UNIVERSIDAD CATÓLICA "NUESTRA SEÑORA DE LA ASUNCIÓN"

PROYECTO FINAL DE CARRERA

Desarrollo de un sistema de gestión de programas de estudio orientado a resultados pedagógicos

Autor:
Luis F. VILLALBA V.

Facultad de Ciencias y Tecnología
Departamento de Electrónica e Informática

28/05/2017

Desarrollo de un sistema de gestión de programas de estudio orientado a resultados pedagógicos

---

Autor:
Luis F. VILLALBA V.

Tutores:
Ing. Sebastián ORTÍZ
Ing. Wilfrido INCHAUSTTI

Una tesis presentada en cumplimiento de los requisitos para el título de Ingeniería Informática en la

Universidad Católica "Nuestra Señora de la Asunción"
Facultad de Ciencias y Tecnología
Departamento de Electrónica e Informática

28/05/2017

Declaración de autoría

Yo, Luis F. VILLALBA V., declaro que la tesis titulada, "Desarrollo de un sistema de gestión de programas de estudio orientado a resultados pedagógicos" y todo el trabajo presentado son de mi propiedad. Además, confirmo que:

- Este trabajo fue llevado a cabo bajo las normas y regulaciones del Reglamento De Proyecto Final de Carrera (09/2015) del Departamento de Ingeniería Electrónica e Informática.
- Este trabajo fue realizado en su totalidad con fines investigativos y para obtener el título antes mencionado en esta universidad.
- Cuando cualquier parte de esta tesis haya sido previamente sometida a un título en esta universidad o cualquier otra institución, esto se ha manifestado claramente.
- Donde he consultado trabajo publicado por otros autores, eso ha sido siempre claramente atribuido.
- Donde he citado trabajo de otros autores, la fuente siempre se ha dado. Con la excepción de dichas citas, esta tesis es enteramente mi propio trabajo.
- He reconocido todas las fuentes principales de ayuda.
- Donde la tesis está basada en mi trabajo en conjunto con otros autores, he expresado claramente que fue hecho por ellos y en que he contribuido.

Firma:

Fecha:

UNIVERSIDAD CATÓLICA "NUESTRA SEÑORA DE LA ASUNCIÓN"

Resumen

Facultad de Ciencias y Tecnología
Departamento de Electrónica e Informática

Ingeniería Informática

Desarrollo de un sistema de gestión de programas de estudio orientado a resultados pedagógicos

by Luis F. VILLALBA V.

Agradecimientos

The acknowledgments and the people to thank go here, don't forget to include your project advisor…

Índice General

Lista de Figuras

Lista de Tablas

List of Abbreviations

AMS **A**ssessment **M**anagement **S**ystem
CMS **C**urriculum **M**anagement **S**ystem

Capítulo 1

Introducción

1.1 Justificación

En la actualidad, la tecnología forma parte de nuestra vida cotidiana donde el software ha avanzado con el paso del tiempo. La tendencia apunta a que las aplicaciones puedan ser accedidas en cualquier momento desde cualquier lugar, ya sea en un viaje de negocios o haciendo compras desde cualquier dispositivo con acceso a internet.

Últimamente, de manera a facilitar el acceso a la aplicación por parte de los usuarios, las aplicaciones web se han vuelto populares porque estas se ejecutan en un navegador y no es necesario descargar ningún tipo de software adicional debido a que el peso principal de la aplicación se ejecuta remotamente.[1].

Al mismo tiempo, en la educación han surgido avances tecnológicos aplicables donde se aprovecha la misma para potenciar el aprendizaje adquirido en los estudiantes. Con esto se busca idear nuevas técnicas de evaluación para impulsar un aprendizaje significativo en los alumnos, pero en muchos casos nos encontramos que la forma de evaluar los cursos o actividades no reflejan los conocimientos o capacidades de los alumnos. Para llenar este vacío surge la evaluación basada en competencias, la cual se enfoca en aquellas adquiridas por un estudiante en el proceso de un programa educativo[2].

Así, las aplicaciones de evaluación académica basadas en competencias han adquirido mucha importancia en los últimos años[2]. En dichas aplicaciones se buscan conocer las fortalezas y debilidades del estudiante de una manera modular, en comparación a los métodos cuantitativos de evaluación. Dichos aprendizajes y competencias son expresados por segmentos de estudios o actividades, mediante resultados esperados medibles a nivel institucional, de programa, grado, o de curso; expresados en calificaciones[3].

Para las personas ajenas al entorno educativo, en específico aquellas que no participan directamente en el diseño de planes y programas, puede resultar un tanto complejo comprender el proceso del diseño curricular y reconocer la importancia de involucrar a todas aquellas autoridades que forman parte del mismo. Por esta razón, definir las bases teóricas que servirán de referencia para el diseño curricular, describir sus conceptos básicos, y distinguir sus elementos, es fundamental para el desarrollo del curso[4].

En el ámbito de las aplicaciones académicas basadas en competencias, si bien existen aplicaciones que abarquen el diseño y la emisión de planes de estudio por parte de los profesores o encargados de las universidades, además de su revisión y posterior aprobación por el comité curricular, no se

ha encontrado durante el proceso de investigación una aplicación que integre todos estos a un sistema de gestión de evaluaciones basadas en competencias.

1.2 Objetivos

1.2.1 Objetivo general

Diseñar una aplicación que permita integrar y estructurar tareas separadas de un sistema académico para una gestión de programas educativos orientados a resultados pedagógicos, que provea soporte a flujos de trabajo para sus diferentes etapas de aprobación.

1.2.2 Objetivos específicos

- Realizar un relevamiento de los requerimientos, diseño e implementación de un sistema de gestión de programas orientado a competencias.
- Analizar el impacto de la herramienta en los usuarios.
- Realizar el proyecto en un marco de programación ágil, estimando y desarrollando la aplicación de acuerdo a las directrices brindadas por esta metodología[5].

1.3 Diseño de la investigación

El presente trabajo ilustra como caso de estudio la aplicación de la metodología ágil en la gestión de un proyecto de desarrollo de un módulo, integrado a un sistema de gestión y evaluación de competencias de una organización ubicada en los Estados Unidos. El trabajo a realizar fue propuesto por la organización que brinda el sistema de gestión de competencias, donde el mismo sirve como base al módulo de gestión curricular a desarrollarse.

El caso refleja de manera práctica como debería operar un ambiente de desarrollo ágil y de la interrelación entre usuarios ubicados en diferentes localidades como parte del sistema de trabajo. En este proyecto los miembros ubicados en forma remota se encargan del diseño de las tareas, aquí llamadas historias de usuario, a realizarse, y de la validación de las características entregadas.

Debido a que los requisitos de la aplicación a desarrollarse se irán esclareciendo acorde se vayan completando cada valor de negocio, se optó por la metodología ágil como técnica de gestión de desarrollo de software, puesto que es el enfoque más adecuado para poder encarar esta problemática.

La observación participante de parte del investigador, es un paso inicial para el desarrollo del sistema en un nuevo equipo, donde se identifican y guían las relaciones con los informantes, lo ayuda a sentir la organización del equipo y las prioridades de desarrollo. También, lo permite integrarse con los demás miembros del mismo y de esa manera le facilita el proceso de investigación, además de proveerle una cantidad de interrogantes a ser dilucidadas con los participantes[6].

El primer paso es adaptarse a los cambios y a la metodología de trabajo del equipo de desarrollo. Dicho equipo se constituirá luego de a una

encuesta previa de capacidades de todos los desarrolladores de la empresa, donde cada uno colocará sus conocimientos en herramientas o en partes del sistema de gestión de competencias, para así poder aprovechar las virtudes de cada persona. Acto seguido se procede al análisis de las herramientas a utilizarse para corroborar que cumplen con las exigencias del sistema a desarrollarse.

Entre algunas técnicas de recolección de información se utilizan las encuestas[7]. Para este sistema un equipo en Estados Unidos utilizó la misma para proporcionar una visión general del área a ser investigada.

Los casos de estudios con enfoque HCI[1] tienen como meta la comprensión de problemas o situaciones mediante la exploración. Además, se busca una documentación descriptiva del sistema y de su proceso de desarrollo, que apunta a la evolución del modelo propuesto durante el diseño de la misma, y, por último, demostrar que la herramienta fue utilizada de manera exitosa mediante demostraciones al usuario o validaciones por parte del mismo[8].

Por lo tanto, el caso de estudio propuesto es uno con enfoque HCI que utiliza la observación participante como método adicional de recolección de información.

1.4 Dominio de la problemática

El proyecto objetivo a desarrollarse, utiliza como base una aplicación web AMS[2] que integrará un módulo de gestión curricular a la misma. También se busca la ubicuidad en dispositivos inteligentes mediante la nube. Los dispositivos inteligentes que van a poder utilizar la aplicación se definirán durante el proceso de desarrollo de la misma. Los AMS son utilizados por las universidades en Estados Unidos para evaluar las competencias adquiridas de los estudiantes durante el proceso de su carrera o grado.

En las aplicaciones dirigidas al ambiente educativo, el hecho de utilizar nuevas tecnologías no asegura una mejor UX[3], es por eso que introduciremos durante el desarrollo de la aplicación el concepto de HCI[8], donde ésta se encarga de establecer los patrones de diseño e interacción a seguir a la hora de construir cada uno de los componentes de la aplicación a ser desarrollada.

Un requisito no funcional que forma parte de la infraestructura del caso de estudio es la utilización de la arquitectura SaaS[4], ya que provee servicios bajo un modelo de pagos de suscripción por las diferentes características que la misma ofrece.

Dentro de la metodología ágil, se definirán épicas a desarrollarse durante el periodo de diseño de la aplicación y las historias de usuario que están contenidas en las mismas, para que el equipo de desarrollo pueda entregar valor de negocio del módulo integrado en iteraciones cortas de dos semanas.

[1] de sus siglas en inglés, Human-Computer Interaction, que significa en español interacción humano-computador.

[2] de sus siglas en inglés, Assessment Management System, que significa en español sistema de gestión de evaluaciones.

[3] de sus siglas en inglés, User eXperience, que significa en español experiencia de usuario.

[4] de sus siglas en inglés, Software as a Service, que significa en español software como servicio.

Capítulo 2

Marco teórico

2.1 Evaluación basada en competencias

La creciente profesionalización trajo al campo educativo elementos evaluativos tales como calidad, equidad, competitividad, eficiencia y eficacia; junto con ellos surgieron las competencias, que pasaron a jugar papel importante en el contexto educativo. En la formación de profesionales, resalta la necesidad de reflexionar sobre los aprendizajes que se ofrecen en las instituciones educativas, las cuales deben servir al estudiante para ser útil a la sociedad, que es su entorno inmediato [3].

De esa necesidad va surgiendo la idea de las evaluaciones orientadas a competencias. Cabe resaltar cómo se desenvuelve el aprendizaje basado en competencias usando aplicaciones como herramientas para la evaluación de estudiantes, mediante el análisis de los aportes que introduce la tecnología en este campo, que modifican significativamente las prácticas tradicionales[9].

Uno de los factores de motivación relevantes para el aprendizaje es la evaluación. Cada actividad ofrece a los estudiantes la oportunidad de conocer cuáles son sus resultados de aprendizaje en lo que se refiere al "qué" se ha aprendido y al "cómo" habría podido hacerse. Cualquier proceso de evaluación debería ser diseñado teniendo en cuenta este principio básico.

En un sistema de gestión de evaluaciones basado en competencias, los encargados hacen evaluaciones según las evidencias obtenidas de diversas actividades de aprendizaje, que definen si un estudiante alcanza o no los requisitos recogidos por un conjunto de indicadores en un determinado grado. Una evaluación por competencias asume que pueden establecerse indicadores posibles de alcanzar por los estudiantes, que diferentes actividades de evaluación pueden reflejar los mismos indicadores[10].

La evaluación por competencias ofrece nuevas oportunidades a los estudiantes al generar entornos significativos de aprendizaje que acercan sus experiencias académicas al mundo profesional, y donde pueden desarrollar una serie de capacidades integradas y orientadas a la acción, con el objetivo de ser capaces de resolver problemas prácticos o enfrentarse a situaciones cotidianas [9].

2.2 Sistemas de gestión de aprendizajes

Las plataformas LMS[1] son espacios virtuales de aprendizaje orientados a facilitar la experiencia de aprendizaje a distancia, donde permite una

[1] de sus siglas en inglés, Learning Management System, que significa en español sistema de gestión de aprendizajes.

mejor interacción de profesores y alumnos. También se pueden hacer evaluaciones, intercambiar archivos y participar en foros y chats, además de otras herramientas de interacción estudiante a profesor.

La centralización y automatización de la gestión del aprendizaje es una de las principales características de los LMS. La plataforma puede ser adaptada tanto a los planes de estudio de la institución como a los contenidos y estilo pedagógico de la misma.

El usuario se convierte en el protagonista de su propio aprendizaje a través del autoservicio y los servicios guiados por los tutores o profesores mediante la herramienta. Además, permite utilizar los cursos desarrollados por terceros, personalizando el contenido y reutilizando el conocimiento adquirido. Además, posee prestaciones y características que hacen que cada plataforma sea adecuada según los requerimientos y necesidades de los usuarios.

Los LMS que almacenan información de programas académicos no involucran el aspecto de resultados pedagógicos en sus estructuras de datos o procesos de aprobación. Esta aprobación se realiza a través de flujos de trabajo[11].

2.3 Sistemas de gestión de evaluación basadas en competencias

Un AMS es un sistema, generalmente basado en tecnologías web, que permite a la institución la recolección, el manejo, y reporte de datos relacionados a las evaluaciones, por lo general basadas en competencias, del estudiante. Los AMS basadas en competencias permiten a la institución y a los educadores listar sus competencias, guardar, y mantener datos para cada competencia, facilitar conexiones a competencias similares de la institución, y generar reportes[12].

Los AMS permiten que las competencias sean enlazadas a nivel institucional, departamental, de programas, y de división. Esta estructura permite examinar la competencia acorde al nivel que pertenece. Una representación común de este tipo de enlace es el mapa curricular[13]. Algunos de estos sistemas de manejo de evaluaciones generan sus propios mapas curriculares.

Varios sistemas de manejo de evaluaciones comerciales existen; incluyendo Blackboard Learn, Campus Lab, eLumen, LiveText, TaskStream, TracDat/Webfolio, Waypoint Outcomes y WEAVEOnline. Además de estas aplicaciones comerciales para manejo de evaluaciones también existen otras hechas por las propias instituciones para manejar sus datos de evaluaciones.

Mientras que cada AMS tiene un conjunto diferente de capacidades, todas manejan, mantienen, y permiten generar reportes de los datos de las evaluaciones. Generalmente, teniendo como ejemplo los distintos sistemas, los AMS tienen una estructura jerárquica basada en unidades organizacionales (programas, departamentos, escuelas, colegios o la misma institución), por lo tanto, las metas y/o las competencias también se ven adaptadas a esta estructura.

2.3.1 Capacidad de evaluación

La característica más importante de todo AMS es la capacidad de soportar evaluaciones con sus diferentes tipos de evaluación. Por ejemplo, algunos AMS se centran soportar evaluaciones acumulativas; mientras que otros permiten además el seguimiento de las evaluaciones formativas. Además, las capacidades de evaluación apoyadas por una AMS pueden residir en una escala de la unidad.

Algunos AMS permiten evaluaciones a nivel de estudiante. Un número cada vez mayor de las AMS apoyan la documentación, desarrollo o aplicación de criterios de evaluación específicos, más comúnmente a las rúbricas que se aplican a los productos creados por los estudiantes.

Como una faceta adicional, muchos AMS permiten evaluaciones para vincularse con las normas educativas y profesionales, de modo que la información de evaluación de múltiples unidades puede ser adherido como datos de reportes.

2.3.2 Alineación de capacidades

Una importante característica de cualquier AMS es la habilidad de enlazar competencias con cualquier nivel de la institución u organización.

Primeramente, algunos AMS permiten que las competencias sean enlazadas a nivel institucional, departamental, de programas, de división. Esta estructura permite examinar la competencia acorde al nivel que pertenece. Una representación común de este tipo de enlace es el mapa curricular. Algunos de estos sistemas de manejo de evaluaciones generan sus propios mapas curriculares.

Un sistema de manejo de evaluaciones o AMS es un sistema web que provee soporte a iniciativas de mejora continua educacionales de instituciones y competencias de aprendizaje que necesitan las evaluaciones. [bibliografía pendiente] Un AMS puede ser levantado y utilizado para mantener y alcanzar altos estándares de calidad y cumpliendo los requisitos de acreditación. [bibliografía pendiente] Esta evaluación es un proceso complejo que requiere la contribución y la retroalimentación de todo el personal, profesores y alumnado de un centro de educación superior. El sistema AMS facilita la esquematización, la recopilación de pruebas, documentación y presentación de las contribuciones que cada uno de los programas académicos de la institución y los servicios de apoyo hace la consecución de los objetivos de calidad y eficacia institucionales.

Un ciclo de evaluación completo incluye la coordinación, la planificación, la medición, la reflexión y la toma de acción del proceso de evaluación de la institución enteras, programas o cursos.

2.4 Diferencias entre LMS y AMS

Por lo general, en las universidades norteamericanas utilizan los AMS y LMS para mejorar el proceso de evaluación de sus estudiantes. Sin embargo, para las personas que no están familiarizadas con estos sistemas pueden quedar ambiguas las diferencias entre estos sistemas de aprendizaje. Estas diferencias se encuentran sumarizadas y tabuladas en la tabla 2.1.

TABLA 2.1: Comparación de características entre plataformas AMS y LMS.

Características	AMS	LMS
Soporte de evaluaciones a los estudiantes.	✓	✓
Soporte de evaluación colectiva.	✓	
Diferentes tipos de rúbricas para evaluaciones.	✓	
Permite acceder a cursos realizados por terceros (aula virtual para el estudiante).		✓
Permite la evaluación de desempeño estudiantil.	✓	
Permite generar reportes de cursos, progresos, etc.	✓	✓
Soporte de presupuestos (solicitud de profesores, coordinadores, etc.).	✓	
Soporta competencias de aprendizaje del estudiante o Student Learning Outcomes.	✓	
Soporte de alineación de competencias.	✓	
E-portfolio y evaluación de proyectos.	✓	✓
Soporte de comunicación entre estudiantes y profesores (foro).		✓
Soporte de evaluación sumativa y formativa.	✓	
Software distribuido y desarrollado libremente.		✓

Los LMS permiten al estudiante a una capacitación flexible a distancia sin limitaciones de horarios con costos reducidos. Además, como es una plataforma intuitiva, permite a las personas con nivel de conocimiento básico en informática un aprendizaje constante y actualizado a través de la interacción entre alumnos y profesores. En cambio, los AMS son más bien utilizados para evaluaciones de las competencias de los alumnos que como vínculo entre el alumno y el profesor mediante un aula virtual, y permiten la mejora continua del aprendizaje estudiantil.

Los LMS permiten la integración de competencias mediante herramientas externas, pero de una manera superficial en comparación a los AMS.

2.5 Programas de estudio

En términos generales, se puede definir un programa de estudios como una herramienta educativa que regula y ordena el proceso de enseñanza-aprendizaje a desarrollar en una unidad de aprendizaje determinada, orientando las actividades que profesor y alumno han de llevar a cabo para el logro de los objetivos planteados en dicha unidad, en relación con los objetivos del plan de estudios, de tal manera que el egresado concluya su carrera con el perfil deseado. En pocas palabras, es un esquema organizado de los contenidos situados dentro de una determinada unidad de aprendizaje[14].

El termino “unidad de aprendizaje” sustituye al de “asignatura” o “materia” que evocan los tradicionales cursos unidisciplinarios, generalmente teóricos y sobrecargados de información. Un programa resume las características de la unidad de aprendizaje, su contenido mínimo obligatorio, y sus objetivos, principalmente.

En la primera etapa de diseño curricular, se requiere la elaboración de la propuesta de los programas para su aprobación de parte de las autoridades con la colaboración de las academias y, de ser necesario, con la asesoría de externos de la unidad académica.

2.5.1 Proceso curricular

2.6 Sistemas de gestión curricular

Un CMS [2] es un aplicación automatizada que apoya todo el proceso curricular, desde la planificación hasta la implementación y evaluación. Posee una interfaz única y cohesiva en línea que permite proponer, crear, evaluar, revisar, aprobar y aplicar cursos, programas y competencias.

Curriculum es una mezcla sofisticada de estrategias educativas, contenido del curso, resultados de aprendizaje, experiencias educativas y evaluación[15]. Esta visión amplia de un CMS se deriva del ambiente actual de educación elemental y secundaria que es impulsado por los estándares de contenido de cursos obligatorios federales y estatales, y la necesidad de auditorías continuas de currículo[16].

En los enfoques actuales del desarrollo de Curriculum por lo general gira en torno a los comités curriculares. Un comité curricular departamental comienza el proceso de desarrollo curricular considerando como entrada cualquiera de los aspectos mostrados en la figura 2.1.

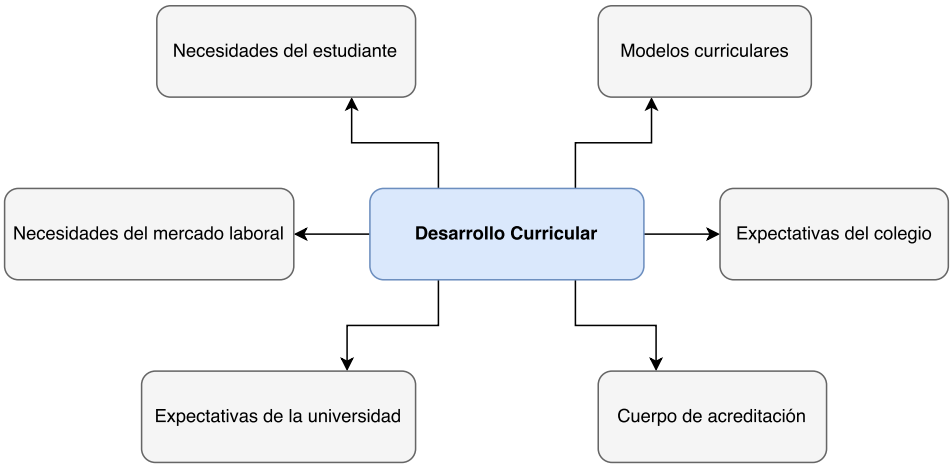

FIGURA 2.1: Esquema de diseño curricular y sus variables determinantes.

Dichos aspectos, sin embargo, sólo se consideran en un alto nivel de abstracción basado en la comprensión tácita de los miembros del comité sobre la disciplina.

2.7 Aplicaciones web

La necesidad de ejecución de operaciones complejas de manera remota y portable, tales como el uso de software sin depender de la potencia del hardware del usuario, la disponibilidad de uso en múltiples plataformas, y muchas otras, han guiado al desarrollo y evolución de las aplicaciones web. En las ciencias de la computación, una aplicación web es un software con arquitectura cliente-servidor donde el cliente (o interfaz de usuario) corre en un navegador web[1].

[2] de sus siglas en inglés, Curriculum Management System, que significa sistema de gestión curricular

Los usuarios acceden a la aplicación utilizando un navegador web y no es necesario descargarse ningún tipo de software adicional, ya que la aplicación se ejecuta en el servidor. Para esclarecer el panorama, la lógica de la aplicación web se ejecuta remotamente, y el navegador web sólo se limita a la representación de los datos.

La evolución de las aplicaciones web ha crecido tan vertiginosamente que además de ofrecer una multitud de servicios, mejoran la UX a través de interfaces gráficas impactantes e intuitivas para los usuarios[17].

2.8 Cloud computing

Casi toda solución de infraestructura remota es llamada hoy día con el termino de computación en la nube o simplemente la nube.

La computación en la nube es una metáfora para abastecimiento y consumición de recursos de infraestructura. El nivel de abstracción ofrecida por la nube puede variar de hardware virtual a complejos sistemas distribuidos, debido a que los recursos están disponibles a demanda en enormes cantidades y pagados por uso[18].

La infraestructura remota o nube puede ser manejada por una organización abierta para uso público, o puede ser privada, donde una nube que virtualiza y comparte la infraestructura con una sola organización o híbrida como son mostrados en la figura 2.2.

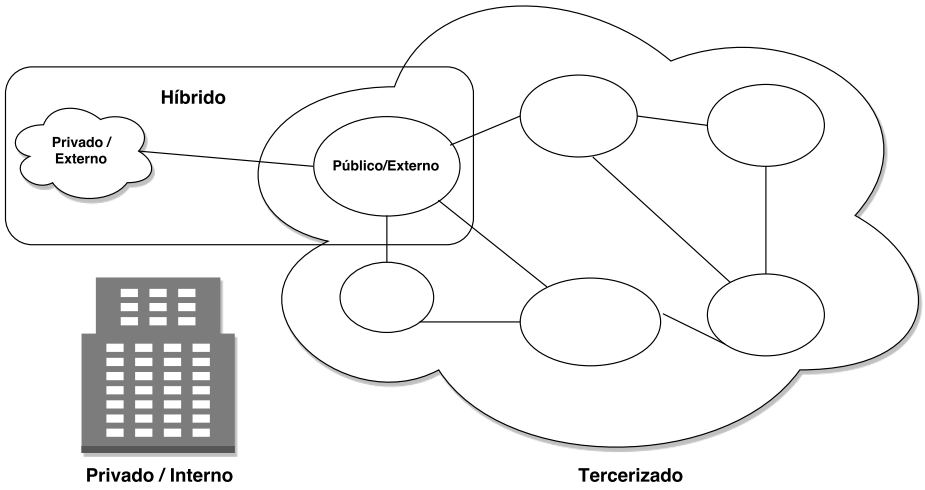

FIGURA 2.2: Esquema de tipos de computación en la nube.

2.9 Software as a service

SaaS es un paradigma de entrega de software donde la misma se encuentra alojada por lo general en la nube y se entrega como servicio a través de Internet a un gran número de usuarios a través de un modelo de suscripción. Se trata de un modelo de entrega de negocio en el que tanto la aplicación y el alojamiento son gestionados y compartidos con varias empresas, que alquilan y utilizan los servicios de aplicaciones de forma centralizada[19].

El software como servicio es equivalente a servicios de proveedores externos que manejan todo mantenimiento, personalización, actualización y cobro para los servicios que su cliente utiliza de manera mensual o anual. El proveedor se encarga de ofrecer el software basado en un conjunto de códigos y datos definidos junto a las diferentes configuraciones para los diferentes clientes. Los suscriptores al servicio acceden a la aplicación con la sensación de que son los únicos usuarios de la aplicación. Sin embargo, los cambios de configuraciones como cambios de datos, flujo de trabajo, interfaz y el flujo de negocio son realizados de manera masiva y transparente para ellos[20][21].

2.9.1 Características

Entre las características relevantes de las aplicaciones SaaS, y las que las remarcan como aplicaciones bien construidas, son construidas a la medida, escalables y soportan multitenancy. No todas las aplicaciones SaaS comparten todas las características, pero las más comunes son las siguientes: [1][3][4][5]

2.9.1.1 Configurabilidad

Las aplicaciones con esta característica poseen el mismo código base y provee a los tenants con múltiples opciones de configuración tal que cada tenant pueda tener sus propias configuraciones de software únicas y pueda tener la sensación de que es el único usuario en utilizar la aplicación. Esta es la clave del éxito para las aplicaciones SaaS y se puede encontrar esta característica en aplicaciones de Nivel 2 para adelante. Por ejemplo, cada tenant puede tener configurado su sitio para que muestren fondos de pantallas o logos en las páginas de inicio de sesión o páginas principales que ellos especifican. Esta característica también puede ser llamada personalización de la aplicación.

2.9.1.2 Multitenancy

Con esta característica, una sola instancia de la aplicación corriendo puede servir a todos los clientes. Los diferentes modelos de datos que están disponibles para soportar SaaS son las bases de datos aisladas, arquitectura de bases de datos aisladas compartidas y la construcción de datos. Utilizando la arquitectura multitenant los proveedores de aplicaciones SaaS pueden innovar de forma sencilla y ahorrar tiempo valioso gastado en mantener varias versiones de código deprecado y/o desactualizado. Esta característica se incluye en todas las aplicaciones de nivel 3 o nivel 4.

2.9.1.3 Escalabilidad

Esta característica es la más complicada de adherir a una aplicación SaaS debido a su elevado costo. La escalabilidad es soportada por la virtualización, pero teniendo en cuenta el costo y el problema de complejidad muchas veces el desarrollador de la aplicación no se complica con esta característica.

2.9.2 Ventajas

Además, aparte de estas características, algunas ventajas que poseen las aplicaciones con arquitectura SaaS son las siguientes: [4][6][11]

- Las aplicaciones SaaS pueden ser utilizadas por los usuarios mediante thin-client por medio de sus navegadores Web. Esto ahorra en costos operacionales para el usuario junto con los requerimientos de hardware mínimo, por lo tanto, reduce el costo que el usuario necesita gastar en hardware. Además de los costos de mantenimiento, costos de licencias del software también son minimizados.

- Mejor utilización de recursos, debido a que los recursos requeridos por las aplicaciones con arquitectura SaaS son mínimos.

- El avance de la tecnología Web permite que los proveedores de SaaS se ubiquen en el extranjero y también ofrezcan servicios de alta calidad. De esta manera, permite a los usuarios de la aplicación ahorrar en infraestructura.

- Usualmente, las soluciones SaaS residen en entornos en la nube donde son escalables y poseen integración con otras ventajas SaaS. En comparación al modelo tradicional, los usuarios no tienen que comprar otro servidor o software.

2.10 Metodología ágil de desarrollo de software

La metodología Ágil envuelve un enfoque para la toma de decisiones en los proyectos de software, que se refiere a métodos de ingeniería del software basados en el desarrollo iterativo e incremental, donde los requisitos y soluciones evolucionan con el tiempo según la necesidad del proyecto. Los métodos tradicionales, como Waterfall, pretenden ser capaces de modelar completamente el dominio del problema de entrada y luego esperar que se produzcan pequeños cambios (o inclusive ninguno)[22]. Los métodos ágiles asumen que el cambio es inevitable, por lo que abordan el desarrollo de software de tal manera a facilitar la adaptación de los nuevos requisitos mientras vayan surgiendo.

Las metodologías ágiles, en comparación a otras metodologías de desarrollo, ofrecen un modelo de diseño flexible que fomenta al desarrollo evolutivo. Los desarrolladores trabajan en pequeños módulos cada vez y la retroalimentación proveída por el cliente ocurre simultáneamente en el desarrollo. Además, la metodología puede ser bastante útil en situaciones donde los objetivos finales del proyecto no están claramente definidos donde los requisitos del cliente se clarificarán gradualmente a medida que el proyecto avance.

El uso de la metodología ágil como método de entregas de las características del módulo entra como requisito no funcional.

2.10.1 Historias de usuario

Las historias de usuario conforman la parte central de muchas metodologías de desarrollo ágil, tales como XP[3], Scrum, entre otras. Estas definen lo que se debe construir en el proyecto de software, tienen una prioridad asociada definida por el cliente de manera a indicar cuales son las más importantes para el resultado final. Son divididas en tareas y su tiempo es estimado por los desarrolladores.

Por lo general, se espera que una estimación de tiempo de cada historia de usuario se sitúe entre horas y el tiempo máximo de iteración. Estimaciones superiores a este tiempo máximo son indicativas de que la historia es muy compleja y debe ser dividida en varias historias.

Una historia de usuario es una representación de un requisito escrito en una o dos frases utilizando el lenguaje común del usuario[22]. Ellas son utilizadas para la especificación de requisitos acompañadas de las discusiones con aquellos y las pruebas de validación.

Cada historia de usuario debe ser limitada. La metodología estipula que las mismas deben ser escritas por los clientes. Son una forma rápida de administrar los requisitos sin tener que elaborar gran cantidad de documentos formales y sin requerir de mucho tiempo para administrarlos.

Las historias de usuario deben ser:

- Independientes unas de otras. De ser necesario, combinar las historias dependientes o buscar otra forma de dividir las historias de manera que resulten independientes.
- Negociables. La historia en sí misma no es lo suficientemente explicita para considerarse un contrato, la discusión con los usuarios debe permitir esclarecerse y éste debe dejarse explicito bajo la forma de pruebas de validación.
- Valoradas por los clientes o usuarios. Los intereses de los clientes y de los usuarios no siempre coinciden, pero en todo caso, cada historia debe ser más importante para los clientes que para el desarrollador.
- Pequeñas. Las historias grandes son difíciles de estimar e imponen restricciones sobre la planificación de un desarrollo iterativo. Generalmente se recomienda la consolidación de historias muy cortas en una sola historia.
- Verificables. Las historias de usuario cubren requerimientos funcionales, por lo generalmente son verificables. Cuando sea posible, la verificación debe automatizarse, de manera que pueda ser verificada en cada entrega del proyecto.

Las iniciales de estas características, con sus nombres en inglés, forman la palabra INVEST, que significa "inversión". Es porque toda historia de usuario, si se construye bien, es una inversión.

Al momento de implementar las historias, los desarrolladores deben tener la posibilidad de discutirlas con los clientes. El estilo sucinto de las historias podría dificultar su interpretación, podría requerir conocimientos de base sobre el modelo, o podría haber cambiado desde que fue escrita.

[3] de sus siglas en inglés, eXtreme Programming, que significa en español programación extrema

Cada historia de usuario debe tener en algún momento pruebas de validación asociadas, lo que permitirá al desarrollador, y más tarde al cliente, verificar si la historia ha sido completada. Como no se dispone de una formulación de requisitos precisa, la ausencia de pruebas de validación concertadas abre la posibilidad de discusiones largas y no constructivas al momento de la entrega del producto.

2.10.2 Épicas

Una épica es esencialmente una historia de usuario de un tamaño mucho mayor, siempre superior al tiempo de iteración máximo, y tiene como propósito el de asociar historias de usuario individuales relacionadas con un propósito de más alto nivel que cumplir. La misma es, por lo general, muy grande para que un equipo del proyecto pueda trabajar directamente sin partir en diversas historias de usuario[23].

El uso de las épicas en proyectos de gran tamaño ayuda a organizar tareas complejas en un tipo de estructura para que la interrelación de historias de usuario esté bien entendida. Por lo tanto, el diseño de épicas en el proceso de desarrollo es fundamental antes de comenzar el proyecto.

2.11 Interacción humano-computador

En HCI definen la funcionalidad y la usabilidad de los sistemas que se desarrollan, donde la funcionalidad de un sistema es definida por un conjunto de acciones o servicios que son proveídas a los usuarios, sin embargo, el valor de la funcionalidad es verificada cuando es eficientemente utilizada por el usuario[24]. La usabilidad de un sistema con cierta funcionalidad es el rango y grado por el cual el mismo puede ser utilizada de manera eficiente y adecuada para cumplir ciertas metas para ciertos usuarios. La eficiencia de un sistema es alcanzada cuando se cumple un balance entre la usabilidad y la funcionalidad[25].

HCI es un diseño que debe producir un ajuste entre el usuario, la máquina y los servicios requeridos con el fin de lograr un balance óptimo entre la calidad y la eficiencia de los servicios.

La definición de la estrategia de UI[4] es importante para una mejor usabilidad del sistema, donde este proceso debería comenzar antes que el diseño y desarrollo de las aplicaciones. Es la visión de una solución que necesite ser verificado con potenciales usuarios que prueben que necesite el mercado[26].

Por lo tanto, al inicio del diseño de la aplicación se llevará a cabo una serie de diseños de funcionalidad y usabilidad que llevar a la mejor experiencia de uso del módulo de gestión curricular, donde dichos diseños serán validados por el equipo en los Estados Unidos antes de iniciar el desarrollo.

[4] de sus siglas en inglés, User Interface, que significa en español experiencia de usuario.

Capítulo 3

Estado del arte

3.1 CurricUNET

CurricUNET es una aplicación web diseñada para automatizar la emisión y aprobación del plan de estudios emitido por profesores y/o encargados de universidades norteamericanas; que incluyen programas, cursos y competencias.

En la misma se desarrollan propuestas de cursos y programas de estudio mediante formularios de la aplicación, con el objetivo de reemplazar solicitudes en papel que universidades utilizaban para emitir propuestas. Además, ofrece almacenamiento e información de plan de estudios históricos, activos y propuestos.

Todas las entradas, revisiones y reportes son accedidas por la web desde los navegadores. Posee un sistema de notificaciones integrado que permite al usuario un mejor seguimiento del progreso de las propuestas y cursos en revisión. Además, dispone de un control de versionamiento de cursos, planes de estudio y competencias.

Los usuarios del sistema pueden acceder a los reportes e historial de versiones de sus cursos y planes de estudio por lo que ayuda a una mejora continua del programa universitario.

3.2 Courseleaf

El módulo de Curriculum de CourseLeaf es una solución de gestión basada en la web, mejorando los procesos de profesores y del comité de Curriculum de al menos 70 instituciones.

Cuando el módulo de Curriculum de CourseLeaf se combina con su módulo de catálogo, colegios y universidades son capaces de gestionar y realizar un seguimiento de la información del programa, desde la propuesta hasta publicar en una aplicación integrada con facilidad.

CourseLeaf Curriculum es una solución de gran alcance con una completa funcionalidad. El software proporciona la generación de flujo de trabajo automático, notificaciones automáticas. Además, identifica todos los cursos, programas y departamentos que se ven afectados por los cambios propuestos en el inicio del proceso de propuesta y puede ayudar en la actualización con los cambios completados. El módulo de Curriculum de CourseLeaf se puede implementar con o sin su catálogo de Cursos.

El módulo de Curriculum dispone de cuatro componentes:

- Un listado de cursos donde los usuarios pueden encontrar información sobre el curso. También pueden iniciar, editar y presentar propuestas de cursos.

- Una interfaz de los usuarios autorizados a opinar, anotar, rechazar, y aprobar las propuestas.
- Un formulario de curso, un formulario dinámico con campos inteligentes y el formato de respuesta que optimiza la experiencia del usuario.
- Un formulario de los programas, utilizando la misma funcionalidad que los formularios de curso para proponer y mantener la información y los requerimientos para los programas y grados.

3.3 DECA: Curriculum Navigator

DECA ofrece, mediante su módulo de Curriculum Navigator, una solución de desarrollo y administración de Curriculum.

Frecuentemente, descrito como complejo e ineficaz, los métodos tradicionales basados en papel de gestión de programas de estudio proporcionan una visibilidad limitada, lo que resulta en una visión restringida de las etapas implicadas en la creación, modificación y aprobación de planes de estudio. Además, busca eliminar esta complejidad al acelerar el desarrollo curricular y el proceso de aprobación.

Cuando se utiliza como módulo integrado de su Catálogo, denominado como Catalog Navigator, proporciona una solución que les permite iniciar el camino de aprobación de alguna carrera de grado. Asesores y administradores, por otra parte, utilizan tanto Curriculum Navigator y Catalog Navigator para desarrollar y proporcionar datos públicamente del plan de estudios para sus estudiantes actuales y futuros.

Curriculum Navigator ofrece:

- El acceso a un repositorio de datos curricular.
- Editar, guardar y proponer planes de estudio o carreras de grado.
- Historial de cambios y de accesos de revisión.
- Seguimiento en tiempo real de propuestas abiertas.

3.4 Comparación entre plataformas

Una investigación para comprobar otros proyectos o productos con las mismas características propuestas, buscando innovación para el mercado es expuesta en la tabla 3.1.

TABLA 3.1: Relación entre sistemas de gestión curricular.

Características	CurricUNET	CourseLeaf	DECA
Creación y versionamiento de competencias.			
Creación y versionamiento de cursos.	✓	✓	✓
Creación y versionamiento de programas de estudio.	✓	✓	
Cumple los Estándares de códigos de California.	✓		
Historial de versiones de competencias.			
Historial de versiones de cursos.	✓	✓	✓
Historial de versiones de programas de estudio.	✓		
Reporte de Comparación entre versiones de cursos.	✓		✓
Soporta competencias de aprendizaje del estudiante.			
Plantilla de flujo de trabajo customizable.	✓		
Permite asignar roles evaluadores en la aplicación.	✓	✓	
Permite asignar usuarios como colaboradores.	✓		
Sistema de alertas para colaboradores y evaluadores.	✓	✓	
Buzón de entrada para colaboradores y autoridades.	✓		✓
Soporte de correlatividades entre cursos.	✓		
Incluye un catálogo de cursos.	✓	✓	✓
Incluye un catálogo de programas de estudio.	✓		
Incluye un catálogo de competencias.			
UX intuitiva y efectiva.		✓	✓

Capítulo 4

Caso de estudio y observación participante

Capítulo 5

Propuesta de solución

5.1 Modelo de arquitectura de módulo

5.2 Requerimientos no funcionales

Realizamos una revisión de las tecnologías y abordajes brindadas como requerimientos funcionales. Sin embargo, al utilizar un abordaje ágil, la validación de las decisiones tecnológicas depende en ultima instancia de la validación del proceso de desarrollo realizada por expertos y usuarios.

Las decisiones de tecnologías para el módulo de gestión curricular fueron basadas a los conocimientos adquiridos de los miembros del equipo de desarrollo, con el fin de optimizar tiempos utilizados en curvas de aprendizaje. Sin embargo, se hicieron previos análisis a su uso para corroborar que dichas tecnologías cumplen con el propósito de desarrollo.

Las siguientes secciones son las tecnologías utilizadas y un previo análisis previo a su uso.

5.2.1 Java

La elección del lenguaje de programación, establecida por la organización, fue establecida como lenguaje para la lógica del módulo curricular ya que facilita su mantenimiento por parte del equipo de desarrolladores. El equipo de desarrollo posee conocimiento en este lenguaje de programación o en lenguajes de programación orientado a objetos por lo que se aprovechó el tiempo utilizado para curvas de aprendizaje del lenguaje en investigar buenas prácticas para el proyecto.

El lenguaje de programación Java fue diseñado para alcanzar los desafíos del desarrollo de aplicaciones en el contexto de ambientes heterogéneos y distribuidos en red. Lo más importante de solucionar entre estos desafíos era la entrega segura de aplicaciones que consumen lo mínimo de recursos del sistema, correr en cualquier plataforma de hardware y/o software, y que pueda ser extendido de manera dinámica.

Operar en múltiples plataformas con redes heterogéneas invalida la arquitectura tradicional de distribución de binarios, “release”, actualización, parcheo, etc. Para lograr sobresalir, JAVA se convirtió en una arquitectura neutral, portable y adaptable de manera dinámica.

Hoy día, Java es uno de los lenguajes de programación más utilizados a nivel mundial debido a su portabilidad y evolución con el paso del tiempo. Además, al ser un lenguaje de programación multiplataforma cualquier proyecto se puede desarrollar en cualquier sistema operativo o plataforma para luego ser levantada en el servidor independiente a su plataforma.

El lenguaje es reconocido por ser intuitivo a la hora de programar, altamente portátil y portable. Además, la comunidad es uno de los fuertes de Java, ya que hay cursos en línea para todos los niveles de conocimiento. Además, en la Web hay una gran cantidad de foros y librerías de la comunidad para resolver diferentes problemáticas de los desarrolladores. En el gráfico 5.1 se puede observar cuáles son los lenguajes de programación más usados en la comunidad de desarrolladores en los últimos años.

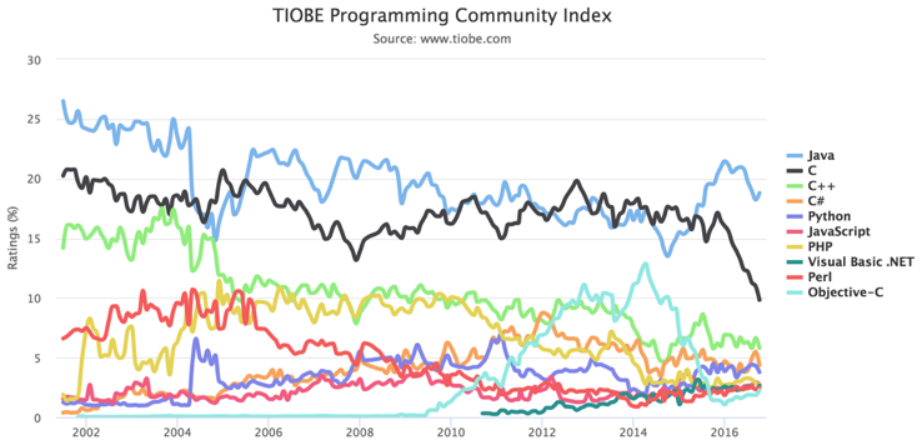

FIGURA 5.1: Gráfico de uso de lenguajes de programación con respecto al tiempo.

Como análisis de riesgos se hizo una breve búsqueda de vulnerabilidades, pero como no hubo riesgos importantes a tener en cuenta se decidió seguir utilizando el lenguaje de programación Java como estuvo previsto. El lenguaje Java proporciona un nivel de rendimiento adecuado para la plataforma donde se aloja la aplicación base.

Dicho uso del lenguaje fue validado durante el proceso de desarrollo al poder resolver los criterios de aceptación de las historias de usuario.

5.2.2 MySQL

Para ayudar a que el AMS basado en competencias sea adaptable y fácil de mantener, se extenderá la base de datos ya utilizada por la aplicación base. Por lo tanto, la elección de la base de datos del módulo queda a criterio de la organización y como requisito no funcional para el proyecto final. El sistema de base de datos que se utiliza para el sistema de evaluación de competencias es MySQL, por ser un sistema open source y con una de las comunidades más grandes entre las bases de datos existentes.

MySQL es el sistema de base de datos open source más popular disponible. Es particularmente eficaz para sitios web públicos que requieren base de datos rápidas y estables. [Learning MySQL and MariaDB – O'Neill] Como sistema de administración de Base de Datos, es una colección estructurada de datos. Puede ser usada para cualquier tipo de solución, desde una lista de compras hasta una enorme cantidad de información de una red empresarial. Para añadir, acceder y procesar datos almacenados en una base de datos informática, se necesita de un sistema de administración de Base de Datos como MySQL Server. Como las computadoras son muy buenas manejando grandes cantidades de datos, los sistemas de administración de

base de datos juegan un rol principal en la computación, como utilidades autónomas, o como partes de otras aplicaciones.

Para representar los datos que almacena utiliza el modelo lógico de datos relacional, donde almacena sus datos en tablas separadas antes que almacenar todos los datos en un solo lugar. La estructura de la base de datos está organizada en archivos físicos optimizados para mayor velocidad. El modelo lógico, con objetos tales como bases de datos, tablas, vistas, filas y columnas, ofrecen un ambiente de programación flexible donde se establecen reglas que gobiernan las relaciones entre las de los distintos tipos de campos, tales como uno a uno, uno a muchos, únicos, requeridos u opcionales y punteros entre diferentes tablas.

Hemos usado MySQL sin darnos cuenta; en Google, Amazon, Facebook, Wikipedia y otros sitios web populares. Es el guardián de los datos detrás de grandes sitios con cientos de páginas de datos, sin dejar atrás los pequeños sitios con solo algunas páginas. También es utilizado en aplicaciones que no son basadas en web. Es rápida, estable y pequeña cuando se requiere.

En los siguientes gráficos podemos observar que MySQL es un sistema de manejo de base de datos muy utilizado por la comunidad y va en aumento de popularidad hasta casi alcanzar a uno de los gigantes que es Oracle.

Rank			DBMS	Database Model	Score		
Nov 2016	Oct 2016	Nov 2015			Nov 2016	Oct 2016	Nov 2015
1.	1.	1.	Oracle	Relational DBMS	1413.01	-4.09	-67.94
2.	2.	2.	MySQL	Relational DBMS	1373.56	+10.91	+86.71
3.	3.	3.	Microsoft SQL Server	Relational DBMS	1213.80	-0.38	+91.48
4.	↑ 5.	↑ 5.	PostgreSQL	Relational DBMS	325.82	+7.12	+40.13
5.	↓ 4.	↓ 4.	MongoDB	Document store	325.48	+6.67	+20.87
6.	6.	6.	DB2	Relational DBMS	181.46	+0.90	-21.07
7.	7.	↑ 8.	Cassandra	Wide column store	133.97	-1.09	+1.05
8.	8.	↓ 7.	Microsoft Access	Relational DBMS	125.97	+1.30	-14.99
9.	9.	↑ 10.	Redis	Key-value store	115.54	+6.00	+13.13
10.	10.	↓ 9.	SQLite	Relational DBMS	112.00	+3.43	+8.55

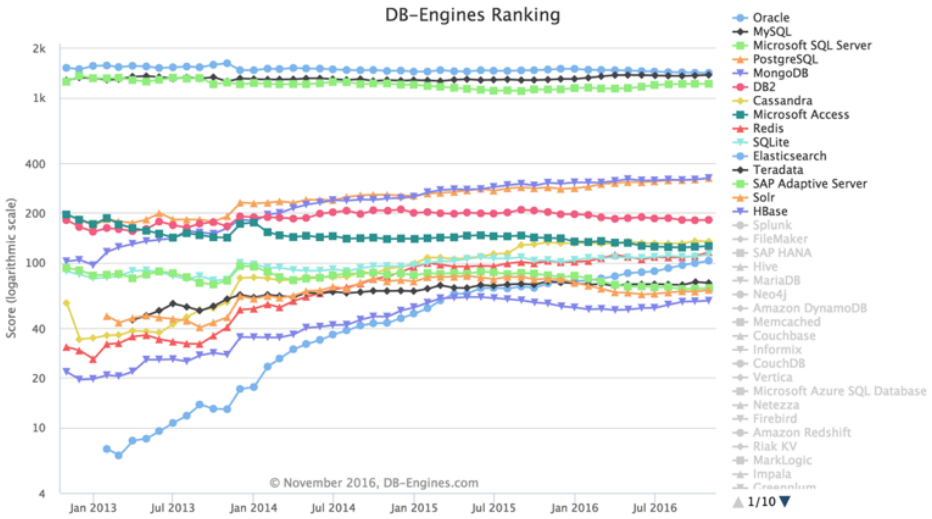

FIGURA 5.2: Gráfico de uso de motores de base de datos con respecto al tiempo.

Como el motor de base de datos es otro requisito no funcional propuesto por la organización, como es una tecnología utilizada para la aplicación base. Sin embargo, se hizo un breve análisis de vulnerabilidades para verificar que el uso de base de datos relacionales podría ser efectivo para el proyecto final pero como no se encontraron vulnerabilidades críticas se continuó la utilización de la base de datos MySQL, además de que los desarrolladores están familiarizados con la misma.

5.2.3 NoSQL

5.2.4 Amazon Web Services

Amazon Web Services (AWS) es una plataforma de servicios web que ofrece soluciones de procesado, almacenaje y redes en diferentes capas de abstracción. Se pueden usar estos servicios para hospedar sitios web, correr aplicaciones complejas y para minería de grandes cantidades de datos. [X] El término servicios web se refiere a servicios que pueden ser controlados o accedidos por medio de una interfaz web. La interfaz web puede ser manejada por máquinas o por usuarios por medio de una interfaz de usuario gráfica. Los servicios más prominentes que ofrece son EC2, que ofrece servidores virtuales, y S3, que ofrece capacidad de almacenamiento.

AWS es una nube pública, donde tiene las siguientes clasificaciones como nube:

- Infraestructura como Servicio (IaaS) – Ofrece recursos fundamentales como procesamiento, almacenamiento, y capacidades de servicios, utilizando servidores virtuales tales como Amazon EC2, Google Compute Engine y Microsoft Azure en las máquinas virtuales.
- Plataforma como Servicio (PaaS) – Provee plataformas para desplegar aplicaciones customizadas a la nube, tales como AWS Elastic Beanstalk, Google App Engine, y Heroku.
- Software como Servicio (SaaS) – Combina infraestructura y software corriendo en la nube, incluyendo aplicaciones de oficina como Amazon WorkSpaces, Google Apps for Work, y Microsoft Office 365.

En el siguiente gráfico se puede apreciar el porcentaje de usuarios de las plataformas de servicios web. Amazon Web Services se encuentra en el top debido a su buen servicio y ser uno de los primeros de ofrecer servicios de infraestructura como servicio.

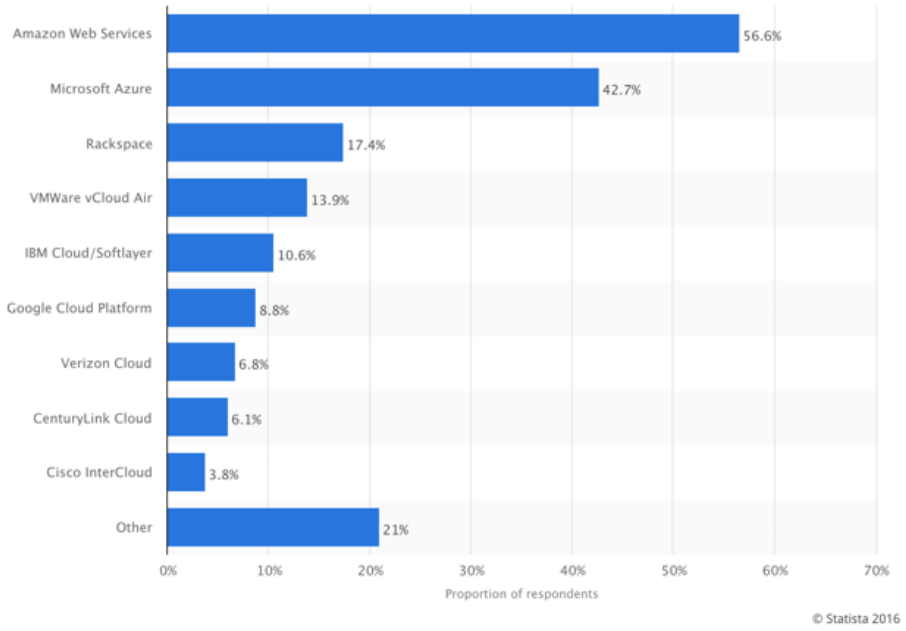

FIGURA 5.3: Ranking de compañias que brindan servicios de cloud computing.

La aplicación base del módulo de gestión de programas de estudio se encuentra alojado en la plataforma de servicios de Amazon o AWS. Por lo tanto, se utilizaron los servidores ya en línea para correr el módulo de la aplicación.

5.2.5 Git

En esta sección se tratará de cómo se utilizará Git para el desarrollo del proyecto final, donde la organización dueña de la aplicación Web de evaluación académica basada en competencias brindó permisos en su repositorio privado para subir los cambios y para controlar las versiones de dicho módulo. Además, el uso de un sistema de versionamiento permite minimizar y optimizar el tiempo de unión de módulos que van agregando o actualizando los miembros del equipo de desarrollo. Por lo tanto, un trabajo en equipo requiere de un sistema de control de versiones eficiente. Partiremos explicando los conceptos relativos a las herramientas de control de versiones.

El control de versiones es un sistema que registra los cambios realizados sobre un archivo o conjunto de archivos a lo largo del tiempo, de esta forma permite recuperar versiones específicas más adelante. [X]

Git modela sus datos más como un conjunto de instantáneas de un mini sistema de archivos. Cada vez que se confirma el cambio de un archivo, o se guarda el estado de un proyecto en Git, Git básicamente hace una foto del aspecto de todos los archivos en ese momento y guarda una referencia a esa instantánea. Para ser eficiente, si los archivos no se han modificado, Git no almacena el archivo de nuevo, solo un enlace al archivo idéntico anterior que ya tiene almacenado. Tiene un comportamiento como en el siguiente gráfico.

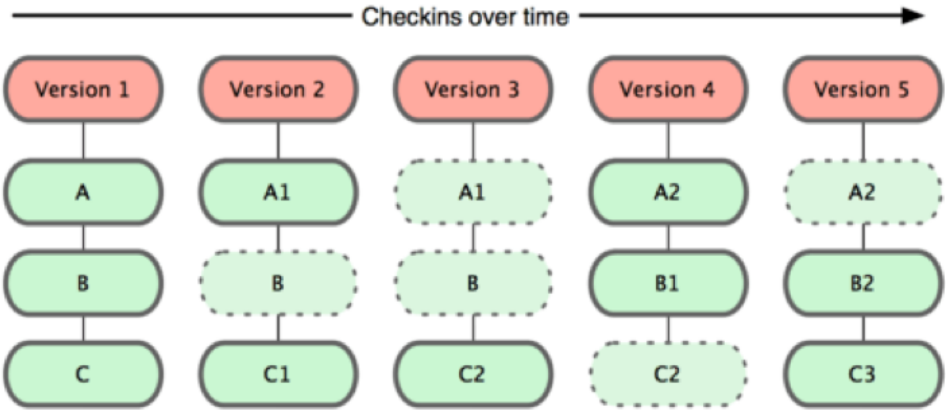

FIGURA 5.4: Flujo de versiones de Git.

Uno de los fuertes importantes de Git como VCS es su integridad, debido a que toda versión es verificada mediante una suma de comprobación (checksum en inglés) antes de ser almacenado, y es identificado a partir de ese momento dicha suma. Esta suma es utilizada para volver a un estado anterior del repositorio en caso de ser necesario o ver cuáles fueron los cambios que se realizaron en ese pedazo de código trabajado o más conocido como commit.

5.2.6 Spring

El sistema de evaluación académica cuenta con un framework deprecado y uno nuevo en el cuál se estuvo trabajando para los nuevos módulos que se fueron desarrollando, como requisito no funcional que establece la empresa entra el uso de Spring MVC como framework para la parte lógica de la aplicación, por lo que requiere un estudio de la documentación que ofrece Spring para empezar a usar el framework y las buenas practicas establecidas por el equipo de desarrollo. En esta sección estaremos analizando Spring como framework a ser utilizado en el proyecto final, algunas de sus características resaltantes que ayudarían a entender la estructura interna y funcionamiento como framework java que es.

Spring es un marco de trabajo o framework que facilita el desarrollo de aplicaciones escritas en Java. El propósito de Spring es manejar la infraestructura de las aplicaciones y el programador sólo se centre en la aplicación. Es por ello, que el programador se encargará de programar la lógica de negocio usando objetos simples de Java o POJOs (Plain Old Java Objets) y Spring se encargará de añadir las capacidades de empresa o J2EE a nuestra aplicación. [X]

Spring tiene estas características:

- Simplicidad y acoplamiento débil – permite programar Java de una forma más sencilla. Busca ser simple y se basa en la inyección de dependencias para obtener un acoplamiento débil.

- Funciona como contenedor – que gestiona el ciclo de vida de los objetos y como se relacionan entre ellos. Proporciona una gran infraestructura que permite que el programador se dedique a la lógica de la aplicación.

- Ligero – es muy rápido en tiempo de procesamiento y no es invasivo a la hora de programar.
- Orientado a aspectos – soporta la programación orientada a aspectos, lo que permite facilitar una capa de servicios que son ideales para este tipo de programación como auditoría, o gestión de transacciones.

Spring se utiliza en el proyecto como framework para toda la aplicación, por lo tanto, usar Spring es también un requerimiento no funcional de parte de la empresa que nos permite utilizar la aplicación como esqueleto y fuente de información mediante el base de datos de las instancias.

5.2.7 AngularJS

Para empezar a trabajar con la interfaz de usuario se hizo un estudio previo de las ventajas entre Frameworks como JQuery y AngularJS, debido a que la organización dueña del sistema de evaluación académica dio total libertad a la hora de elegir cual utilizar. Por lo tanto, la elección del framework Javascript del módulo de Curriculum entra como decisión de diseño para el proyecto final.

Con la aparición de los framework de Javascript, tales como JQuery, las páginas web ya no tenían la necesidad de volver a renderizar las páginas cada vez que se necesitaba información del servidor, ya que con la aparición de las llamadas asíncronas se mejoró la experiencia del usuario al darle la posibilidad de no perder el trabajo ya realizado en la página.

JQuery ha hecho un excepcional trabajo al proveer de herramientas que manipulen el DOM de una página, pero no ofrece una guía real de cómo organizar el código en la estructura de la aplicación. Ante la desesperada búsqueda de escribir aplicaciones grandes y fáciles de mantener en Javascript ha dado a luz a un renacimiento de Frameworks de Javascript, entre ellos se encuentra el framework de Google denominado AngularJS.

AngularJS es un framework de aplicaciones web de código abierto que ofrece a un desarrollador una base estable de código con una comunidad enorme y un entorno rico de librerías hechas por la comunidad.

5.3 Diseño

En esta sección tiene como propósito documentar el proceso de desarrollo de parte del equipo encargado del desarrollo del módulo de Curriculum, debido a que el módulo de Curriculum es un proyecto y dicho proyecto está dividido en varias épicas utilizadas para implementar las exigencias de los clientes de Estados Unidos.

Una épica se encuentra dividida en varias historias de usuario, donde las historias de usuario tienen el propósito de entregar valores de negocio al cliente en un periodo máximo de 2 semanas. Estas historias de usuario pueden ser a la vez divididas buscando la simplicidad de las historias donde se busca que se encuentren de ser posible bajo el estándar INVEST (independientes, negociables, valoradas, estimables, pequeñas, verificables) de la metodología ágil.

Como una épica consiste en un conjunto de historias de usuario que los desarrolladores podían terminar en un sprint, esta épica tiene una variedad

de historias de usuario. Cada historia de usuario está compuesta de subtareas que tiene como propósito facilitar al desarrollador recordar cuales eran algunas labores pendientes a la hora de desarrollar la historia. Cada tarea debía tener un encargado, pero eso no significaba que esa persona debía hacer sola la implementación.

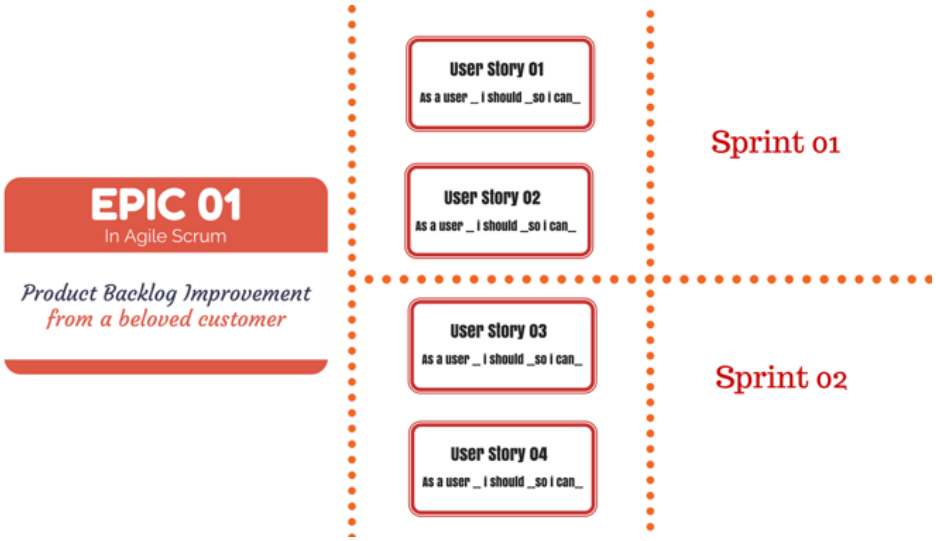

FIGURA 5.5: HACER MI PROPIO DIAGRAMA

El PO (siglas en inglés que significa dueño o responsable del producto) se encarga de la creación de épicas e historias de usuario, en caso de que la historia sea muy grande para terminar en un solo sprint se vuelve a partir en historias más pequeñas.

Cada sprint consta de 2 semanas de trabajo, donde los desarrolladores como equipo se comprometen a entregar cierto valor de negocio que ellos estiman poder terminar en dicho periodo. En caso de que el equipo presienta que la totalidad de historias no van a poder ser entregadas antes de que termine el periodo se pasa al siguiente sprint o se achica la historia con los criterios de aceptación y los criterios de aceptación restantes se agregan en otra historia de usuario para el siguiente sprint.

Cada equipo tiene un líder, donde cada líder tiene como rol ser la brecha que une al PO con los desarrolladores. El PO se reúne con el líder de cada equipo para verificar las prioridades de las historias de usuario que están pendientes en el backlog.

Al terminar una historia de usuario, se deben cumplir los criterios de aceptación para que sea considerada como realizada y los clientes puedan empezar a utilizar las nuevas características del sistema. En caso de que la historia no consiga cumplir los criterios de aceptación correspondientes se considera que la historia no está terminada y que debe mantenerse abierta o se debe volver a abrir en caso de que fue cerrada. Estos criterios de aceptación se verifican primeramente por los mismos desarrolladores antes de cambiar el estado de la historia a resuelto o "done". Luego, la historia pasa a ser verificada por un grupo experto en dominios de didáctica en universidades norteamericanas incluyendo a un PhD en educación. Si en las dos partes la historia es aprobada, entonces pasa al estado de realizado.

Cualquier otro problema o error de código que tenga la nueva característica se debe crear un ticket de bug especificando como reproducir el

problema y el comportamiento esperado. En caso de no poder reproducir este comportamiento se pide más información al respecto o pasa al estado de "won't do" en caso de que el comportamiento ya no se pueda reproducir.

Esta sección va a consistir en explicar cómo se dividieron algunas épicas.

5.3.1 SCRUM

Scrum es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos.

En Scrum se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, Scrum está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad la productividad son fundamentales.

Scrum también se utiliza para resolver situaciones en que no se está entregando al cliente lo que necesita, cuando las entregas se alargan demasiado, los costes se disparan o la calidad no es aceptable, cuando se necesita capacidad de reacción ante la competencia, cuando la moral de los equipos es baja y la rotación alta, cuando es necesario identificar y solucionar ineficiencias sistemáticamente o cuando se quiere trabajar utilizando un proceso especializado en el desarrollo de producto.

5.3.2 Proceso

En Scrum un proyecto se ejecuta en bloques temporales cortos y fijos (sprints o iteraciones que normalmente son de 2 semanas, aunque en algunos equipos son de 3 y hasta 4 semanas, límite máximo de feedback y reflexión). Cada iteración tiene que proporcionar un resultado completo, un incremento de producto final que sea susceptible de ser entregado con el mínimo esfuerzo al cliente cuando lo solicite.

El proceso parte de la lista de objetivos o requisitos priorizada del producto, que actúa como plan del proyecto. En esta lista el cliente prioriza los objetivos balanceando el valor que le aportan respecto a su coste y quedan repartidos en sprints y entregas.

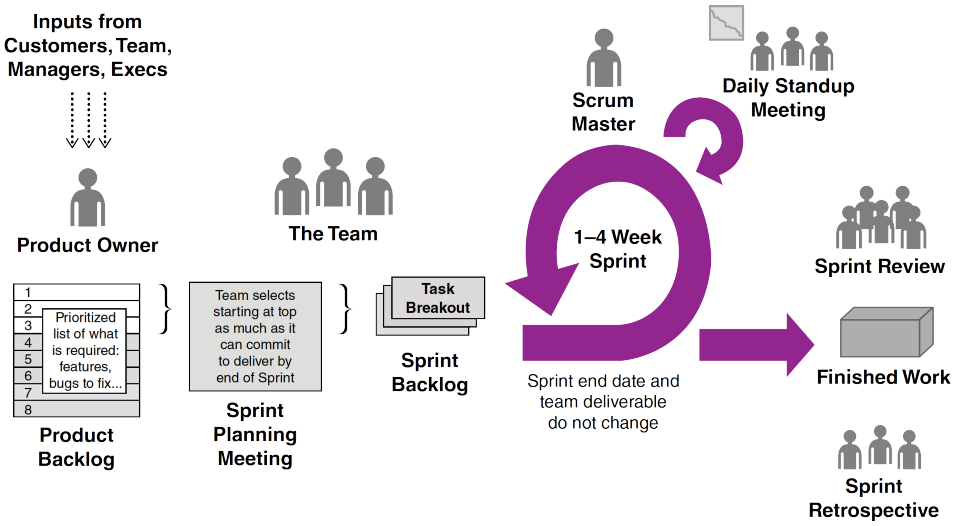

FIGURA 5.6: Flujo de la técnica SCRUM.

5.3.3 Planificación de iteraciones

El primer día de la iteración se realiza la reunión de planificación de la iteración. Tiene dos partes:

- **Selección de requisitos** (4 horas máximo) – El cliente presenta al equipo la lista de requisitos priorizada del producto o proyecto. El equipo pregunta al cliente las dudas que surgen y selecciona los requisitos más prioritarios que se compromete a completar en la iteración o sprint, de manera que puedan ser entregados si el cliente lo solicita.
- **Planificación de la iteración o sprint** (4 horas máximo) – El equipo elabora la lista de tareas de la iteración necesarias para desarrollar los requisitos a que se ha comprometido. La estimación de esfuerzo se hace de manera conjunta y los miembros del equipo se auto-asignan las tareas.

5.3.4 Ejecución del Sprint

Cada día el equipo realiza una reunión diaria (15 minutos aproximadamente). Cada miembro del equipo inspecciona el trabajo que el resto está realizando (dependencias entre tareas, progreso hacia el objetivo de la iteración, obstáculos que pueden impedir este objetivo) para poder hacer las adaptaciones necesarias que permitan cumplir con el compromiso adquirido. En la reunión cada miembro del equipo responde a tres preguntas:

- ¿Qué he hecho desde la última reunión diaria?
- ¿Qué voy a hacer a partir de este momento?
- ¿Qué impedimentos tengo o voy a tener?

Durante la iteración el Scrum Master se encarga de que el equipo pueda cumplir con su compromiso y de que no se merme su productividad. Además, elimina los obstáculos que el equipo no puede resolver por sí mismo.

Durante el sprint, el cliente junto con el equipo refina la lista de requisitos (para prepararlos para los siguientes sprints) y, si es necesario, cambian

o vuelven a planificar los objetivos del proyecto para maximizar la utilidad de lo que se desarrolla y el retorno de inversión.

5.3.5 Inspección y adaptación

El último día de la iteración se realiza la reunión de revisión del sprint. Tiene dos partes:

- **Demostración** (3 horas aproximadamente) – El equipo presenta al cliente los requisitos completados en la iteración, en forma de incremento de producto preparado para ser entregado con el mínimo esfuerzo. En función de los resultados mostrados y de los cambios que haya habido en el contexto del proyecto, el cliente realiza las adaptaciones necesarias de manera objetiva, ya desde la primera iteración, volviendo a planificar el proyecto.

- **Retrospectiva** (1 hora) - El equipo analiza cómo ha sido su manera de trabajar y cuáles son los problemas que podrían impedirle progresar adecuadamente, mejorando de manera continua su productividad. El Facilitador se encargará de ir eliminando los obstáculos identificados.

Capítulo 6

Desarrollo de la aplicación

6.1 Análisis de herramientas utilizadas y posibles potenciales para el desarrollo del módulo

La primera etapa del proyecto fue realizar una encuesta de las capacidades del equipo de desarrolladores donde se completan los conocimientos de los miembros de cada equipo con su líder correspondiente, en la cual se podían apreciar los conocimientos de dominio de la aplicación y cuáles eran las tecnologías o herramientas que estaban familiarizados utilizar para resolver problemáticas. Por lo tanto, luego de hacer dicha cuadrícula en todos los equipos, se seleccionaron las tecnologías y miembros de los equipos que podrían facilitar el desarrollo, siguiendo un enfoque ágil.

6.2 Diseño de modelo de datos para versionamiento de competencias, cursos y programas

6.2.1 Course, SLO and Assessment versioning design spike

Antes de empezar con las historias de versionamiento, hubo un periodo estimado como curva de diseño para adaptar las tablas existentes de los cursos, competencias y evaluaciones de los profesores para que soporten versionamiento. El ticket fue realizado en el sprint 47 con un estimado de puntos de historia de 5.

Al finalizar la historia se llegó con el siguiente esquema de datos:

- Cada tabla de eLumen posee un identificador único. Se agregó un nuevo campo <tabla>_atid que tenía como propósito apuntar al origen de la versión. Por ejemplo; si el usuario crea un nuevo curso para el año lectivo, este curso tiene su identificador y su curso_atid apunta a su mismo identificador por ser el origen de las versiones posteriores. Luego, se crea una nueva versión para el año posterior, esta nueva versión tiene su propio identificador pero su campo curso_atid apunta al primer curso creado u origen. Y así sucesivamente.

- Para hacer más sencilla la búsqueda de competencias, cursos o evaluaciones actuales se agregó un campo a cada tabla identificando los actuales. Este campo denominado is_current o “es actual” es una bandera que indicaba la validez del registro.

- Además de registrar el origen, se registra la versión previa o de donde parte el registro con el campo previous_<tabla>_id.

- Como cada registro de cualquier tabla ahora tiene un periodo de validez, se diseñaron tablas de relación entre cada tabla y la tabla calendario "calendar". Por ejemplo; slo_term_rel para las competencias, new_course_term_rel para los cursos y asmt_term_rel para las evaluaciones.

6.3 Flujo de trabajo para el versionamiento de competencias

6.3.1 SLO Versioning

Esta historia de usuario se inició en el sprint 47, se estimó terminar en un sprint, pero debido a la cantidad de partes que suponía cambios se utilizaron tres sprints para terminar la historia. Se estimó que la historia era de unos 13 puntos, se finalizó en el sprint 49 con un total de 170hs cargadas en el JIRA.

Esta historia de usuario tenía como descripción los siguiente "Como coordinador de eLumen, me gustaría ser capaz de versionar competencias con la finalidad de que se puedan redefinir competencias con el paso del tiempo, sin perder datos de corrección de las mismas".

Algunas tareas que se definieron en la historia de usuario son las siguientes:

- Investigar y diseñar el versionamiento de las competencias. Se desarrolló de la manera en que toda competencia versionada apunta al origen y el origen se apunta a sí mismo, de esta manera se puede saber la familia de versiones de una competencia.
- Actualizar todos esos lugares de la aplicación que listan las competencias, donde solamente deberían traer las competencias actuales.
- Manejar la distribución de competencias a periodos futuros, de manera que una competencia no pueda ser distribuida a periodos en las que no tiene validez.
- Diseñar y mantener pruebas automatizadas con Selenium.

6.3.2 Simple Workflow with Serializer

En la siguiente historia de usuario inicia el proceso de creación de flujos de trabajo. Las plantillas de los Workflow pueden ser creados, editados y eliminados por el administrador encargado de la aplicación de cada universidad. Para esta historia se debe diseñar y desarrollar las plantillas de manera que el administrador pueda agregar los diferentes pasos del Workflow si así lo decide.

Además de que el usuario sea capaz de agregar pasos para la plantilla de Workflow, también puede agregar "named steps". Los named steps son pasos que puede diseñar el usuario, donde puede colocar una pregunta como título y por cada título tiene un campo que puede llenar el usuario. Por lo general, un named step puede tener una o más preguntas definida por el usuario.

Cada paso tiene que ser aprobado por un rol del sistema, donde cualquier usuario con dicho rol puede aprobar o rechazar el Workflow con solo rechazar uno de los pasos. Dando inicio al personalizador de plantillas de

Workflow se comenzó con las competencias, donde se podía asignar un tipo de Workflow para cada plantilla ya sea de creación o versionamiento de los diferentes niveles de competencias.

El que inició el Workflow es el único que puede llenar los campos del formulario.

La historia tiene la siguiente descripción: "Como coordinador de eLumen, me gustaría ser capaz de crear workflows simples para administrar la aprobación de revisiones de competencias, para que se pueda tener un mejor manejo de las creaciones y aprobaciones de las mismas en el campus".

Como las plantillas eran algo ya conocida en otra parte de la aplicación, se imitó el comportamiento de la misma utilizando las mismas tablas para el almacenamiento de los datos en la base de datos relacional. Se diseñaron las nuevas pantallas con la definición de las plantillas de Workflow y también la pantalla de listado de los workflows donde el usuario administrador puede crear, editar si aún no ha sido usada, eliminar y clonar plantillas.

La historia fue estimada con 8 puntos de historia y fue cerrada en el sprint 48.

6.3.3 Approval Workflow Execution for SLOs

Luego de la historia de diseño de las plantillas para creación o versionamiento de competencias, el siguiente paso es que un usuario designado desde la plantilla pueda iniciar y completar un flujo de trabajo para crear o revisar una competencia de cualquier nivel.

Cada paso que debe completar el usuario debe estar completa antes de pasar a la etapa de revisión por parte de los encargados. Luego de enviar el formulario, cada rol debe hacer su revisión para que el sistema pueda agregar la nueva competencia.

La historia tiene la siguiente descripción: "Como aprobador de una revisión o creación de competencias, me gustaría un simple paso-a-paso para que pueda de manera fácil revisar y/o aprobar competencias".

Como en las reuniones de demostración de cada sprint se notaban ciertos aspectos de las historias de usuario que no llenaban las expectativas de los clientes, los desarrolladores decidieron diseñar maquetas de pantallas que mostraban el posible diseño de la página. Luego de recibir feedback de parte de los clientes, se empezaba a desarrollar las nuevas pantallas. Finalizando la historia de usuario con pruebas automatizadas.

La historia se finalizó en el sprint 48 con una cantidad de 5 puntos de historia estimados por el equipo de desarrolladores, en un periodo de tiempo de 44hs.

6.3.4 Reject Workflow Steps

Esta historia de usuario tiene como propósito permitir a la persona que hace la revisión de los pasos rechazar y dejar feedback para que se puedan hacer los cambios correspondientes. Cuando se rechaza un paso, se rechaza un Workflow, y por lo tanto vuelven a estar activos los campos para que se hagan los cambios correspondientes.

La historia tiene como descripción: "Como aprobador de Curriculum de eLumen, me gustaría ser capaz de rechazar ítems de workflows y poder dar feedback a partes que no cumplen con nuestros estándares".

El trabajo se inició la actualización del modelo de base de datos actual, luego de crear las clases correspondientes en el código para su utilización. Luego, se actualizaron las páginas donde el usuario puede aprobar los steps para que soporte rechazar pasos y poder así dejar algunos comentarios.

Se estimó con 5 puntos de historia y tuvo una duración de 1 sprint con un total de 53hs de desarrollo.

6.4 Buzón de entrada para evaluadores y colaboradores del flujo de trabajo

6.4.1 Workflow Queue Visibility or Inbox

La siguiente historia tiene como propósito mostrar a cada usuario la lista de workflows pendientes que requiere de su aporte. Además de adaptar el nuevo buzón de entrada para otros rasgos de la aplicación como son las evaluaciones, los planes de acción y preguntas de parte del usuario a profesores.

La historia tiene como descripción: "Como aprobador de eLumen, me gustaría una vista unificada de los workflows que tengo que revisar – además de mis evaluaciones, planes de acción y mis preguntas a profesores – para que no vaya cazando workflows por la aplicación"

Las tareas de la historia de usuario eran las de crear la página que listen los workflows inicialmente, luego de ese hacer las pruebas correspondientes. Luego de que funcione la lista de workflows, agregar los planes de acción y RFI en la lista a la misma lista y volver a hacer las pruebas de funcionamiento.

Se estimó con 5 puntos de historia y tuvo una duración de 2 sprints debido a inconvenientes en el camino con un total de 56hs de desarrollo.

6.4.2 Cycle Time Notifications for Workflow Steps

https://elumen.atlassian.net/browse/EL-3986

6.5 Versionamiento encadenado de evaluaciones debido al versionamiento de competencias

6.5.1 Versioning for Assessments

La historia de versionamiento de evaluaciones tiene como propósito permitir el versionamiento automático de evaluaciones existentes que utilizar competencias del sistema. Por ejemplo, en caso de que una evaluación hecha por un profesor tenga una nueva versión en el nuevo periodo de su sección, el sistema versiona la evaluación para ese periodo obteniendo las competencias actuales.

Esta historia de usuario se inició en el sprint 48 con un total de 13 puntos de historia, se finalizó en el sprint con un total de 87hs cargadas en el JIRA.

Esta historia de usuario tenía como descripción los siguiente "Como usuario de eLumen, me gustaría ser capaz de versionar mis evaluaciones, para que se observen los cambios a través del tiempo (y que la interfaz y los reportes sigan presentando datos para los diseños históricos)".

Algunas de las tareas de la historia fueron las siguientes:

- Adaptar versionamiento para el modelo de datos de las evaluaciones.
- Actualizar la biblioteca de evaluaciones de los usuarios para que soporte versionamiento de las mismas.
- Actualizar el selector de evaluaciones de los profesores, que puedan seleccionar evaluaciones actuales.
- Actualizar el widget de profesores que utilizan las evaluaciones como datos.

6.6 Flujo de trabajo para el versionamiento de cursos

6.6.1 Course Versioning

Esta historia de usuario se inició en el sprint 47, se estimó terminar en un sprint, pero debido a la cantidad de partes que suponía cambios se utilizaron dos sprints para terminar la historia. Se estimó que la historia era de unos 13 puntos, se finalizó en el sprint 49 con un total de 96hs cargadas en el JIRA.

Esta historia de usuario tenía como descripción los siguiente "Como coordinador de eLumen, me gustaría poder hacer una versión de mi plan de curso para que pueda realizar un seguimiento de los cambios para cosas como la revisión de programas y los acuerdos de articulación y transferencia en eLumen.".

Algunas de las tareas realizadas en la historia fueron las siguientes:

- Buscar técnicas y herramientas de versionamiento para verificar posibles implementaciones parecidas para implementar.
- Luego de buscar algunas técnicas de versionamiento, diseñar una posible solución a la problemática.
- Implementar cambios en la base de datos mediante scripts en el proyecto.
- Actualizar clases de Java existentes en el proyecto de cursos.
- Implementar la solución para el flujo de creación de cursos de Curriculum, conocida como Workflow en inglés. Además, incluir nuevas clases de Java para las mismas.
- Actualizar la creación de curso con los cambios aplicados mediante scripts de base de datos.
- Adaptar la relación de cursos y competencias para que soporte el versionamiento de los mismos.
- Actualizar la lista de competencias por cursos.

6.6.2 Course Detail: Course Cover Info

Esta historia de usuario tiene como propósito de diseñar páginas que permitan al usuario completar la información básica de curso que buscan diseñar.

Tiene la siguiente descripción: "Como miembro del comité de Curriculum, me gustaría ser capaz de administrar la página de información básica de cursos, para que no tenga que buscar por documentos a la hora de crear o versionar cursos".

- Diseñar un modelo de datos que soporte el nuevo paso de información de curso.
- Adaptar tablas existentes y crear clases nuevas para las nuevas entidades de base de datos.
- Actualizar la plantilla de creación de Workflow para que soporte el nuevo paso.
- Actualizar el visualizador de Workflow.
- Diseño de pruebas automatizadas.

La historia fue estimada con 5 puntos de historia y se terminó en un sprint con un total de 53hs de desarrollo.

6.6.3 Course Details: Units and Hours

En esta historia se desarrolló un nuevo paso para el desarrollo de Workflow, en la cual el que inició el mismo va a poder detallar las horas y unidades que requiere el curso o cree que se va a requerir.

La organización proveyó de algunas muestras de cómo debería ser la página y era un criterio de aceptación de parte del ticket que siga el modelo de la misma.

La historia tenía la siguiente descripción: "Como aprobador de Curriculum de eLumen, me gustaría tener una página de horas y métricas para que pueda conseguir información básica sobre mi curso en eLumen".

La historia a desarrollar se dividió entre miembros del equipo de desarrollo en las siguientes tareas:

- Crear scripts en la base de datos para adaptar el modelo de datos para que soporte los nuevos campos de curso.
- Crear y/o editar las clases de las entidades de Java que utiliza o utilizará la aplicación.
- Actualizar la plantilla de workflows.
- Actualizar el visualizador de workflows.
- Pruebas de funcionalidad.

La historia fue estimada con 3 puntos de historia y se terminó en un sprint con un total de 66hs de desarrollo.

6.6.4 Curriculum: Course Specifications

En esta historia de usuario se desarrolló un nuevo paso para el Workflow, el cual era un paso de tipo formulario en la cual el que inició el Workflow puede agregar objetivos, información acerca de los métodos de evaluación de la materia, algunos equipos requeridos y libros que se necesitará en el curso.

La organización proporcionó de modelos de pantallas para el nuevo paso y era un criterio de aceptación de parte del ticket que siga el mismo formato.

La historia de usuario proporciona la siguiente descripción: "Como especialista de Curriculum de eLumen, me gustaría ser capaz de agregar o editar especificaciones de curso como parte del Workflow de creación y/o versionamiento de mi curso, con el objetivo de no hacerlo en papel."

Las tareas fueron separadas y desarrolladas por los desarrolladores y eran las siguientes:

- Crear scripts de base de datos para que soporte el nuevo formato de cursos.
- Crear y/o editar clases de Java para las nuevas entidades de la base de datos.
- Actualizar la página de creación y/o edición de plantillas de Workflow para que soporte el nuevo paso.
- Actualizar el visualizador de Workflow.
- Actualizar los servicios de guardado para creación y versionamiento de cursos y workflows.
- Actualizar el servicio de aprobación de Workflow.
- Test de la historia.

La historia fue terminada en el sprint 50 con 5 puntos de historia de usuario y 40hs de desarrollo cargadas en el sistema

6.6.5 Prereq & Entrance Skills

Esta historia tiene como objetivo que el usuario pueda colocar una lista de cursos como pre-requisitos, co-requisitos, anti-requisitos y recomendaciones para su nuevo curso. Además de ciertas capacidades que el alumno debe tener como requisito para tomar el curso.

Para entrar un poco en contexto de la historia vamos a definir cuáles son los tipos de requisitos que puede tener un curso:

- **Pre-requisito:** es un tipo de requisito que impide al usuario tomar o cursar un curso sin haber pasado antes del curso que está como pre-requisito.
- **Co-requisito:** es un tipo de requisito impide al usuario tomar o cursar un curso si no toma también el curso que tiene como co-requisito.
- **Anti-requisito:** es un requisito impide al usuario tomar o cursar un curso si ya curso o va a cursar un curso que tiene como anti-requisito.

- **Recomendación:** es una recomendación de parte del sistema que curso tomar para aprovechar mejor la materia o curso. Es opcional.

La historia de usuario tiene como descripción: "Como especialista de Curriculum de eLumen, me gustaría ser capaz de introducir requisitos para cursos y capacidades de entrada en la creación o revisión de Workflow, para que podamos seguir durante su desarrollo y aprobación".

La historia fue dividida en partes para que los desarrolladores puedan trabajar en partes independientes durante el proceso de la misma, y eran las siguientes:

- Crear scripts para el nuevo modelo de datos.
- Generar o editar clases de entidades para el nuevo modelo de datos.
- Actualizar la plantilla de Workflow para que soporte un nuevo paso.
- Actualizar el visualizador de Workflow.
- Actualizar los servicios de guardado y aprobación.
- Pruebas de funcionalidad.

La historia de usuario fue cerrada en el sprint 50 con 3 puntos de historia y 44hs de desarrollo cargadas.

6.6.6 Course Creation/Approval Review

La historia de usuario tenía como criterio de aceptación los siguientes puntos:

- Las páginas para revisar los Workflows tienen una región de retroalimentación o feedback debajo de cada paso, con la opción de ocultar y mostrar, para que el usuario que está revisando el Workflow en desarrollo pueda dejar comentarios al encargado de la creación o versionamiento del curso.
- La UI tiene elementos de status que indican que cierta parte es "new" o nueva, "approved" o aprobada y "rejected" o rechazada.
- Los pasos tienen regiones que permiten aceptar o rechazar los campos propuestos por los desarrolladores del curso. Por lo tanto, deben tener elementos de UI que indiquen al usuario que puede aprobar o rechazar cada parte.

Además de los criterios de aceptación, había que volver a actualizar el buzón de entrada para que acepten los cambios que tiene la historia de usuario. Debido a que más de una persona puede revisar el Workflow y podría trancar el proceso si es que no se le notifica debidamente que hay nuevos cambios que revisar.

Algunas de las tareas descompuestas de la historia de usuario son las siguientes:

- Actualizar el visualizador de Workflow para que pueda soportar la nueva característica de aprobación o rechazo de cada parte.

- Actualizar el buzón de entrada de Cursos.
- Pruebas de funcionamiento.

La historia se cerró en el sprint 50 con 3 puntos de historia y 68hs cargadas de desarrollo

6.6.7 Learning Outcomes

Esta historia tiene como propósito de crear o versionar competencias para el curso a ser creado o versionado.

La organización proporcionó de modelos de pantallas para el nuevo paso y era un criterio de aceptación de parte del ticket que siga el mismo formato.

La historia de usuario tiene como descripción: "Como especialista de Curriculum de eLumen, me gustaría ser capaz de articular las competencias de mi nuevo curso".

Como criterio de aceptación de la historia fue la de agregar el Workflow de competencias en el Workflow de cursos. Algunas de las tareas de la historia fueron:

- Modificar la base de datos para que soporte el nuevo modelo de datos de las competencias dentro de Workflow de curso.
- Modificar o agregar clases de las entidades que van a ser usadas durante la historia.
- Actualizar la plantilla de Workflow para que soporte el nuevo paso para la creación o versionamiento de competencias.
- Actualizar el visualizador de Workflow para que soporte el nuevo paso de competencias.
- Actualizar los servicios de guardado y de versionamiento de cursos y competencias.
- Pruebas de nuevas funcionalidades.

La historia fue cerrada en el sprint 51 e iniciada en el sprint 50 con un total de 76hs cargadas en el sistema.

6.6.8 Curriculum: Assign Classification Codes

La siguiente historia tiene como propósito la de asignar Classification Codes a los cursos.

Para entrar en contexto, habría que definir primero que son los Taxonomy of Programs (TOP) o taxonomía de programas en español.

La taxonomía de programas o TOP es un sistema numérico de códigos usados a nivel de Estado para recolectar y reportar información en cursos y programas, en diferentes instituciones educativas [sacado de TOP manual] por todo el Estado.

TOP ha sido diseñado para agregar información acerca de los programas. Sin embargo, un código TOP debe ser asignado a cada curso del sistema. Aunque el TOP no contiene tantas opciones específicas como lo haría

un sistema diseñado para cursos, a cada curso se le debe dar el código TOP que se aproxima a describir el contenido del curso.

Algunos usos a los códigos TOP:

- En el inventario de programas aprobados y rechazados, para tener información que tipos de cursos y programas son ofrecidas por el Estado.
- En bases de datos de administración de información, para recolectar y reportar información en logros estudiantiles (licenciaturas y certificados) en ciertos programas.
- En contabilidad vocacional estudiantil, para reportes de compleción de programas y cursos de ciertos programas vocacionales.

La descripción de la historia fue la siguiente: "Como miembro del comité curricular, me gustaría ser capaz de asignar a mis cursos de códigos de clasificación como parte de la aprobación de mis workflows para asegurar que estén correctos, como esto es motivo de rechazo en la oficina del canciller del Estado".

Algunas de las tareas fueron las siguientes:

- Diseño del nuevo modelo y creación de scripts de base de datos, donde se debían generar tablas para cada nueva entidad del modelo de datos ajustado para las taxonomías de programas. Además, cargar todos los datos de códigos de cursos existentes para el estado de California.
- Creación de clases Java de las nuevas entidades de la base de datos.
- Crear páginas CRUD de las nuevas tablas (disciplina, sub-disciplina, campo).
- Diseño e implementación de la nueva página de asignación de códigos de clasificación para los cursos en proceso de diseño.
- Hacer servicios para cada una de las nuevas páginas.
- Pruebas de funcionalidad.

La historia de usuario, con 5 puntos de historia, fue terminada con un total de 80 horas en el sprint 51.

6.7 Flujo de trabajo para el versionamiento de programas de estudio

6.7.1 Program workflow & Cover Info

https://elumen.atlassian.net/browse/EL-3820

6.7.2 Program Learning Outcomes Workflow Step

https://elumen.atlassian.net/browse/EL-3821

6.7.3 Course Blocks for Program

https://elumen.atlassian.net/browse/EL-3822

6.7.4 Visualize Changes in ALL fields in Curriculum [UX]

https://elumen.atlassian.net/browse/EL-4585

6.8 Soporte de etapas en los flujos de trabajo

6.8.1 Improved Workflow Steps behaviours for Roles

https://elumen.atlassian.net/browse/EL-3938

6.8.2 Editor and Creator Roles for Curriculum Steps

https://elumen.atlassian.net/browse/EL-3937

6.8.3 Curriculum Workflow Behavior: Composable Steps/Stages

https://elumen.atlassian.net/browse/EL-4084

6.8.4 Optional steps/parts per stage in workflow review

https://elumen.atlassian.net/browse/EL-4257

6.9 Reportes de versiones de cursos

6.9.1 Course Outline of Record (COR) Report

https://elumen.atlassian.net/browse/EL-3941

6.10 Soporte de versionamiento en el AMS

6.10.1 TOP-CIP Crosswalk UI

https://elumen.atlassian.net/browse/EL-3943

6.10.2 Rename/Reorganize Tabs for better Curriculum Module Appearance

https://elumen.atlassian.net/browse/EL-3987

6.10.3 Improved Filter/View for Course/Program Listings

https://elumen.atlassian.net/browse/EL-4035

6.10.4 Program CRUD should work without Curriculum

https://elumen.atlassian.net/browse/EL-4038

6.10.5 Curriculum Listing (Course/Program) Improved View

https://elumen.atlassian.net/browse/EL-4086

6.10.6 Report Support for SLO, Course and Program version

https://elumen.atlassian.net/browse/EL-4040

6.11 Retoques finales

6.11.1 Update Course Workflow

https://elumen.atlassian.net/browse/EL-4524

Capítulo 7

Validación del desarrollo

7.1 Validación por parte del equipo educativo

7.2 Validación por parte de los usuarios

Capítulo 8

Conclusión

Bibliografía

[1] *NET application architecture guide*. 2nd ed. Patterns & practices. OCLC: ocn320803280. Redmond, Wash.: Microsoft, 2009. ISBN: 978-0-7356-2710-9.

[2] George D. Kuh et al. *Knowing What Students Know and Can Do: The Current State of Student Learning Outcomes Assessment in U.S. Colleges and Universities*. English. Tech. rep. National Institute for Learning Outcomes Assessment, Jan. 2014.

[3] George D. Kuh, ed. *Using evidence of student learning to improve higher education*. First edition. San Francisco, CA: Jossey-Bass, 2015. ISBN: 978-1-118-90339-1.

[4] Bill Boyle and Marie Charles. *Curriculum development: a guide for educators*. OCLC: ocn919483148. Los Angeles: SAGE, 2016. ISBN: 978-1-4462-7330-2 978-1-4462-7329-6.

[5] Thomas Stober et al. "Overview of Agile Software Development". In: *Agile Software Development*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 35–59. ISBN: 978-3-540-70830-8. URL: http://dx.doi.org/10.1007/978-3-540-70832-2_3.

[6] David A. Erlandson, ed. *Doing naturalistic inquiry: a guide to methods*. Newbury Park, Calif: Sage, 1993. ISBN: 978-0-8039-4937-9 978-0-8039-4938-6.

[7] Colin Robson. *Real world research: a resource for users of social research methods in applied settings*. eng. 3. ed. OCLC: 729956086. Chichester: Wiley, 2011. ISBN: 978-1-4051-8240-9 978-1-4051-8241-6.

[8] Jonathan Lazar, Jinjuan H. Feng, and Harry Hochheiser. *Research Methods in Human-Computer Interaction*. 1st ed. Published: Paperback. Wiley, Feb. 2010. ISBN: 0-470-72337-8. URL: http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/0470723378.

[9] Ronald S. Carriveau. *Connecting the dots: developing student learning outcomes and outcomes-based assessments*. Second edition. Sterling, Virginia: Stylus Publishing, LLC, 2016. ISBN: 978-1-62036-479-6 978-1-62036-480-2.

[10] Casey A. Barrio Minton, Donna M. Gibson, and Carrie A. Wachter Morris. *Evaluating student learning outcomes in counselor education*. Alexandria, VA: American Counseling Association, 2016. ISBN: 978-1-55620-337-4.

[11] Wil van der Aalst and Kees van Hee. *Workflow management: models, methods and systems*. eng. 1. MIT Press paperback ed. Cooperative information systems. OCLC: 255171394. Cambridge, Mass.: MIT Press, 2004. ISBN: 978-0-262-72046-5 978-0-262-01189-1.

[12] Rebecca Cartwright, Ken Weiner, and Samantha Streamer-Veneruso. "Student learning outcomes assessment handbook". In: *Montgomery County, MD: Montgomery College* (2009).

[13] Megan Oakleaf, Jackie Belanger, and Carlie Graham. "Choosing and Using Assessment Management Systems: What Librarians Need to Know". In: *ACRL* (Apr. 2013). URL: http://www.ala.org/acrl/sites/ala.org.acrl/files/content/conferences/confsandpreconfs/2013/papers/OakleafBelangerGraham_Choosing.pdf.

[14] Angela Di Michele Lalor. *Ensuring high-quality curriculum: how to design, revise, or adopt curriculum aligned to student success*. Alexandria, VA: ASCD, 2017. ISBN: 978-1-4166-2279-6.

[15] RM Harden. "AMEE Guide No. 21: Curriculum mapping: a tool for transparent and authentic teaching and learning". In: *Medical teacher* 23.2 (2001), pp. 123–137.

[16] Gary West. "Technology tools to make educational accountability work". In: *THE Journal (Technological Horizons In Education)* 28.5 (2000), p. 60.

[17] Brad A. Myers and Andrew J. Ko. *The Past, Present and Future of Programming in HCI*. Feb. 2009. URL: http://www.academia.edu/2669908/The_past_present_and_future_of_programming_in_HCI.

[18] Michael Wittig and Andreas Wittig. *Amazon Web Services in action*. OCLC: ocn934475856. Shelter Island, NY: Manning, 2016. ISBN: 978-1-61729-288-0.

[19] Nitu Gupta and Varshapriya JN. "Software as a Service". English. In: *International Journal of Innovative Research in Advanced Engineering (IJIRAE)* 1.6 (2014). ISSN: 2349-2163.

[20] Santosh Kumar and R. H. Goudar. "Cloud Computing – Research Issues, Challenges, Architecture, Platforms and Applications: A Survey". English. In: *International Journal of Future Computer and Communication* 1.4 (Dec. 2012), p. 356.

[21] Jihyun Kang. "Web based development Framework for Customizing Java-based Business Logic of SaaS Application". In: *ICACT* (2012).

[22] Christopher W. H. Davis. *Agile metrics in action: how to measure and improve team performance*. OCLC: ocn907160912. Shelter Island, NY: Manning, 2015. ISBN: 978-1-61729-248-4.

[23] Charles G Cobb. *The project manager's guide to mastering Agile: Principles and practices for an adaptive approach*. John Wiley & Sons, 2015.

[24] Ben Shneiderman and Catherine Plaisant. *Designing the user interface: strategies for effective human-computer interaction*. 5th ed. OCLC: ocn244066651. Boston: Addison-Wesley, 2010. ISBN: 978-0-321-53735-5.

[25] Jakob Nielsen. *Usability engineering*. eng. Nachdr. OCLC: 760142137. Amsterdam: Kaufmann, 2010. ISBN: 978-0-12-518406-9.

[26] Jaime Levy. *UX strategy: how to devise innovative Digital products that people want*. First edition. Beijing ; Sebastopol: O'Reilly Media, 2015. ISBN: 978-1-4493-7286-6.