 2013. DOI: 10.1109/CSEET.2013.6595249

[13] Laporte C.Y. and O'Connor, R.V., Software process improvement in industry in a graduate software engineering curriculum, Software Quality Professional Journal, 18(3), pp. 4-17, 2016.

[14] Camacho, W.A.A., Jiménez-Builes, J.A. and Gaviria-Giraldo, J., SoftRace—The software development race under the SEMAT kernel, In: Software Engineering: methods, modeling and teaching, vol. 3, Facultad de Minas, Universidad Nacional de Colombia, 2014, pp. 91-95.

[15] Laporte, C.Y., International software engineering standards applied in undergraduate and graduate software quality assurance courses, IEEE Standards University E-Magazine, [online]. 5(1), 2015. Disponible en: http://espace2.etsmtl.ca/id/eprint/12782

[16] Galván-Cruz, S., Mora, M. and O'Connor, R., A means-ends design of SCRUM+: an agile-disciplined balanced SCRUM enhanced with the ISO/IEC 29110 Standard, in: International Conference on Software Process Improvement, 2017. DOI: 10.1007/978-3-319-69341-5_2

[17] Salazar-Bermúdez, G., Desafíos del curso de Ingeniería de Software, Revista Educación en Ingeniería, [en línea]. 7(13), pp. 32-43, 2012. Disponible en: https://educacioneningenieria.org/index.php/edi/article/view/33

[18] Pasini, A.C., Esponda, S., Boracchia M. y Pesado, P.M., Q-Scrum: una fusión de Scrum y el estándar ISO/IEC 29110. En: XVIII Congreso Argentino de Ciencias de la Computación, 2013. Disponible en: http://sedici.unlp.edu.ar/handle/10915/32421

[19] Yépes-González, J.D., AgileFM: modelo de desarrollo ágil formal basado en la ISO/IEC 29110 para las micro, pequeñas y medianas empresas, Universidad EAFIT, [en línea]. 2017. Disponible en: http://hdl.handle.net/10784/11831

[20] ISO/IEC, Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 5-1-1: Management and engineering guide: Generic profile group: Entry profile, [en línea]. 2012. Disponible en: https://standards.iso.org/ittf/PubliclyAvailableStandards/c060389_ISO_IEC_TR_29110-5-1-1_2012(E).zip

[21] Hummel, M., State-of-the-Art: a systematic literature review on agile information systems development, en: 47th Hawaii International Conference on System Science, 2014. DOI: 10.1109/HICSS.2014.579

[22] Schawaber, K. and Sutherland, J., The Scrum Guide. 07 2016. [En línea]. Disponible en: http://www.scrumguides.org/docs/scrumguide/v2016/2016-Scrum-Guide-US.pdf.

[23] Pastrana, M., Ordóñez, H., Ordóñez, A. and Merchan, L., Requirements elicitation based on inception deck and business processes models in scrum, in: Colombian Conference on Computing, 2017. DOI: 10.1007/978-3-319-66562-7_24

[24] Krajcik, J. and Delen, I., The benefits and limitations of educative curriculum materials, Journal of Science Teacher Education, 28(1), pp. 1-10, 2017. DOI: 10.1080/1046560X.2017.1279470

[25] Foster, E., Design principles guide educators in choosing and using curriculum materials, The Learning Professional, [online]. 39(1), pp. 20-23, 2018. Available at:: https://learningforward.org/wp-content/uploads/2018/03/design-principles-guide-educators-in-choosing-and-using-curriculum-materials.pdf

S.V. Hurtado-Gil, recibió el título de Ing. de Sistemas y Computación, con énfasis en administración e informática en 1997, en la Universidad ICESI, Cali, Colombia, y MSc. en Ingeniería de Sistemas y Computación en 2000, en la Universidad de los Andes, Bogotá, Colombia. Se desempeñó como analista y desarrolladora al comienzo de su carrera, y posteriormente se vinculó con la academia como docente. Estuvo vinculada a la Universidad ICESI, de Cali, a la Universidad Católica de Manizales y a la Universidad Autónoma de Manizales. Desde el año 2017 se encuentra vinculada como docente de planta en la Universidad de Caldas, Colombia.
ORCID: 0000-0003-0788-5086