Universidad de las Ciencias Informáticas

Facultad 4

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Aplicación de escritorio para los estudiantes del software ATcnea

Autor:

Victor M Plasencia Costales

Tutores:

MSc. Mailin Carballosa Infante

Ing. Odenys Almora Rodríguez

Ing. Yaritza Bárbara González Ramírez

La Habana, julio de 2017

Año 59 de la Revolución

Declaración de autoría

Declaro que soy el autor del trabajo “Aplicación de escritorio para los estudiantes en el software ATcnea” y delego a la Universidad de Ciencias Informáticas los derechos patrimoniales de la misma con carácter exclusivo.

Para que así coste firmo la presente a los ___ días de mes _____del año_____.

Autor:

_____

Victor M Plasencia Costales

Tutores:

_____ _____

Ing. Yaritza Bárbara. González Ramírez Ing. Odenys Almora Rodríguez

_____

MSc. Mailin Carballosa Infante

Dedicatoria

A mi madre, que me acompañó en este largo camino, dándome fuerzas y ánimos, contando con todo su apoyo y sacrificio; teniendo su enseñanza como estandarte de mi moral y recibiendo todo su amor.

A mi padre y amigo, que esta, siempre presente en todas mis acciones, en mis dificultades, dándome su apoyo y en mis logros, festejando junto a mí. Su ejemplo ha forjado a un dedicado hijo.

A mi novia que fue la luz cuando creía que no había salida y que su amor me hizo vencer barreras que parecían infranqueables.

A mi hermana que ha sido mi complemento en todos los momentos de mi vida, dándome su amor, tomando sus consejos y esforzándome cada día más para que vea en mi un ejemplo a seguir.

A mi abuela y familia por constar por su preocupación en todo momento.

Victor M Plasencia Costales

Agradecimientos

A mis padres por haberme mostrado su apoyo moral y material, por su comprensión y confianza en mí.

A mis tutores Yaritza, Odenys y Mailin; por ser mi mano derecha y la guía para la realización de este trabajo, contando siempre con su ayuda y dedicación. Sirviéndome como ejemplo de excelentes profesionales. Sus consejos e indicaciones son el reflejo del resultado de dicho trabajo.

A mi amigo Odenys, que por este curso ha sido como un hermano mayor para mí, atento siempre a sus consejos y aprendiendo de su experiencia y conocimiento. Gracias por tener la oportunidad de conocerte y por haber recorrido este camino bajo tu protección.

A mi amiga Yaritza por ser tan preocupada y exigente en todo este largo proceso, por ayudarme incondicionalmente, por tu sacrificio y entrega en la elaboración de este trabajo.

A mi novia por siempre estar ahí para mí, cuando más lo necesitaba en las buenas y en las malas, ayudándome en todo lo posible.

A mis profesores de la Universidad, que en muchas líneas de este trabajo está reflejado la educación y el conocimiento que he recibido a lo largo de la carrera.

A mis amigos del aula por compartir cinco años conmigo, por contar con su preocupación y formar parte de momentos de alegría y por afrontar retos y dificultades juntos.

A todas aquellas amistades que la universidad me dio la oportunidad de conocer que, de una manera u otra, ya sea por su preocupación o por su ayuda, hoy estoy aquí leyendo estas palabras.

Resumen

El desarrollo alcanzado en la actualidad por las Tecnologías de la Información y las Comunicaciones (TIC) ha potenciado un gran impacto en todas las esferas de la vida, principalmente en el Proceso de Enseñanza-Aprendizaje (PEA). El mismo se encuentra en contante cambio, en gran medida debido al avance adquirido por las TIC, siendo fundamental para su desarrollo la intervención de los profesores y estudiantes. Actualmente durante las actividades docentes, los alumnos asumen un rol más activo haciendo uso de contenidos y herramientas digitales creadas para apoyar el PEA. El objetivo de la presente investigación es desarrollar una aplicación de escritorio en el lenguaje Java para los estudiantes del *software* ATcnea que se adapte a las tecnologías de los entornos educativos, lográndose que el estudiante interactúe con la aplicación que gestiona la clase mediante una PC. Para su cumplimiento se realiza un estudio de los principales elementos teóricos del proceso de gestión de aulas. Se analizaron los principales *softwares* existentes en el mundo destinados a la gestión de aulas. Además, se selecciona justificadamente las tecnologías, herramientas, lenguajes y metodología a utilizar en el desarrollo de la aplicación, obteniéndose como resultado una solución que cumpla con los objetivos propuestos. También se utilizan pruebas de caja blanca y caja negra para garantizar la calidad de la solución.

Palabras clave: aplicación de escritorio, entornos educativos, gestión de aulas.

Índice de Contenido

INTRODUCCIÓN

En los últimos años, la implantación de la sociedad de la información y del conocimiento en todos los estamentos de la sociedad es un hecho incuestionable. El aprendizaje a lo largo de la vida es una de las claves de la educación de los ciudadanos del siglo XXI. El éxito en la sociedad del conocimiento requiere de toda la capacidad. Es importante llevar a cabo aprendizajes de diversa naturaleza a lo largo de la vida; adaptarse rápido y eficazmente a situaciones sociales, laborales y económicas cambiantes (Escobar, 2007).

Las TIC poseen un potencial para apoyar el aprendizaje. El creciente avance adquirido por las mismas ha propiciado que nuevas tecnologías se inserten en la educación para reforzar el aprendizaje, aumentar la motivación, favorecer la búsqueda, estimular el desarrollo del razonamiento e incentivar el trabajo grupal y colaborativo (Belloch, 2014).

Actualmente son disímiles las herramientas y dispositivos que complementan el PEA como resultado del auge alcanzado por las TIC. La mayoría se encuentran online en Internet, desde sitios web, blog personal, plataformas de gestión del conocimiento y sistemas tecnológicos[1]. Debido al avance que ha ido adquiriendo la tecnología, las aplicaciones ya no solo son accedidas desde ordenadores, sino que se han ido creando una gama de dispositivos digitales como son tabletas o teléfonos inteligentes capaces de lograr el mismo propósito.

Las aulas interactivas o tecnológicas pertenecen a esta familia, surgen producto de la aplicación de las nuevas tendencias educativas y su vinculación con las tecnologías. Estas últimas son una realidad actual que constituyen una solución educativa contemporánea para el método de enseñanza-aprendizaje, brindando una experiencia única en el aula. Tienen como objetivo la creación de un ambiente colaborativo, que propicia la introducción de tecnologías; parte de la didáctica y enriquece el contenido académico de la figura docente de que se trate. Permite a profesores y alumnos establecer una profunda comunicación, cuya interactividad en el intercambio de ideas e información, motivará la participación y profundización en los temas objeto de estudio (Garrido, 2001).

[1] Sistemas tecnológicos: conjunto de dispositivos físicos que tributan al desarrollo de la educación

El país está consciente de que una sociedad para ser más eficaz, eficiente y competitiva debe aplicar la informatización en todas sus esferas y procesos. En este sentido, Cuba ha identificado desde muy temprano la conveniencia y necesidad de dominar e introducir en la práctica social las TIC. Lograr una cultura digital como una de las características imprescindibles del hombre nuevo, facilitaría a la sociedad acercarse más hacia el objetivo de un desarrollo justo, equitativo, sostenible y alcanzable.

Con el triunfo de la Revolución, la educación cubana se ha ido enriqueciendo, proporcionalmente al desarrollo del país. Se han destinado recursos para hacer de las entidades educativas o dedicadas a la capacitación un ejemplo para el mundo. Estas instituciones cuentan con laboratorios de computación equipados con computadoras personales, aulas con un televisor y video, conexión a portales cubanos y multimedias educativas. Las universidades amplían su margen tecnológico a una mayor escala, contando con conectividad a internet, poseen centros de investigación y producción, destinados al desarrollo de sector científico del país. Bajo este entorno de formación se afirma, según la Organización de las Naciones Unidas para la Educación, la Ciencia y la Cultura (UNESCO), que Cuba es el país de América Latina y el Caribe con mayor índice de desarrollo de la educación, con un Producto Interno Bruto (PIB) dedicado a la enseñanza de un 13%. Lograr la soberanía tecnológica ajustada al entorno de formación cubano propone un avance en cuanto a indicadores de calidad en la educación se refiere (Ojeda, 2014).

Durante el curso 2015-2016 comenzó en el Centro de Tecnología para la Formación (FORTES) de la Facultad 4 de la UCI, el desarrollo de un *software* para la gestión de una clase en un aula con el empleo de las TIC, generalmente conocido como aula tecnológica. En este caso, el *software* se ha denominado ATcnea y se ha desarrollado para su funcionamiento sobre tecnología HAIER, ensamblada por La Empresa Industrial para la Informática, las Comunicaciones y la Electrónica (GEDEME) y con la colaboración de soluciones libres brindadas por el Centro de Soluciones Libres (CESOL) de la Facultad 1 de la UCI.

El 1 de marzo del 2016 se inicia el proyecto Desarrollo del producto ATcnea. Nombre dado, luego de un análisis exhaustivo de miembros del proyecto, donde se concluye que, debido a los requisitos identificados y las características del sistema, el adecuado sería: ATcnea,

sinónimo de Aula Tecnológica debido a que, en su primera versión, no se le incluirían elementos de inteligencia artificial que la podrían hacer merecedora del adjetivo inteligente.

Teniendo en cuenta las especificaciones de *hardware* y *software* presentadas por el cliente, y una vez culminado el estudio de sistemas homólogos, el equipo determinó desarrollar la aplicación destinada al profesor, bajo la tecnología Java, en su versión 8.0; y la del estudiante sobre Android 5.0. En junio del 2016 se obtiene la versión beta presentada en la *Feria CUBAINDUSTRIA*, donde la misma tuvo gran aceptación y a su vez se identificaron nuevas necesidades:

- Debido a que actualmente solo se puede utilizar la aplicación para el estudiante desde terminales Android surge la necesidad de permitir la interacción de los estudiantes con la aplicación gestora de la clase desde una PC con sistemas operativo GNU/Linux.
- La potencia del *hardware* de una PC comparado con los terminales Android es considerablemente superior. La aplicación del estudiante puede hacer uso de la potencia de cálculo para ejecutar sus funcionalidades.
- El *hardware* proporcionado por GEDEME tiene limitaciones en la velocidad de la conexión, porque posee infraestructura inalámbrica; mientras que la red cableada alcanza mayor velocidad y estabilidad, variable de suma importancia en la comunicación de las dos aplicaciones que conforman ATcnea.
- Existen limitantes en las aplicaciones que complementan el *software* ATcnea en los terminales Android, las cuales tienen un reducido número de funcionalidades o no se encuentran disponible porque requieren un *hardware* más potente para su ejecución. Estas dificultades no representan obstáculo para las PC, donde existen un gran número de aplicaciones que cuentan con todas sus funcionalidades disponibles.

Por las limitantes antes planteadas y con el objetivo de buscar alternativas de solución a esta situación problémica identificada, se define como **problema a resolver** ¿Cómo extender el uso del aula tecnológica ATcnea, teniendo en cuenta la tecnología disponible en los entornos actuales de formación cubano?

Se ha determinado como **objeto de estudio** de la presente investigación para resolver el problema ilustrado: El proceso de gestión de la clase en las aulas tecnológicas.

Como **campo de acción** queda definido: El proceso de gestión de la clase del aula ATcnea.

El **objetivo general** que se persigue con la investigación es: Desarrollar una versión de escritorio de la aplicación del estudiante del software ATcnea, para aprovechar los entornos actuales de formación cubanos.

Para darle respuesta al objetivo general propuesto se definen los siguientes **objetivos específicos**:

- Realizar un estudio del estado del arte y los principales elementos teóricos relacionados con las aplicaciones de gestión de aulas.
- Realizar un estudio de las principales funcionalidades de los softwares existentes en el mundo encargados de la gestión de aulas.
- Diseñar la propuesta de solución.
- Implementar la propuesta de solución.
- Validar la propuesta de solución mediante pruebas al sistema.

Los **resultados** esperados con esta investigación son:

- Aplicación de escritorio para el estudiante, disponibles para sistemas operativos GNU/Linux.
- Mayor espectro de utilización del producto ATcnea para diferentes entornos tecnológicos.
- Documentación como guía para futuros proyectos y utilización de la aplicación del estudiante.

Para el logro de las metas de esta investigación, atendiendo a las necesidades teóricas de este trabajo, se requiere desde el punto de vista metodológico el empleo de los siguientes **métodos de investigación científicos**:

Analítico-Sintético: Su uso se ha tenido en cuenta para el análisis de teorías, documentos y materiales relacionados con el desarrollo de aplicaciones para aulas tecnológicas arribando a conclusiones que sustenten la necesidad de la investigación.

Análisis histórico-lógico: Utilizado para analizar la evolución histórica de soluciones similares, para el estudio de su comportamiento y la identificación de necesidades para adecuar la propuesta a desarrollar al marco actual de la aplicación.

Método observación: Mediante este método se tomaron experiencias del desarrollo de una clase común, haciéndose énfasis en el estudiante, adaptando a la aplicación, un reflejo de su comportamiento.

El presente trabajo posee la siguiente estructura capitular:

Capítulo I: Fundamentación teórica.

Se abordan los principales conceptos relacionados con el proceso de enseñanza-aprendizaje; con la utilización de herramientas digitales, el papel del estudiante y el profesor dentro de la clase, las aulas tecnológicas como herramientas para potenciar la adquisición y divulgación del conocimiento; sus características principales, ventajas y el papel del profesor dentro de esta nueva forma pedagógica. Se definen aspectos teóricos que rigen la investigación y se precisan las técnicas, tecnologías y herramientas.

Capítulo II: Propuesta de solución

Trata los aspectos de diseño a través de artefactos y diagramas que facilitan la comprensión del objetivo propuesto. Se realiza una descripción minuciosa de la propuesta de solución, modelo del dominio, patrones arquitectónicos, patrones de diseño y modelo de datos.

Capítulo III: Implementación y Pruebas

Se obtiene la descripción del proceso de implementación de la herramienta a través de los diagramas de componentes. La fase de prueba se ilustra con el desarrollo de las pruebas de sistema y aceptación, donde es realizado un levantamiento de las deficiencias de la aplicación.

CAPÍTULO I: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción del capítulo

La realización de un estudio sobre el desarrollo de técnicas y aplicaciones informáticas relacionadas con el PEA cobra una remarcada importancia para la consecución de los objetivos de este trabajo. El diverso espectro de *software* ha permitido el despliegue de una serie de herramientas digitales para apoyar los procesos vinculados en la enseñanza y el aprendizaje, como por ejemplo las aulas tecnológicas donde el estudiante y el profesor son usuarios que intervienen y comparten escena con la tecnología y el contenido de la clase.

1.2 Proceso de enseñanza-aprendizaje

La enseñanza y el aprendizaje es un binomio que se relaciona para lograr la adquisición de conocimientos, ideas y experiencias. El aprendizaje es la forma de cómo se adquieren o perfeccionan las habilidades y los conocimientos que se manifiesta en la conducta y contribuyen a la resolución de situaciones concretas que realiza en su interior el estudiante.

Autores como Nogueral, explican que: *"(…) la enseñanza se centra en la realización de actividades significativas respecto a contenidos conceptuales que permiten desarrollar la ejercitación de los procedimientos en una secuencia ordenada de explicaciones que conlleven al dominio de cada uno de ellos, con una actitud crítica-reflexiva. Es así, como el proceso de enseñanza con todos sus componentes debe considerarse como un sistema estrechamente vinculado con la actividad práctica de los estudiantes (…)"* (Benitez, 2007).

Por tanto, el proceso de enseñanza-aprendizaje debe concebirse y desarrollarse desde una perspectiva desarrolladora, donde se promueva el desarrollo integral de la personalidad, y se potencie el tránsito progresivo de la dependencia a la independencia. Para lograr este proceso de enseñanza-aprendizaje en los estudiantes es necesario que el profesor asesor en su labor de dirección atienda a los indicadores que caracterizan el proceso como lo son: la planificación, la ejecución y la evaluación del proceso de enseñanza-aprendizaje (Domínguez, 2012).

En la actualidad, las nuevas tecnologías se integran a la actividad humana, de modo tal que se introducen cambios significativos en la sociedad actual. Las mismas constituyen un excelente recurso didáctico a utilizar para aprovechar las posibilidades que ofrecen en las

distintas áreas y niveles educativos, para incidir de forma positiva y dinámica en el PEA, y para motivar al alumno a desarrollar su aprendizaje de forma independiente y activa.

1.3 Herramientas digitales para el aprendizaje

Las aulas tradicionales ilustran el pizarrón de tizas, las paredes colmadas de objetos de aprendizaje y pupitres para los alumnos. Paulatinamente se fueron insertando los ordenadores, televisores, proyectores digitales y otras tecnologías cuya finalidad es facilitar la tarea de enseñanza de los docentes y el proceso de aprendizaje de los alumnos. Este nuevo ámbito extiende la labor educativa fuera del escenario del centro docente y activa el papel, tanto de los alumnos como del profesorado, en el contexto escolar. Participación, colaboración o interacción son algunos de los términos que la herramienta educativa ha integrado en el vocabulario académico y que reflejan el importante cambio en la metodología y estrategia de enseñanza que se comienza a experimentar ya en muchas instituciones educativas (Reina, 2011).

La definición formal de herramientas digitales, hace referencia:" *a todos aquellos softwares o programas intangibles que se encuentran en las computadoras o dispositivos, donde se realizan todo tipo de actividades*" (Vázquez, 2014).

Una de las grandes ventajas que tiene el manejo de estas herramientas, es que pueden ayudar a interactuar más con la tecnología de hoy día, ayuda a comunicarse, desarrolla competencias y habilidades en los estudiantes para ser utilizadas en la educación. Estas herramientas además de ser un apoyo para el aprendizaje, también dan paso a la innovación de una búsqueda hacia mejores utilizaciones de los materiales. Sin embargo, para que esto se logre con éxito, se necesita tener a un personal capacitado que pueda sacar el mejor provecho posible para crear ambientes de aprendizaje en las aulas y ofrecer las herramientas necesarias que se puedan emplear en situaciones de la vida real.

El uso de las herramientas digitales garantiza (Vázquez, 2014):

- Un medio de comunicación ya que superan las barreras del espacio y el tiempo.
- Permiten que dos o más personas establezcan comunicación por medio de mensajes escritos o video desde distintas partes del mundo en tiempo real.
- La posibilidad de que la información circule de manera rápida y efectiva.
- El trabajo en clase sea más entretenido y provechoso.

- Son un material de apoyo para enriquecer el contenido que se aborda, los alumnos pueden buscar información sobre un tema de su interés.

Se clasifican en (Vázquez, 2014):

- CMS (Content Management System): Un sistema de gestión de contenido es una plataforma ideal para crear y administrar contenido digital. Debido a que los CMS se especializan en el contenido muchos de estos gestores permiten crear documentos, modificarlos y publicarlos en la web sin necesidad que el usuario requiera conocimientos sobre programación. Algunos de estos ejemplos son: Blogs, Wordpress, Blogger, Wikis, Pb Works y Wikia.
- Redes Sociales: Son plataformas web que permiten a los usuarios generar contenido, interactuar y crear comunidades sobre intereses similares. Poseen una interfaz dinámica para compartir datos y fomentar la comunicación. Los datos que se comparten varían desde textos simples, fotos, audio, hasta videos en HD (*High Definition*). Algunos de estos ejemplos son: Twitter, Facebook y Yahoo Respuestas.
- Lector de RSS: RSS son las siglas de Really Simple Syndication, en español se refiere a un formato XML para sindicar o compartir contenido en la web. Se utiliza para difundir información actualizada frecuentemente a usuarios que se han suscrito a la fuente de contenidos. El formato permite distribuir contenidos sin necesidad de un navegador, utilizando un *software* diseñado para leer estos contenidos RSS (agregados). A pesar de esto, es posible utilizar el mismo navegador para ver los contenidos RSS. Algunos de estos ejemplos son: Google Reader, RSS Reader, BlogLines y Freed Reader.
- Multimedias: Un comunicador en la actualidad requiere de herramientas digitales que permitan modificar, retocar y mejorar la calidad de los contenidos que elabora. Algunos de estos ejemplos son: Movie Maker, Picassa, Photoshop online, Soundation y Audacity.
- FTP (Protocolo de Trasferencia de Archivos): Es el servicio que permite transmitir archivos entre sistemas conectados. Por lo general se usa para levantar una página web hacia un *hosting* seleccionado. Algunos de estos *softwares* son FilleZilla y FTP Comander Free.
- Streaming: Es un tipo de tecnología que permite observar y escuchar elementos multimedia sin necesidad de descargar en la computadora. Algunos de los *softwares* utilizados son Ustream y Livestream.

- Marcadores Sociales: Debido a la gran información que se encuentra en internet se hace muy tedioso encontrar páginas de interés. Es por eso que para solucionar este problema se ha desarrollado los marcadores sociales. Una innovadora forma de almacenar, clasificar y compartir elementos de interés. Algunos de estos ejemplos son Digg y Delicious.

Esta gran familia de aplicaciones forma una parte muy importante de los métodos de aprendizaje que se usan en la educación. Herramientas como las computadoras/*laptops* con acceso a internet son muy propios para permitir que los escolares estén a la vanguardia y puedan acceder a millones de sitios para conseguir información que los ayude a complementar los conocimientos que adquieren en la escuela.

El uso y aplicaciones de cada una de estas, está delimitado por las necesidades y características de cada usuario. Cada docente debe verificar cuáles son las aplicaciones que se adaptan a su forma de trabajar y a las particularidades de su alumnado, para delimitar las que pueden ser más efectivas y de utilidad para su clase. Es importante que los estudiantes cuenten con determinadas competencias que les permitan el trabajo autónomo y la construcción de sus conocimientos. Saber buscar, seleccionar y procesar la información, expresarse y comunicarse en el ciberespacio, conocer sus riesgos, trabajar en equipo, tener capacidad crítica, imaginación y creatividad son algunas de las habilidades que deben adquirir los estudiantes. Por su parte, los docentes, además de contar con una actitud favorable hacia la integración de las TIC, tienen que tener también las suficientes competencias digitales y didácticas que les ayuden a contextualizar los contenidos curriculares en este entorno tecnológico (Zappalá, 2003).

1.4 El papel del estudiante y el profesor dentro de la clase

“La única pedagogía posible es estimular la curiosidad del educando” (Tierno, 2004).

Dentro del contexto constructivista se habla sobre el verdadero aprendizaje, donde los estudiantes se interesen y se multiplica la motivación hacia la adquisición del conocimiento ¿En qué consiste este verdadero aprendizaje? ¿Qué debe hacer el docente para que el alumno aprenda y se interese? La respuesta es simple, así como lo señala Tracey Tokuhama Espinosa: “El alumno debe ser el protagonista de las clases, no el maestro” (Espinosa, 2012).

Actualmente, captar la atención del alumno se ha convertido en una tarea desafiante para todos los docentes, sin embargo, esto tiene una solución funcional y práctica, que consiste en dar el protagonismo al estudiante dentro de la conocida tríada didáctica, compuesta por el alumno, el docente y el contenido.

Esta mirada constructivista obliga de alguna manera al docente a abandonar su protagonismo excluyente, adoptando el rol de guía, haciendo del estudiante un actor participativo y no sólo un receptor de información. Es importante además que el docente conozca a sus alumnos ya que difieren sus habilidades académicas, sus medios, intereses y motivación. Asimismo, provienen de culturas diferentes, incluyendo valores, actitudes y tradiciones variadas; puntos que influyen en gran parte en el aprendizaje (Nancy, 2011).

El objetivo del docente es facilitar la adquisición de conocimientos, comprensión, raciocinio y habilidades en el alumno, mediante actitudes positivas que influyen directamente en ellos, estás pueden ser: alentador, tolerante, activador, compendiador, orientador, flexible y responsable, sin dejar de lado la autoridad que inspire confianza y respeto por parte del alumno; estas actitudes pueden llegar a ser eficaces en diferentes situaciones. Parte de estas, reside en la comprensión de sus propias habilidades, fortalezas, experiencias y en la actualización de conocimientos continuos (Zappalá, 2003).

1.5 Aulas tecnológicas

El uso de recursos tecnológicos dentro de un salón de clase se ha dado como respuesta a la demanda de los alumnos de las nuevas generaciones. A partir de la década de los 80 se ha dado un aumento en el uso de estos recursos tecnológicos en países como Estados Unidos y otros del primer mundo. Los materiales audiovisuales y la computadora junto con *software* especializado para diferentes materias y el Internet, se han convertido en una herramienta muy útil para la impartición de clases (Martínez, 2000).

La composición de las aulas se diseña sobre modelos educativos y didácticos. Los aspectos arquitectónicos, ambientales, de acabado o mobiliario conforman el equipamiento físico. Se puede incluir en esta propuesta las computadoras o software compatibles y la conectividad adecuada que garantice la integración del equipamiento. Las aplicaciones forman parte esencial para el desarrollo colaborativo de los contenidos e intercambios.

Otros equipamientos son el pizarrón interactivo, amplificadores de audio, reproductores de DVD, el proyector conectado a la PC y al reproductor de DVD, así como la conectividad a internet, por vía inalámbrica o conmutada. Es bueno considerar los mandos a distancia o controles remotos de todos los equipos, *"Accespoint"* o puntos de acceso que permitan suministrar a cualquier equipo portátil la conectividad requerida, así como insumos de plumas interactivas como complemento del pizarrón, pizarrones de escritura que apoyen las proyecciones, entre otros aditamentos que permiten la conectividad e intercambio.

Las aulas tecnológicas son una solución educativa concebida para transformar la enseñanza y el aprendizaje. Suelen estar estructuradas por dispositivos para cada estudiante, una pizarra interactiva, un centro de control y monitoreo para el profesor y un *software* que permita la interacción entre todos los dispositivos mencionados. Favorece el desarrollo de capacidades y participación de los estudiantes (Perales, 2014).

A continuación, se exponen algunas de las ventajas y desventajas de esta propuesta educativa, que combina tecnología con elementos tradicionales de enseñanza (Martínez, 2000):

Ventajas

- Uso más amplio e intensivo de las TIC.
- Planificación y organización del aprendizaje.
- Forma telemática de llevar a cabo la interacción social.
- Desarrollo de las actividades de aprendizaje más centrado en el estudiante.

Desventajas

- El precio de la implementación de esta tecnología es alto.
- La motivación del alumno puede ser complicada.
- Se reducen el tipo de relaciones sociales que se establecen en las aulas tradicionales.

1.5.1 La función del profesor en este nuevo entorno tecnológico

El profesor en este concepto debe olvidarse del protagonismo. Tiene que adquirir plena conciencia de que su función tiene ahora más de tutorial que de docente. Su función no

consiste tanto en "dar clase", sino en "preparar la clase", para que se desarrollen correctamente las actividades. Preparar la clase es prever y disponer de estrategias adecuadas para introducir a los alumnos en cada tema y motivar eficazmente su aprendizaje. También es concebir las líneas esenciales de un proyecto flexible, que deberá ser luego completado con la participación de los propios alumnos en el planteamiento y formulación de preguntas. Los alumnos participarán en la búsqueda de posibles actividades y de medios eficaces para encontrar respuestas a las preguntas planteadas, con lo que les resultará más fácil personalizar el correspondiente proyecto; logrando así que sea su proyecto y no el del profesor. Lograr disponer del material documental diverso, y los instrumentos técnicos necesarios para la realización de los distintos tipos de actividades que deban desarrollarse en el trabajo de las correspondientes unidades didácticas (Ofelia, 2008).

Esta nueva modalidad vigente ya desde hace varios años, lucha por instaurarse en los actuales programas educativos de diferentes centros de enseñanza, aportando una fuente de conocimientos variados que se va más allá de la asimilación del contenido. Además, crea mecanismos que potencian el trabajo investigativo a través de Internet, garantiza un favorable manejo de las tecnologías y hace que los estudiantes sean capaces de gestionarse el conocimiento, gracias a los mecanismos tecnológicos. Conocer las ventajas, características y fundamentos esenciales de las aulas interactivas favorece un mayor manejo de las mismas y garantiza que el trabajo en el aula sea una cuarteta (estudiante, profesor, contenido y tecnología).

1.6 Análisis de soluciones existentes

En la actualidad son diversos los *softwares* desarrollados con el objetivo de brindar apoyo al proceso de enseñanza-aprendizaje, que garantizan la gestión de un aula con soporte para dispositivos que conforman el aula tecnológica, donde los estudiantes interactúan con el profesor mediante tabletas o computadoras. A continuación, se muestra un resumen del estudio realizado donde se exponen las principales características que presentan dichos sistemas:

Mythware

Mythware es una aplicación para la colaboración en el proceso de enseñanza y aprendizaje disponible en múltiples idiomas. Dispone tanto de conexión inalámbrica como cableada.

Sirve como plataforma de manejo para la enseñanza interactiva multimedia en los salones de clases con computadoras. El *software* garantiza la administración, control eficiente, la supervisión del estudiante y el trabajo colaborativo. Algunas de sus funcionalidades son: transmisión de pantalla, transmisión por cámara, chat, película en la red y comandos remotos (Nanjing Universal Networks, 2014).

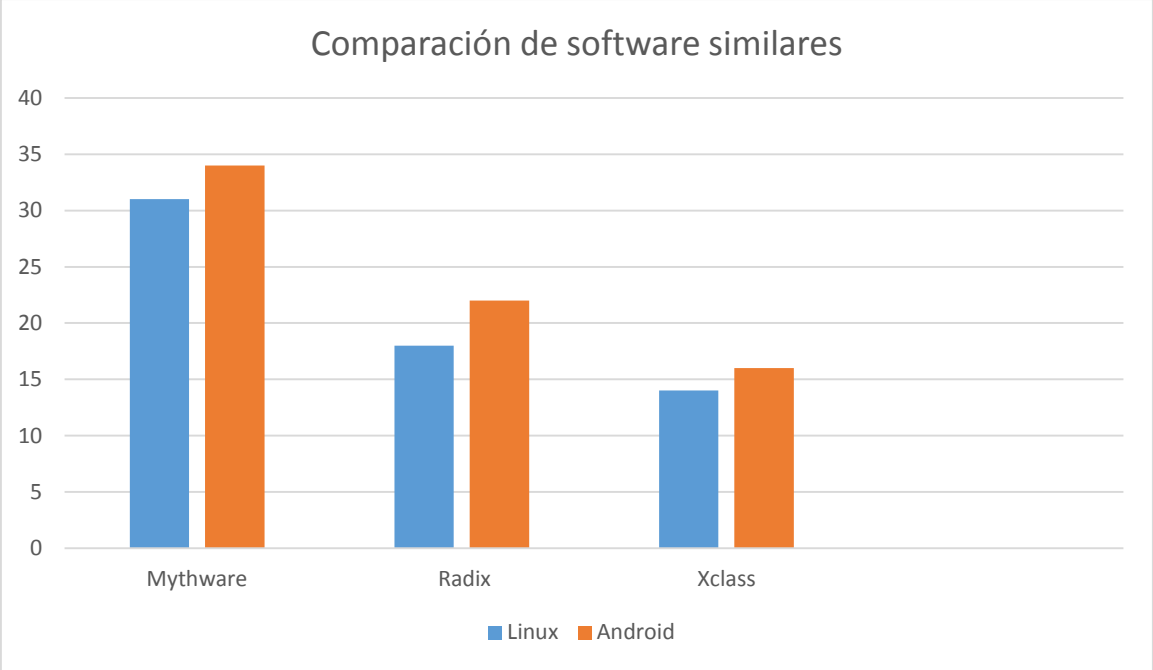
Radix

El *software* Radix, facilita el monitoreo y control de las actividades del estudiante por parte del profesor a través de la plataforma Android. Su funcionamiento se basa en una red WIFI del aula sin necesidad de conexión a Internet o servidor. Los estudiantes entran a la clase y trabajan desde su tableta asignada, mientras que el profesor tiene una computadora personal con el *software* de gestión. Los profesores pueden ver eventos en tiempo real basados en la actividad individual de cada estudiante. Algunas de sus funcionalidades son: monitoreo del estudiante, transmisión de voz del profesor, levantar mano del estudiante y mensajería de textos (Technologies Ltd, 2015).

XClass

XClass es una aplicación de gestión de aula multimedia para laboratorios de computación con un avance evolutivo. Conecta al profesor con los estudiantes en un aula en red y permite a los estudiantes comunicarse, aprender, colaborar entre sí y entre grupos. Además, actúa como un *software* de monitoreo de computadora que ayuda al maestro a supervisar las actividades de los estudiantes y mantener un buen orden en clase. Algunas de sus funcionalidades son: selector del aula, pizarra interactiva colaborativa, mensajes emergentes, colección de archivos y salvar lista de asistencia del aula (Sun Tech, 2016).

En Cuba no existe evidencia de la existencia de aulas tecnológicas, pero en la UCI, dentro del centro FORTES existe un proyecto llamado Desarrollo del producto ATcnea, que realiza la implementación de un *software* de gestión del aula, basándose en algunas de las funcionalidades clave que se extrajeron del estudio de las aplicaciones similares. A continuación, se muestra un gráfico donde se evidencia la cantidad de funcionalidades disponibles en las plataformas Linux y Android.



Se pudo constatar que los *softwares* analizados son privativos y en plataformas GNU/Linux y Android no disponen de todas sus funcionalidades implementadas.

Figura 1 Comparación de software similares

1.7 Ambiente de desarrollo

Actualmente existe una amplia variedad de tecnologías y herramientas que permiten la creación y el desarrollo de *software* encargados de la gestión del aula. Estas nuevas herramientas educativas vienen aparejado al desarrollo de las nuevas plataformas para móviles, tabletas y computadoras. Como resultado de la proliferación de estas plataformas estos *softwares* toman diferentes formatos en dependencia de la tecnología que se utilicen: Android, Java, C++ y .NET.

Cuando se realiza el desarrollo de aplicaciones de este tipo, se trata de desarrollar una misma versión de la aplicación para entornos tecnológicos diferentes, garantizando el desarrollo de sus funcionales acorde al sistema operativo que se utilice. La correcta selección del ambiente de desarrollo y las herramientas permite el adecuado funcionamiento de los sistemas de gestión de aulas.

A continuación, se plasma la selección de las tecnologías utilizadas para cumplir el objetivo de la investigación.

1.7.1 Java

Java es un lenguaje de programación de propósito general, concurrente y orientado a objetos que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible. Su intención es permitir que los desarrolladores de aplicaciones escriban el programa una vez y lo ejecuten en cualquier dispositivo, lo que quiere decir que el código que es ejecutado en una plataforma no tiene que ser recompilado para correr en otra. Es uno de los lenguajes de programación más populares en uso, particularmente para aplicaciones de cliente-servidor, con unos 10 millones de usuarios reportados. Es una plataforma libre bajo licencia GNU (Oracle, 2015).

Se seleccionó Java 8.0 ya que está definido en la especificación de los requisitos del proyecto Desarrollo del producto ATcnea. La aplicación del profesor utiliza Kryonet para la comunicación la cual esta implementada en Java, por lo que la aplicación del estudiante utiliza dicha librería, haciendo uso de la plataforma Java, para evitar problemas de compatibilidad.

1.7.2 Hibernate

Es una herramienta de mapeo de objeto relacional (ORM) para la plataforma Java[2] que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML[3]) o anotaciones en los beans de las entidades que permiten establecer estas relaciones. Hibernate es un *software* libre, distribuido bajo los términos de la licencia GNU LGPL (EducaciónIT, 2016).

Se utilizará Hibernate 5.2.2 ya que el ORM definido en los requisitos del proyecto Desarrollo del producto ATcnea y garantiza el mapeo de entidades y relaciones para la plataforma Java.

1.7.3 JavaFX

JavaFX es una familia de productos y tecnologías de *Oracle Corporation*, para la creación de *Rich Internet Applications* (RIA), que no son más que aplicaciones web que tienen las características y capacidades de aplicaciones de escritorio, incluyendo aplicaciones

[2]Disponible también para .Net con el nombre de NHibernate.
[3] Lenguaje de enmarcado, orientado a los datos a diferencia de su homologo HTML, el cual centra su atención en la presentación de dichos datos.

multimedia interactivas. Las tecnologías incluidas bajo la denominación JavaFX son JavaFX Script y JavaFX Mobile, aunque hay más productos JavaFX planeados (Alegsa, 2017).

Se seleccionó el *framework* JavaFX 2.0 ya que está definido en los requisitos del proyecto Desarrollo del productor ATcnea y como su nombre lo indica porque pertenece a la familia de Java y garantiza las librerías de componentes visuales heredadas de la aplicación de gestión de la clase.

1.8 Herramientas de desarrollo

Las herramientas de desarrollo de *software* son una combinación de útiles que automatiza o soporta al menos una gran parte de las tareas (o fases) del desarrollo: análisis de requisitos, diseño de arquitectura, diseño detallado, codificación, pruebas de unidades, pruebas de integración y validación, gestión de configuración, mantenimiento, etc. Las herramientas deben estar bien integradas, pueden interoperar unas con otras y están formados por el conjunto de instrumentos (*hardware*, *software* y procedimientos) que facilitan o automatizan las actividades de desarrollo (Dart, 1987).

Las herramientas de desarrollo tienen como finalidad disminuir el tiempo en las distintas fases del *software*, además posibilitan una mejora de los resultados obtenidos y ofrece soluciones con mayor calidad a los clientes. A continuación, se presentan la selección de estas herramientas, teniendo en cuenta las necesidades del cliente y la experiencia del equipo de desarrollo en la utilización de las mismas.

1.8.1 Entorno de Desarrollo Integrado (IDE) NetBeans

NetBeans IDE es un entorno de desarrollo, una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java, pero puede servir para cualquier otro lenguaje de programación. Existe, además, un número importante de módulos para extender el NetBeans IDE, el cual es un producto libre y gratuito sin restricciones de uso (Oracle Corporation, 2015).

Este IDE dispone de: soporte para crear interfaces gráficas de forma visual, control de versiones, colaboración entre varias personas y resaltados de sintaxis. Además sus funcionalidades son ampliables mediante la instalación de packs (software.com.ar, 2016).

Se seleccionó NetBeans 8.0 debido a que pertenece a las herramientas gratuitas, libres de coste. Tiene gran poder de refactorización de código, se integra fácilmente con el *framework* JavaFX, librerías y permite desarrollar cualquier producto usando el lenguaje Java.

1.8.2 Visual Paradigm

Visual Paradigm para UML es una herramienta para desarrollo de aplicaciones utilizando modelado UML (Unified Modeling Language) ideal para ingenieros de *software*, analistas de sistemas y arquitectos de sistemas que están interesados en construcción de sistemas a gran escala y necesitan confiabilidad y estabilidad en el desarrollo orientado a objetos. Posee herramientas gráficas para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos, funciones del sistema y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y compuestos reciclados (software.com.ar, 2016).

Particularmente esta última ha sido seleccionada para el modelado, dado su capacidad para satisfacer todas las necesidades de la implementación y la experiencia acumulada por el equipo de desarrollo en el trabajo con ella. La versión a utilizar es la 8.0.

1.8.3 Scene Builder

La aplicación JavaFX Scene Builder permite construir la interfaz gráfica de una aplicación de escritorio Java de forma más sencilla. JavaFX Scene Builder genera archivos descriptores FXML que se cargan en la aplicación evitando la tediosa y no sencilla tarea de construir la interfaz gráfica mediante código.

Permite generar un archivo en formato FXML (declarativo en XML) que contiene la descripción de las ventanas. Este descriptor es similar a la forma de construir interfaces gráficas en la plataforma de Microsoft con los archivos XAML.

Con los archivos FXML que genera la aplicación JavaFX Scene Builder crear aplicaciones gráficas es mucho más sencillo y más fácilmente mantenible. Proporciona un editor que sigue el principio lo que ves es lo que obtienes (*WYSIWYG*[4]).Permite generar los archivos

[4] Se aplica a los procesadores de texto y otros editores de texto con formato (como los editores de HTML) que permiten escribir un documento viendo directamente el resultado final.

FXML que posteriormente se pueden utilizar en la aplicación Java de escritorio para crear la interfaz visual (Perez, 2015).

Se seleccionó esta herramienta para el trabajo con archivos FXML, debido a su fácil manejo. Permite mediante un entorno visual trabajar con los componentes nativos de Java FX y posibilita la visualización y colocación de los elementos dentro de una escena. Se utilizó la versión 2.0.

1.8.4 Librería KryoNet

KryoNet es simplemente una biblioteca de redes Java, que toma un conjunto de clases de redes, y las reduce a lo que se necesitará para producir un servidor y un cliente totalmente funcionales. KryoNet, similar a muchas otras bibliotecas de redes (Game, 2015).

KryoNet proporciona una API sencilla y limpia para una comunicación de red cliente / servidor TCP[5] y UDP[6] eficiente utilizando NIO. KryoNet utiliza la biblioteca de serialización de Kryo para transferir de forma automática y eficiente los gráficos de objetos a través de la red. KryoNet funciona tanto en sistemas operativos *desktop* como en Android. KryoNet es ideal para cualquier aplicación Cliente / servidor. KryoNet también puede ser útil para la comunicación entre procesos (Game, 2015).

Se seleccionó la librería Kriyonet 2.21 debido a su utilización en la aplicación encargada de gestionar la clase para poder establecer la comunicación entre las dos aplicaciones.

1.9 Metodologías para el desarrollo del software

Una metodología de desarrollo de *software* se refiere a un *framework*[7] que es usado para estructurar, planear y controlar el proceso de desarrollo en sistemas de información. Existe una amplia gama de metodologías que son empleadas como buenas prácticas de la ingeniería informática actual. Por citar algunas, se encuentra *Rapid Application Development* (RAD) desde 1991, Programación Orientada a Objetos (OOP) a lo largo de la década de los noventa, *Dynamic Systems Development Method* desarrollado en UK desde

[5] TCP: Protocolo de Control de Transmisión.
[6] UDP: Protocolo del nivel de transporte basado en el intercambio de datagramas.
[7] Framework: infraestructura digital, es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos de software concretos, con base a la cual otro proyecto de software puede ser más fácilmente organizado y desarrollada.

1995, Scrum (desarrollo) en la última parte de los noventa *Rational Unified Process* (RUP) desde 1999, *Extreme Programming* (XP) desde 1999, *Constructionist Design Methodology* (CDM) desde 2004 y *Agile Unified Process* (AUP) desde 2005 por Scott Ambler (Sanchez, 2004). Las metodologías de desarrollo de *software* más empleadas en la UCI son: RUP, Scrum, AUP y XP (Somerville, 2005).

1.9.1 AUP

Agile Unified Process (AUP) es un marco de desarrollo de *software* que se caracteriza por, tener un método iterativo que describe cómo desarrollar software de forma eficaz, utilizando técnicas probadas en la industria. Se caracteriza por estar centrado en la arquitectura y enfocado en el riesgo. Puede ser adaptado a organizaciones o proyectos específicos y no es más que una versión simplificada de RUP, la cual describe en una forma simple, fácil de entender y brinda un enfoque de desarrollo utilizando técnicas ágiles y conceptos de RUP. Cuenta con nueve roles durante su ciclo de vida como son: administrador de proyecto, ingeniero de procesos, desarrollador, administrador de base de datos, moderador ágil, administrador de la configuración, *stakeholder*[8] y administrador de las pruebas (Edeki, 2013).

1.9.2 Variación AUP para la UCI

El proyecto pertenece a uno de los centros productivos de la Universidad de Ciencias Informáticas ajustándose a los estándares y metodologías empleadas para el desarrollo de *software* en la universidad. Por esos motivos la metodología AUP antes descrita, sufre modificaciones para que se adapte al marco de trabajo de los centros productivos. En la fase de inicio se modifica el objetivo de la misma, se unifican las 3 fases de AUP en una sola y se agrega una fase de Cierre. El ciclo de vida de los proyectos de la UCI tiene 7 disciplinas también, pero a un nivel más atómico que el definido en AUP. En esta variante el modelo de negocio, los requisitos y el análisis se consideran disciplinas. AUP UCI cubre con el área de procesos que define CMMIDEV v1.3 para el nivel 2, serían CM (Gestión de la configuración), PP (Planeación de proyecto) y PMC (Monitoreo y control de proyecto). Los requisitos funcionales se encapsulan en historias de usuario, descripción de requisitos por procesos y casos de uso; algunos de estos son obligatorios independientemente del tipo de proyecto y otro serán opcionales en base a las particularidades del mismo. Se

[8] Stakeholder: cliente o proveedor de requisitos

seleccionó el escenario cuatro para encapsular los requisitos funcionales haciendo uso de las Historias de Usuario (HU) ya que no se modela el negocio. El cliente estará siempre acompañando al equipo de desarrollo para convenir los detalles de los requisitos y así poder implementarlos, probarlos y validarlos.

1.10 Conclusiones parciales

El desarrollo del capítulo posibilitó la elaboración del marco teórico de la investigación, para un mejor entendimiento de la aplicación que se desea desarrollar. El estudio realizado a las aplicaciones similares facilitó el conocimiento y recopilación de funcionalidades para implementar la propuesta de solución. El análisis realizado para conformar el ambiente de desarrollo permitió seleccionar como metodología de desarrollo de *software* AUP en su variante UCI. El empleo de herramientas y tecnologías garantizó el desarrollo de la aplicación.

CAPÍTULO II: PROPUESTA DE SOLUCIÓN

2.1 Introducción al capítulo

En este capítulo se realiza una descripción de la solución propuesta. Además, se presenta el diagrama del Modelo del Dominio con la correspondiente descripción de cada uno de sus elementos. Por otra parte, se especifican los requisitos funcionales y no funcionales que debe cumplir el sistema. Se aborda sobre los patrones de diseño y arquitectónicos para lograr una estructuración de la aplicación. También se describen y generan como artefactos pertenecientes a la metodología, las Historias de Usuarios, y se muestran los Diagramas de Clases de Diseño y Diagrama de Secuencia de Diseño. Se diseñó un Diagrama Entidad-Relación para la persistencia de los datos en la aplicación y se explicó una de sus tablas. Se elaboró un Diagrama de Despique donde se muestra la distribución de la arquitectura el sistema desde el punto de vista de despliegue.

2.2 Propuesta de solución

La solución consiste en desarrollar una aplicación para computadoras personales que permita la interacción de los estudiantes con la aplicación gestora de la clase, tiene como objetivo potenciar el aprendizaje y fomentar los conocimientos del estudiante durante el desarrollo de una clase en el aula. El sistema a implementar consiste en una aplicación donde el estudiante primeramente se registra, creándose un perfil, que permite modificaciones. Seguidamente se conecta mediante WIFI o red cableada a una clase creada por el profesor. El estudiante dentro de la clase puede captar la atención del profesor levantando la mano siendo esta solicitud denegada o no por el profesor. La propuesta posibilita visualizar el sumario de la clase, así como los objetivos de la misma. En el trascurso de la clase se reciben notificación para mantener constantemente informado al estudiante. El profesor comprueba el conocimiento de los estudiantes a través de exámenes que presentan un conjunto de ejercicios de diversos formatos, se dispone de un tiempo limitado para responderlos. La clase también está equipada con un chat para interactuar con todos los miembros de la clase.

2.3 Modelo del dominio

El modelo de dominio es utilizado por el analista como un medio para comprender el sistema mediante una representación esquemática de los conceptos y elementos de la vida real que

serán utilizados en el mismo. Se crean con el fin de representar los conceptos clave del dominio del problema, las propiedades más importantes y las relaciones entre los conceptos, facilitando una mejor comunicación entre desarrolladores y clientes al establecer un lenguaje común para el entendimiento (HansErik Eriksson, 2000).

2.3.1 Diagrama de clases del Modelo de dominio

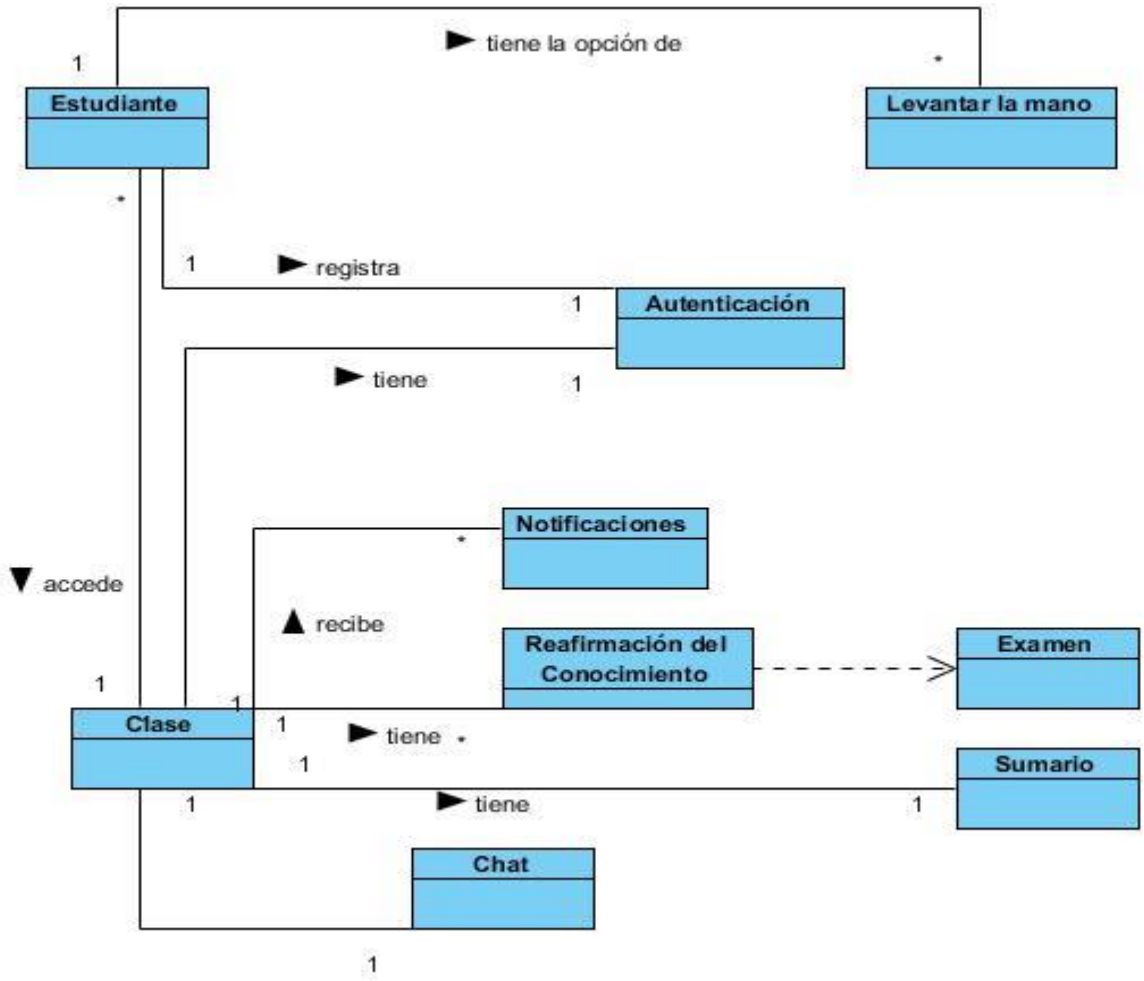


Figura 2 Modelo de dominio

2.3.2 Definición de las clases del Modelo de dominio

Estudiante: Indica a los usuarios de tipo estudiantes que interactúan con la aplicación del estudiante.

Chat: Permite a los estudiantes comunicarse dentro de la clase, para compartir criterios, aclarar una duda con el profesor, o sencillamente debatir sobre algún contenido de interés abordado en la clase.

Sumario: Permite a los estudiantes conocer el tema de la clase, así como los objetivos que se abordan en ella.

Levantar mano: Permite a los estudiantes captar la atención del profesor, solicitando la palabra sin tener que interrumpir la clase.

Clase: Permite al estudiante acceder a una clase para disponer de una serie de funcionalidades que engloban el desarrollo de la misma.

Examen o ejercicios: Permite por parte del profesor enviarles a los estudiantes exámenes o ejercicios con tiempo limitado.

Autenticación: Permite al estudiante autenticarse en la aplicación creándose un perfil y accediendo mediante su usuario y contraseña; las clases creadas por el profesor pueden requerir acceso mediante contraseña o mediante permiso.

2.4 Especificación de requisitos

El levantamiento de los requisitos constituye una de las etapas más importante en el proceso de ingeniería de requisitos. Para lograr el éxito durante el desarrollo de esta etapa, se realiza un trabajo en conjunto entre los participantes y los desarrolladores con el objetivo de identificar el problema, proponer elementos de solución, negociar diferentes enfoques y especificar un conjunto preliminar de requisitos para la solución (Pressman, 2005).

2.4.1 Requisitos funcionales

Los requisitos funcionales son declaraciones de los servicios que debe proporcionar el sistema, de la manera que este debe reaccionar a entradas particulares y cómo se debe comportar en situaciones particulares (Somerville, 2005).

Posteriormente se especifican los requisitos funcionales definidos para la siguiente investigación. Los mismos se encuentran divididos por paquetes para alcanzar un mejor entendimiento:

Paquete cuenta

1. **Iniciar sesión:** El sistema debe permitir a los estudiantes con los permisos necesarios iniciar sesión en el sistema, solicitando los siguientes datos:
    - Usuario
    - Contraseña

2. **Terminar sesión:** El sistema debe permitir al estudiante cerrar la sesión en el sistema a partir de la opción:
    - Cerrar sesión

3. **Crear una cuenta**: El sistema debe permitir al usuario crear una cuenta para acceder al sistema, solicitando los siguientes datos:
    - Imagen de la cuenta
    - Nombre
    - Apellidos
    - Usuario
    - Correo
    - Contraseña
    - Confirmar contraseña

4. **Editar una cuenta**: El sistema debe permitir al estudiante editar los datos de una cuenta creada con anterioridad, excepto el usuario porque constituye un identificador.
    - Imagen de la cuenta
    - Nombre
    - Apellidos
    - Correo
    - Contraseña
    - Contraseña nueva
    - Confirmar contraseña nueva

5. **Listar cuentas:** El sistema debe permitir mostrar en un listado con todas las cuentas de usuarios creadas en el sistema.

Paquete clase

6. **Buscar clase:** El sistema debe permitir buscar las clases creadas por el profesor mediante la opción:
    - Buscar clases

7. **Conectarse a la clase**: El sistema debe permitir al estudiante conectarse a la clase, de manera directa, mediante contraseña o esperando el permiso del profesor.

8. **Desconectarse de la clase:** El sistema debe permitir que el estudiante se desconecte de la clase de manera automática o mediante permiso.

Paquete sumario

9. **Visualizar sumario:** El sistema debe permitir al estudiante visualizar en su computadora el sumario y los objetivos de la clase.

Paquete cuestionario

10. **Visualizar examen:** El sistema debe permitir a los estudiantes visualizar las preguntas de diversas topologías que conforman el examen.

11. **Responder ejercicios del examen:** El sistema debe permitir a los estudiantes responder el examen enviado por el profesor. Una vez respondido el examen los estudiantes deben seleccionar la opción:
    - Enviar.

12. **Notificar evaluación:** El sistema debe permitir al estudiante visualizar mediante una notificación la evaluación obtenida en el examen.

Paquete chat

13. **Establecer conversación:** El sistema debe permitir a los estudiantes enviar y recibir mensajes enviados de los miembros de la clase.

Paquete mano

14. **Levantar la mano:** Permite al estudiante captar la atención del profesor pulsando la opción:
    - Levantar la mano.

15. **Aceptar mano levantada:** El sistema permite notificarle al estudiante que la petición mano levantada ha sido aceptada.
16. **Denegar mano levantada:** El sistema permite notificarle al estudiante que la petición mano levantada ha sido denegada.

Paquete configuración

17. **Configurar red:** El sistema debe permitir configurar los puertos para establecer la comunicación con la clase creada por el profesor.

2.4.2 Requisitos no funcionales

Los requisitos no funcionales, son aquellos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las restricciones de los servicios o funciones ofrecidas por el sistema. Normalmente solo se aplican a características o servicios individuales del sistema (Somerville,2005). A continuación, se muestran los requisitos no funcionales definidos para la investigación siguiendo sus criterios de clasificación:

Interfaz

1. Cumplir con las pautas de diseño establecidas por el proyecto ATcnea (Universidad de Ciencias Informáticas, 2013 - 2014).

Restricciones

2. Garantizar la conexión mediante red WIFI utilizando el estándar IEEE 802.11g con una velocidad de conexión de 54 Mbps.

3. Garantizar la conexión mediante red cableada utilizando el estándar IEEE 802.3 con velocidades de conexión de 10/100/1000 Mbps.

4. Garantizar la ejecución de la aplicación del estudiante en PC con sistemas operativos GNU/Linux.

5. Contar con la plataforma Java 8.0 y el *framework* JavaFX para el funcionamiento de la aplicación.

2.5 Historias de usuarios

Las Historias de Usuario (HU) son una forma de aproximar el lenguaje del usuario al desarrollador, de forma que se permita hablar un mismo lenguaje ambos y se eviten ambigüedades que puedan suponer pérdidas de tiempo notables (Llopis, 2016).

A continuación, se muestra la HU Iniciar sesión correspondiente al requisito funcional **Iniciar sesión** perteneciente al paquete **Cuenta**. Las restantes HU se muestran en el Anexo #1.

Tabla 1 HU: Iniciar sesión

Número: 1	**Nombre del requisito:** Iniciar sesión
Programador: Victor M Plasencia Costales	**Iteración Asignada:** 1
Prioridad: Alta	**Tiempo Estimado:**8 horas
Riesgo en Desarrollo: Falta de experiencia en el desarrollo de las herramientas y tecnologías.	**Tiempo Real:** 8 horas

Descripción:

1- Objetivo:

Permitir a los estudiantes iniciar sesión en el sistema.

2- Acciones para lograr el objetivo (precondiciones y datos):

- El usuario debe tener una cuenta creada en el sistema.

3- Comportamientos válidos y no válidos (flujo central y alternos):

El campo contraseña es obligatorio.

•**Contraseña:** Campo de texto que identifica la clave secreta de los usuarios para autenticarse en el sistema. Admite los caracteres alfanuméricos: guión bajo (_), arroba (@), punto (.), asteriscos (*), mayúsculas y minúsculas. El tamaño límite es de 5 a 7 caracteres. Ejemplo: ATcne4*C8

4- Flujo de la acción a realizar:

El sistema debe ser capaz de brindar a los usuarios la posibilidad de acceder al mismo especificando el dato siguiente:

- (*) Contraseña

Además, debe permitir seleccionar las siguientes opciones destacando el botón Iniciar sesión:

•Cancelar

•Aceptar

Al seleccionar la opción **cancelar** se cierra la interfaz iniciar sesión.

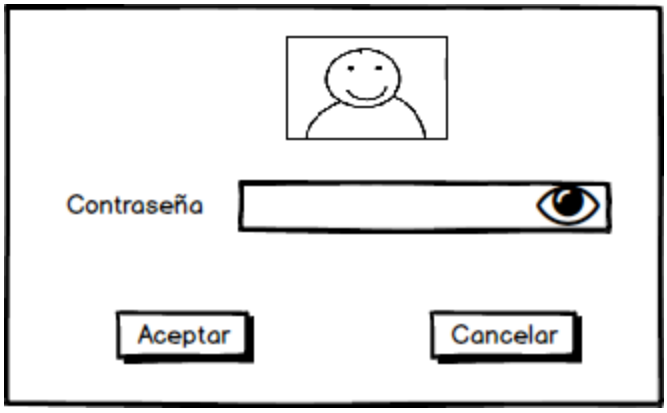
Al seleccionar la opción **acepta**r el sistema muestra un interfaz que permite seleccionar la clase deseada (Ver HU_SeleccionarClase).

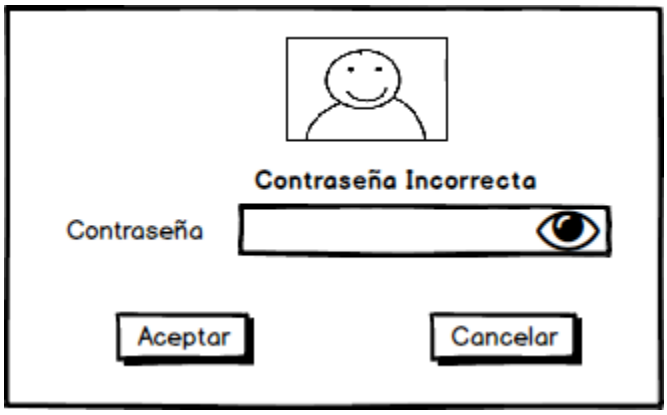
En caso que el usuario no introduzca la contraseña se le mostrará el siguiente mensaje de error: “Introduzca la contraseña”.

Si la contraseña es incorrecta, el sistema mostrará el siguiente mensaje de error: “Contraseña incorrecta”.

Observaciones:

Prototipo de Interfaz:





2.6 Patrón arquitectónico

Los patrones arquitectónicos o patrones de arquitectura ofrecen soluciones a problemas de arquitectura de *software* en ingeniería de *software*. Reflejan una descripción de los elementos y el tipo de relación que tienen seguido de un conjunto de restricciones sobre cómo pueden ser usados (Pressman, 2005).

2.6.1 Patrón Modelo-Vista-Controlador

Como base para el desarrollo de la aplicación propuesta se utilizó el marco de trabajo JavaFX que implementa el patrón arquitectónico Modelo-Vista-Controlador(MVC). Este estilo de arquitectura separa los datos de la aplicación, interfaz de usuario y lógica de control en tres componentes distintos:

Modelo: contiene una representación de los datos que maneja el sistema, su lógica de negocio y sus mecanismos de persistencia.

Vista o interfaz de usuario: compone la información que se envía al cliente y los mecanismos interacción con este.

Controlador: actúa como intermediario entre el Modelo y la Vista, gestionando el flujo de información entre ellos y las transformaciones para adaptar los datos a las necesidades de cada uno.

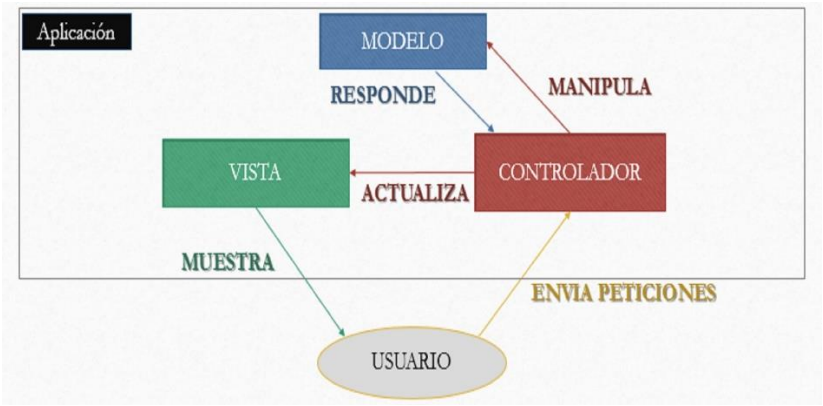


Figura 3 Modelo-Vista-Controlador (Milánes,2000)

En el *framework* JavaFX la vista es un archivo FXML que solo contiene la referencia del componente. Se comunica con su controladora a través de eventos ejecutados desde la

interfaz visual. La clase controladora tiene la responsabilidad de inicializar los elementos y acceder a los datos de los componentes. Por esta razón se puede considerar al subconjunto vista-controlador como una única unidad lógica.

La siguiente imagen muestra la composición del *framework*, donde el escenario es el elemento principal de la vista, seguidamente le sigue la escena. Un escenario puede estar contenido por disímiles escenas. Las escenas contienen los componentes visuales de la aplicación, dígase: *radiobutton*, *checkbox*, *textfield, button, pane*, *anchor pane*, etc. Conocer el ciclo de vida de estas aplicaciones que utilizan el modelo-vista-controlador garantiza su correcta implementación.

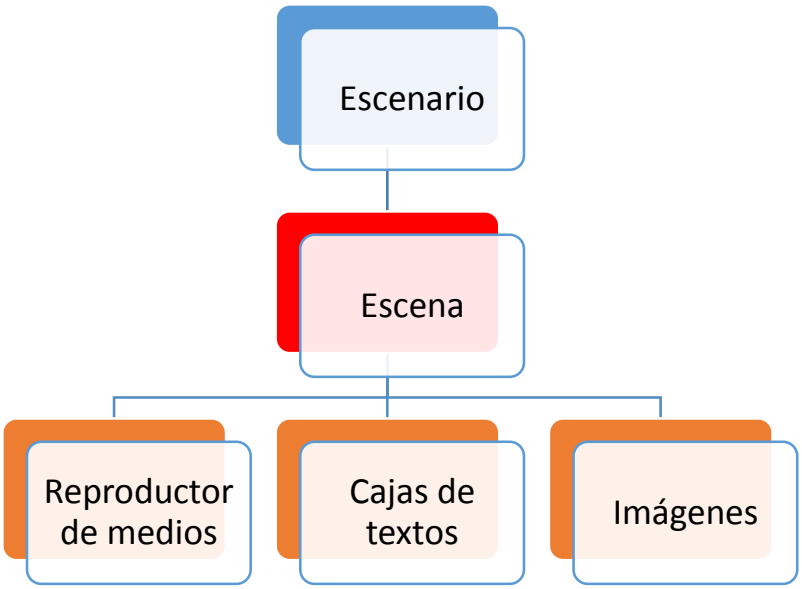


Figura 4 Estructura de JavaFX

2.7 Modelo de diseño

El modelo de diseño se crea para modelar el sistema convirtiéndose en un punto de partida para la implementación. El modelo de diseño ofrece ventajas en cuanto a la reutilización de código, resuelve problemas de efectividad para situaciones similares, y es aplicable a distintos problemas bajo circunstancias similares (Jacobson, 2000).

La propuesta de solución hace uso de los patrones de diseño, para lograr que la aplicación sea explicativa, fácil de mantener y conservar una cohesión en el código fuente. A continuación, se muestran las evidencias del uso de dichos patrones.

2.7.1 Patrón Singleton

Es un patrón estructural de diseño, perteneciente al grupo de patrones GoF[9], planteado para restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto. Su intención consiste en garantizar que una clase, sólo tenga una instancia y proporcionar un punto de acceso global a ella (Cunningham, 2012).

La clase *ATcneaSingleton.java* garantiza una instancia única del objeto creado en todo el ciclo de ejecución. En los Anexo # 8 se muestra un ejemplo claro de la utilización del patrón.

2.7.2 Patrones GRASP

GRASP es el acrónimo de General Responsibility Assignment Software Patterns, estos patrones ayudan a encontrar patrones de diseños concretos, son considerados una serie de buenas prácticas para el diseño del *software*. A continuación, se muestran una serie de patrones utilizados (Johnson, 1994).

Bajo acoplamiento: debe haber poca dependencia entre las clases, con la implementación de este patrón debe ser posible que cada clase sea constituida una sola unidad, sin hacer usos de métodos o atributos procedentes de otras clases (Johnson, 1994). En el paquete **sumario** las clases que lo componen solo comparten el atributo sumario, de esta forma se evidencia la poca dependencia entre las clases que conforman el paquete.

Alta cohesión: cada elemento del diseño debe realizar una labor única dentro del sistema. En la aplicación se hace uso de las *interfaces*, las cuales describen la lógica del comportamiento de los métodos. Estas interfaces garantizan diferentes implementaciones, donde cada clase que implementa dicha interface realice una única labor (Johnson, 1994). En el Anexo #8 se ejemplifica en las clases *LessonConnection*.java y *StudentNodeCmd.java* que implementa las interfaces *CommandInterface.java* y *SocketClient.java* que implementa la interfaz *ComunicationInterface.java.*

Controlador: es un patrón que sirve como intermediario entre una determinada interfaz y el algoritmo que la implementa, de tal forma que es la que recibe los datos del usuario y la

[9] Terminología dada a conocer por el libro *Design Patterns* escrito por el grupo Gang of Four (**GoF**) compuesto por Erich Gamma, Richard Helm, Ralph Johnson y John Vlisides, en el que se recogían 23 patrones de diseño comunes.

que los envía a las distintas clases según el método llamado (Johnson, 1994). En el Anexo #8, se muestra como la clase *LanguageNodeController.java* ejemplifica el uso del patrón controlador para lograr la internacionalización del lenguaje en toda la aplicación.

Polimorfismo: es un patrón que le permite a una misma clase tener diferentes comportamientos (Johnson, 1994). En la aplicación la clase *NewAccountScreenControler.java* tiene los compartimentos de añadir un nuevo usuario y editar un usuario. Este comportamiento está regido por el constructor de dicha clase, que en sí es el que decide qué comportamiento es el que se ejecuta. En el Anexo #8 se muestra un ejemplo del uso del patrón polimorfismo.

2.7.3 Diagrama de clase del diseño (DCD)

Un diagrama de clases de diseño (DCD) representa las especificaciones de las clases e interfaces en una aplicación. Estos diagramas contienen clases, asociaciones, atributos, interfaces con sus operadores, constantes, navegabilidad y dependencias. De forma general los DCD muestran las definiciones de las clases de la aplicación en lugar de los conceptos del mundo real (Larman, 2003).

A continuación, se muestra el DCD del requisito funcional Crear una cuenta, perteneciente al paquete de requisitos **Cuenta**. Los DCD correspondientes a los requisitos funcionales restantes se muestran en el Anexo# 2 donde se muestran las clases con los atributos y sus métodos.

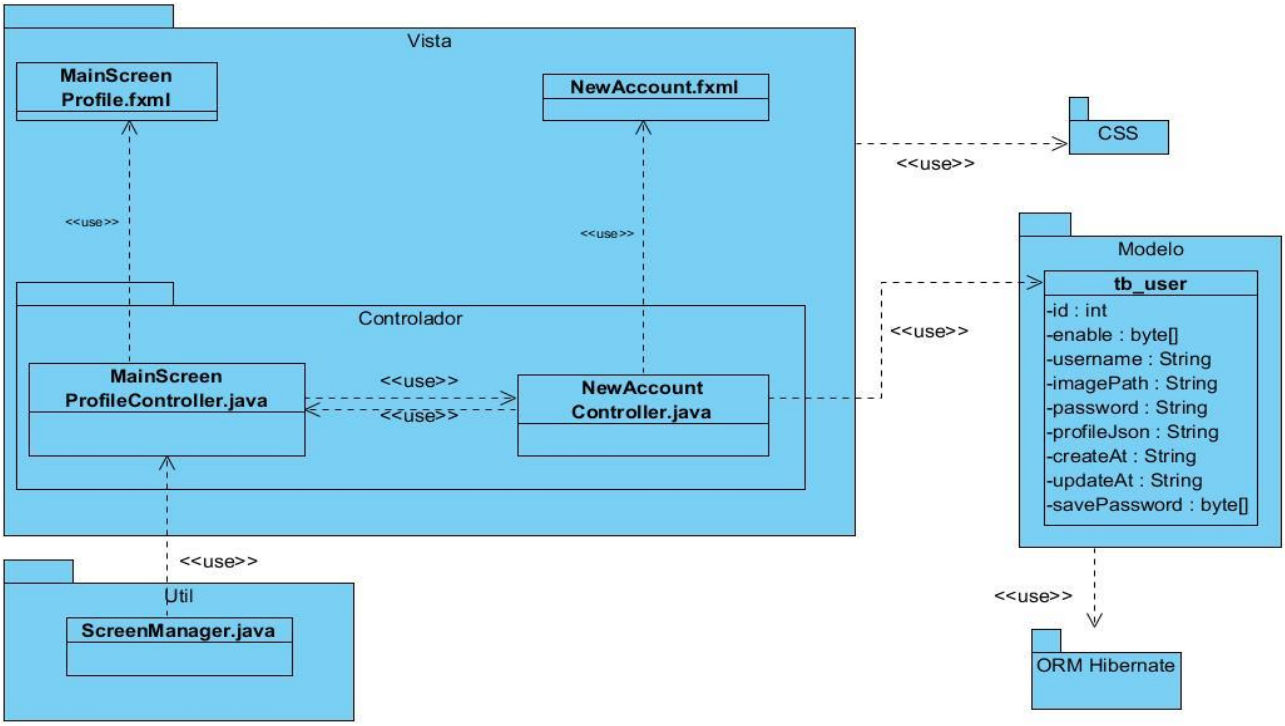


Figura 5 DCD: Crear una cuenta

2.7.4 Diagrama de secuencia (DS)

Los diagramas de secuencia de Diseño (DS) son una generalización de dos tipos de diagramas UML más especializados; se utilizan para mostrar las interacciones entre los mensajes. Ilustran las interacciones en un tipo de formato con aspecto de una valla, utilizan una notación simple y son fácil de entender (Larman, 2003).

A continuación, se muestra el DS del requisito funcional Crear una cuenta, perteneciente al paquete de requisitos **Cuenta**. Los DS correspondientes a los requisitos funcionales restantes se muestran en el Anexo# 3.

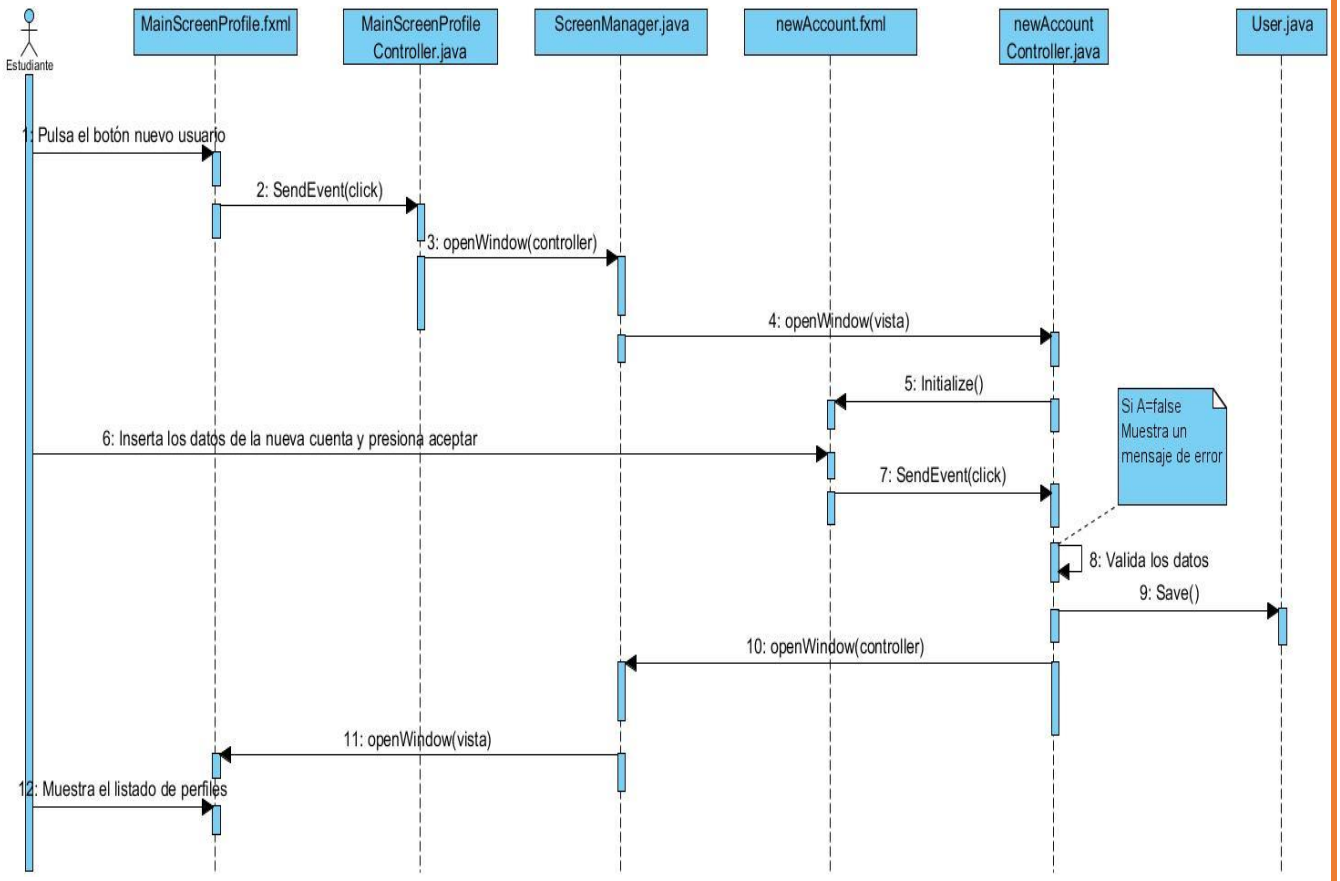


Figura 6 DS: Crear cuenta

2.8 Diseño de la Base de Datos

El modelo entidad-relación (ER) es un modelo de datos que permite representar cualquier abstracción, percepción y conocimiento en un sistema de información formado por un conjunto de objetos denominados entidades y relaciones, incorporando una representación visual conocida como diagrama entidad-relación (Ochando, 2014).

A continuación, se presenta el diagrama ER con el diseño de las clases y los atributos que persisten en la aplicación del estudiante, detallándose la relación entre entidades y el tipo de datos de sus atributos:

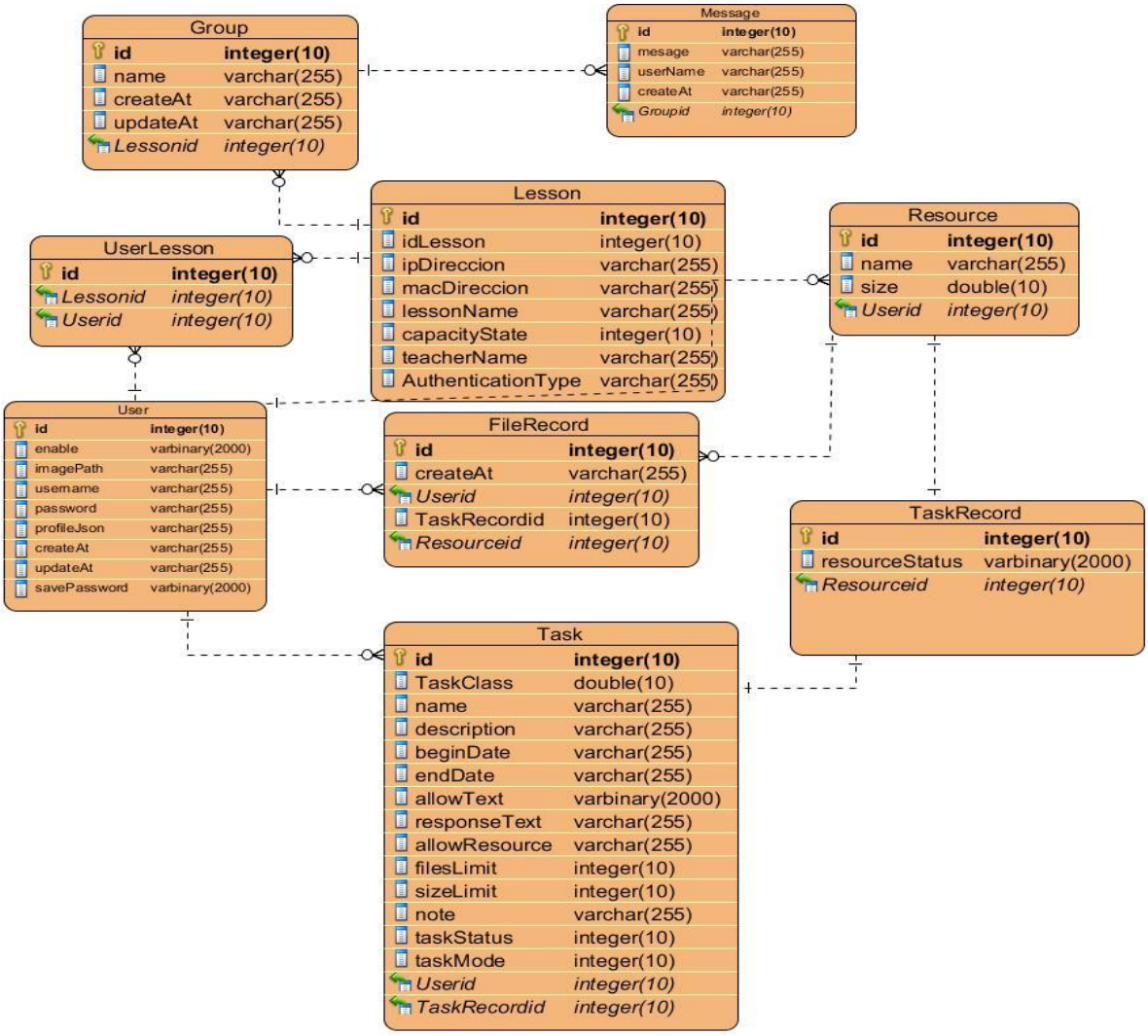


Figura 7 Diagrama Entidad-Relación para la aplicación del estudiante

2.8.1 Descripción de la Base de Datos

A continuación, se muestra la descripción de la tabla **tb_user** creada para almacenar los datos de los estudiantes que se registran en la aplicación. Las descripciones de las restantes tablas se muestran en el Anexo# 4.

Tabla 2 Descripción de la tabla usuario

tb_user		
Atributo	**Tipo**	**Descripción**
id	integer(10)	Campo que contiene el identificador de la tabla
enable	varbinary(255)	Campo que indica si está activo o no el usuario

imagePath	varchar(255)	Campo que almacena la ruta de la imagen
username	varchar(255)	Campo que almacena el nombre de usuario
password	varchar(255)	Campo que almacena la contraseña del usuario
profileJson	varchar(255)	Campo que almacena el nombre y los apellidos del usuario
createAt	varchar(255)	Campo que contiene la fecha de creación del usuario
updateAt	varchar(255)	Campo que contiene la fecha de actualización del usuario
savePasssword	varbinary	Campo que indica, si se salvó o no la contraseña

2.9 Modelo de despliegue

Los modelos de despliegue proveen la vista e implementación del sistema. Describen la topología del sistema, la estructura de los elementos de *hardware* y el *software* que ejecuta cada uno de ellos. Estos modelos incluyen la documentación de características técnicas requeridas, el tráfico de red y el tiempo de respuesta (Sarmiento, 2013).

2.9.1 Diagrama de despliegue (DD)

El Diagrama de despliegue es un diagrama estructurado que muestra la arquitectura del sistema desde el punto de vista de despliegue (distribución) de los artefactos del *software* en los destinos de despliegue (Sarmiento, 2013).

El DD que se presenta a continuación representa la distribución física de la propuesta de solución a través de nodos. Está compuesto por una PC para el estudiante y otra PC para el profesor. Ambas aplicaciones deberán tener instaladas Java 8.0 para ser ejecutadas en sistemas GNU/Linux, contar con infraestructura de red para lograr la comunicación mutua, mediante el uso de datagramas TCP y UDP. En la comunicación se utilizó un puerto para descubrir las clases de una subred, para recibir acciones del profesor y para enviarle acciones al profesor a través de datagramas UDP.

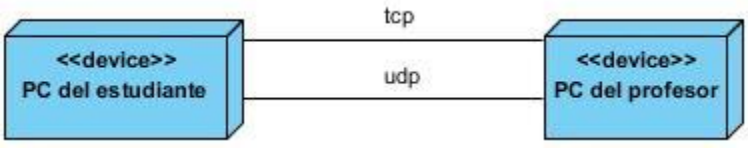


Figura 8 Diagrama de despliegue del software ATcnea

2.10 Conclusión parcial

El desarrollo del flujo del diseño perteneciente a la metodología seleccionada permitió la definición de la propuesta de solución y relacionar sus conceptos clave mediante el modelo de dominio. La identificación de los requisitos funcionales y no funcionales abrieron el camino para generar las historias de usuario. Los diagramas de clases de diseño y los diagramas de secuencia del diseño mostraron una proximidad al flujo básico de la propuesta descrita. La arquitectura del sistema concedió un soporte para la fase de implementación estructurando el desarrollo de la misma; y los patrones de diseño aportaron métodos eficientes de realizar las implementaciones. El diagrama de entidad-relación aseguró la persistencia de los datos en el *software*. El diagrama de despliegue facilitó conocer la infraestructura necesaria para la puesta en marcha del sistema.

CAPÍTULO III: IMPLEMENTACIÓN Y PRUEBAS

3.1 Introducción

En este capítulo se exponen las particularidades de la implementación de la propuesta de solución, con el objetivo de obtener un producto final que esté en correspondencia con los requisitos definidos por el cliente. Se generan clases, componentes u objetos ejecutables que se integran al sistema. Una vez realizadas las pruebas se obtienen los resultados, garantizando la calidad del sistema.

3.2 Modelo de implementación

El modelo de implementación es comprendido por un conjunto de componentes y subsistemas que constituyen la composición física de la implementación del sistema. Entre los componentes podemos encontrar datos, archivos, ejecutables, código fuente y los directorios. Fundamentalmente, se describe la relación que existe desde los paquetes y clases del modelo de diseño a subsistemas y componentes físicos (Somerville, 2005).

Para llevar a cabo la implementación de la solución, fue necesario la utilización de estándares de codificación, para garantizar la legibilidad del código. A continuación, se listan los estándares más significativos:

- Todos los ficheros fuente deben comenzar con un comentario en el que se lista el nombre de la clase, información de la versión, fecha, y *copyright*.
- Los comentarios de documentación (conocidos como "*doc comments"*) existen sólo en Java, y se limitan por /**...*/. Los comentarios de documentación se pueden exportar a ficheros HTML con la herramienta javadoc.
- Los programas pueden tener cuatro estilos de comentarios de implementación: de bloque, de una línea, de remolque, y de fin de línea. Poner las declaraciones solo al principio de los bloques (un bloque es cualquier código encerrado por llaves "{" y "}".). No esperar al primer uso para declararlas; puede confundir a programadores no preavisados y limitar la portabilidad del código dentro de su ámbito de visibilidad.
- Cada línea debe contener como mucho una sentencia.

- Las sentencias compuestas son sentencias que contienen listas de sentencias encerradas entre llaves " {sentencias}".
- Una sentencia *return* con un valor no debe usar paréntesis a menos que hagan el valor de retorno más obvio de alguna manera.

3.3 Diagrama de componente (DCOM)

Los diagramas de componentes muestran las interacciones y relaciones de los componentes de un modelo. Entendiéndose como componente a una clase de uso específico, que puede ser implementada desde un entorno de desarrollo, ya sea de código fuente, binario o ejecutable, dichos componentes poseen tipo, y están unidos mediante relaciones de dependencia (Wilder, 2009). A continuación, se presenta el diagrama de componentes que incluye los paquetes de requisitos **Cuenta** y **Clase.** Los restantes DCOM se encuentran en el Anexo# 5.

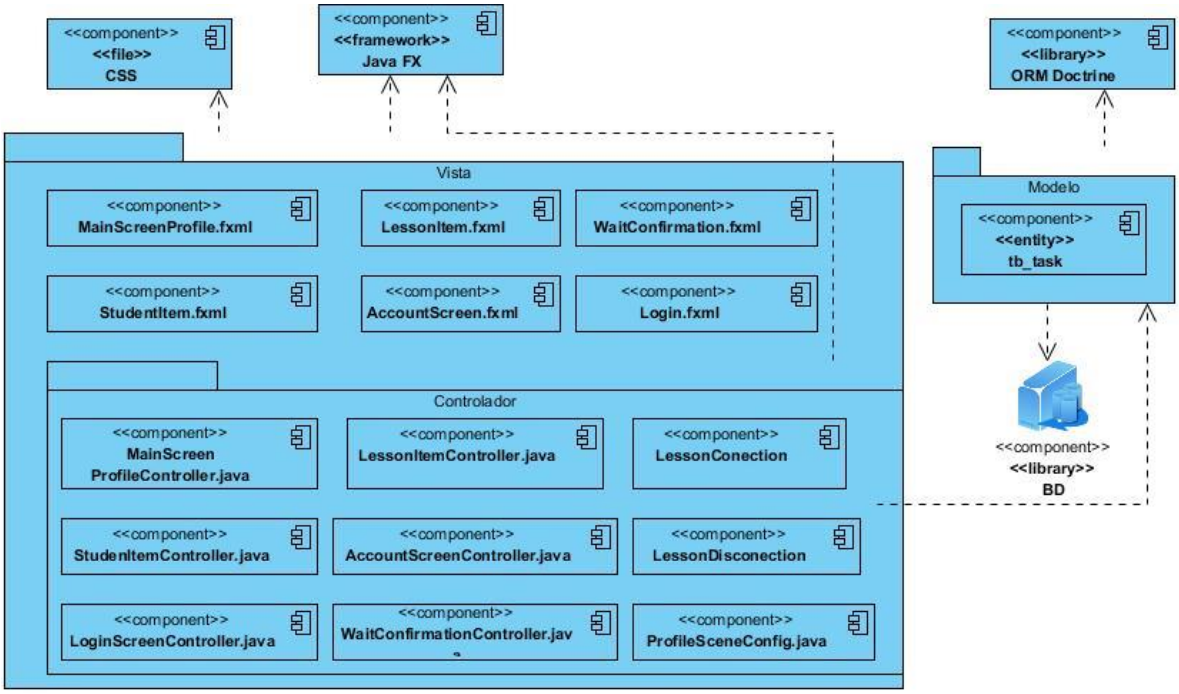


Figura 9 DCOM: Cuenta

3.4 Modelo de pruebas

Una estrategia para las pruebas de *software* proporciona una hoja de ruta que describe los pasos que se realizarán como parte de las pruebas, cuándo se planifican estos pasos y cuánto esfuerzo, tiempo y recursos se necesitarán. Por lo tanto, cualquier estrategia de

prueba debe incorporar la planificación de la prueba, el diseño del caso de prueba, la ejecución de las pruebas y la recopilación y evaluación de datos resultantes. Una estrategia de pruebas de *software* debe ser suficientemente flexible para promover el enfoque de pruebas personalizadas. Al mismo tiempo, debe ser lo suficientemente rígido como para fomentar una planificación razonable y un seguimiento de la gestión del proyecto (Pressman, 2010).

Según la metodología AUP, el objetivo de esta disciplina consiste en realizar una evaluación objetiva para garantizar la calidad. Esto incluye la búsqueda de defectos, validar que el sistema funciona tal como está establecido, verificando que se cumplan los requisitos.

Para llevar a cabo el proceso de pruebas existen cuatro niveles que pueden combinarse o reorganizarse en función de la naturaleza del proyecto o de la arquitectura del sistema: pruebas unitarias, pruebas de integración, pruebas de sistema y pruebas de aceptación. En los siguientes subepígrafes se describen las pruebas aplicadas al sistema propuesto.

Las técnicas de evaluación dinámica o prueba proporcionan distintos criterios para generar casos de prueba que provoquen fallos en los programas (Juristo, 2006). Estas técnicas se agrupan en:

- ✓ **Pruebas de caja blanca o estructural**: se basan en un minucioso análisis de los detalles procedimentales del código a evaluar, por lo que es necesario conocer la lógica del programa (Juristo, 2006). Su uso posibilita la obtención de casos de prueba que garantizan, al menos una vez, que sean ejecutados todos los caminos independientes de cada módulo. Posibilita ejercitar todas las decisiones lógicas en sus vertientes verdaderas y falsas. Permite, además, la ejecución de cada bucle con sus límites operacionales.
- ✓ **Pruebas de caja negra o funcional**: realizan pruebas sobre la interfaz del programa a probar, entendiendo por interfaz las entradas y salidas de dicho programa. No es necesario conocer la lógica del programa, únicamente la funcionalidad que debe realizar (Juristo, 2006).

Dentro de las técnicas de caja negra se incluyen técnicas de pruebas que serán descritas a continuación:

- ✓ **Partición de equivalencia** divide el dominio de entrada de un programa en un número finito de variables de equivalencia. Se definen dos tipos de variables de equivalencia, las válidas, que representan entradas válidas al programa, y las no válidas, que representan valores de entrada erróneos, aunque pueden existir valores no relevantes a los que no sea necesario proporcionar un valor real de dato.
- ✓ **Análisis de valores límites** prueba la habilidad del programa para manejar datos que se encuentran en los límites aceptables.
- ✓ **Grafos causa-efecto** permite validar complejos conjuntos de acciones y condiciones (Pressman, 2010).

Para el desarrollo de las pruebas se hace uso la técnica de partición por equivalencia.

3.4.1 Pruebas de sistema

Las pruebas de sistema buscan discrepancias entre el programa y sus objetivos o requisitos, enfocándose en los errores hechos durante la fase de implementación. El éxito de una prueba depende de descubrir un error no detectado hasta entonces; y su objetivo es diseñar casos de prueba que, sistemáticamente, saquen a la luz diferentes clases de errores, haciéndolo con la menor cantidad de tiempo y de esfuerzo (Pressman, 2010).

Las pruebas de sistema pueden incluir pruebas basadas en riesgos y/o en especificaciones de requisitos, procesos de negocio, casos de uso u otras descripciones de texto de alto nivel o modelos de comportamiento de sistema, interacciones con el sistema operativo y recursos del sistema.

Estas pruebas deben estudiar los requisitos funcionales y no funcionales del sistema y las características de calidad de los datos. Los probadores también deben enfrentarse a requisitos incompletos o no documentados. Las pruebas de sistema de los requisitos funcionales empiezan utilizando las técnicas basadas en la especificación (técnicas de caja negra) más apropiadas para el aspecto del sistema a probar (Müller, 2010).

Para la realización de las pruebas se hace uso de los diseños de casos de pruebas teniendo en cuenta el método de partición por equivalencia.

Diseño de casos de prueba

En los casos de prueba (CP) se incluyen la descripción de los principales escenarios, usuarios, posibles entradas, variables que intervienen en el proceso y flujo central donde se realiza el procedimiento. A continuación, se presenta el CP propuesto para la HU Iniciar sesión perteneciente al paquete **Cuenta**. Los restantes CP se encuentran en el Anexo # 6.

Descripción de las variables

Tabla 3 Descripción de las variables

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Contraseña	Campo de texto	No	Campo de texto que identifica la clave secreta de los usuarios para autenticarse en el sistema. Admite los caracteres alfanuméricos: guión bajo (_), arroba (@), punto (.), asteriscos (*), mayúsculas y minúsculas. El tamaño límite es de 6 a 16 caracteres. Campo de carácter obligatorio. Ejemplo: ATcne4*C8

Tabla 4 CP: Iniciar sesión

Descripción general: Permitir a los usuarios iniciar sesión en el sistema.				
Condiciones de ejecución: - Tener en cuenta la contraseña del usuario. -Tener creado un perfil en el sistema. -Solo de se debe crear un perfil en el sistema.				
Escenario: Iniciar sesión.				
Escenario	**Descripción**	**Contraseña**	**Respuesta del sistema**	**Flujo central**
EC 1.1 Seleccionar perfil	El usuario da clic en el perfil.	N/A	El sistema muestra un formulario donde le permite introducir la contraseña para acceder en el sistema. Permite además seleccionar una de las siguientes opciones: Aceptar y Cancelar	Seleccionar perfil/Iniciar sesión
EC 1.2 Opción Aceptar	Una vez especificado la contraseña el usuario selecciona la opción Aceptar	V	El sistema muestra la interfaz Mis clases.	Seleccionar perfil/Iniciar sesión/ Especificar dato/Aceptar
EC 1.3 Opción Cancelar	El usuario selecciona la opción Cancelar	N/A	El sistema muestra la lista de perfiles.	Seleccionar perfil/Iniciar sesión/Especificar dato/Cancelar
		I		

EC 1.5 Datos incorrectos	Al insertar la contraseña para iniciar sesión este es incorrecto.	V	El sistema debe notificar al usuario a través de un mensaje de información. Además permitirá realizar nuevamente la acción.	Seleccionar perfil/Iniciar sesión dato/Aceptar
EC 1.6 Datos vacíos	El usuario deja el campo vacío.	I	El sistema debe notificar al usuario a través de un mensaje de información. Además permitirá realizar nuevamente la acción.	Seleccionar perfil/Iniciar sesión /Aceptar

Resultado de las pruebas de sistema

Se aplicaron las pruebas del sistema realizando cuatro iteraciones, auxiliándose de los CP, donde se detectaron no conformidades. Las no conformidades identificadas fueron clasificadas en altas (errores en la interpretación de los procesos de la entidad y de funcionalidad), medias (errores de terminología y de diseño de *interface*) y bajas (errores de redacción, ortografía).

A continuación, el siguiente gráfico muestra la cantidad de no conformidades distribuidas en las cuatro iteraciones y agrupadas por clasificación. En la primera iteración se obtuvo 10 no conformidades, destacándose las no conformidades de nivel medio y alto. La segunda iteración mostró seis no conformidades con mayor predominio de no conformidades de nivel bajo. La tercera iteración mostró igual número de no conformidades bajas y medias.

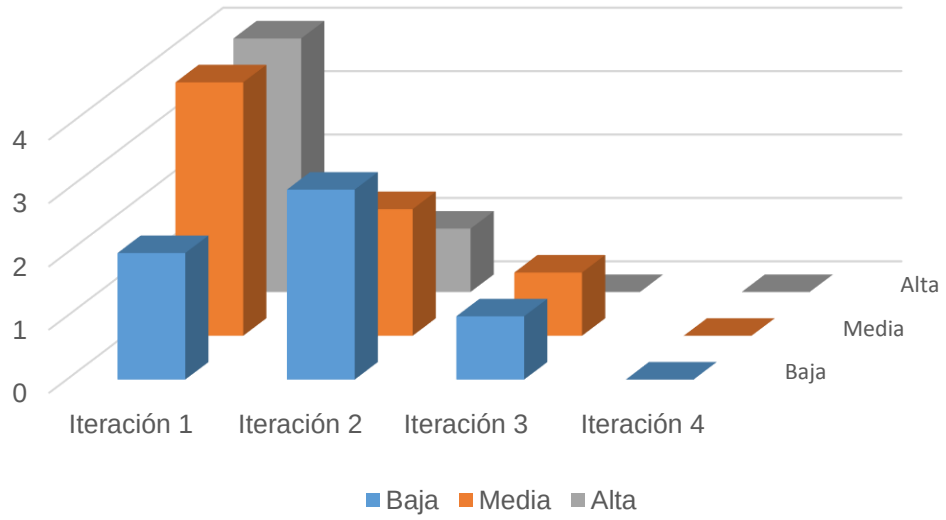


Figura 10 Resultados de las pruebas de sistema

A continuación, se muestra el escenario donde se realizaron dichas pruebas y una tabla que muestra las no conformidades detectadas en la primera iteración clasificadas en alta, media y baja. Las no conformidades restantes se encuentran en el Anexo# 7.

Tabla 5 Características del escenario de pruebas

	Microprocesador	RAM	Disco Duro	Sistema Operativo
PC del Estudiante	Core 2 2.6 GHz	4 GB	150 GB	Nova Escritorio 2015

Tabla 6 No conformidades en la primera iteración

No Conformidad	Clasificación
Cuando se elimina un perfil del estudiante, no se elimina toda la información en la base de datos relacionadas con él.	Media
El título de las ventanas no describe el contenido de la misma.	Media
Los mensajes de confirmación presentan faltas de ortografía.	Baja
Se pueden registrar en la base de datos dos clases con el mismo nombre.	Alta
Cuando se edita un perfil no muestra los cambios realizados.	Media
El estudiante puede editar el sumario de la clase.	Alta
En algunas interfaces, no permite el cambio de lenguaje de los textos.	Alta
Al cerrar la interfaz de la clase, no se notifica al profesor la desconexión de la clase.	Alta
La notificación de solicitud de conexión a la clase presenta errores de redacción.	Baja
Al cerrar la interfaz Cuestionario, se elimina la barra de título de la interfaz clase.	Media

3.4.2 Pruebas de aceptación

En las pruebas de aceptación interviene el cliente con el producto desarrollado, son consideradas como pruebas de caja negra y son tan importantes como las pruebas unitarias dado que significan la satisfacción del cliente. Estas pruebas son realizadas por el cliente, detectando no conformidades en cada iteración. Al concluir las iteraciones, si la aplicación cumple con la calidad requerida, el proyecto recibe por parte del cliente una carta de aceptación.

Resultado de las pruebas de aceptación

El cliente realizó las pruebas a la aplicación detectando no conformidades, a continuación, en la siguiente tabla se muestran las no conformidades encontradas. Las deficiencias detectadas fueron resueltas en dos iteraciones, donde en la última iteración la aplicación quedo excepta de errores.

No Conformidad	Iteración
No se puede acceder a la clase creada por el profesor una vez cancelada el acceso a la clase.	1
Cuando se acepta o deniega la solicitud de levantar mano, el mensaje mostrado por la aplicación es incorrecto.	
No se validan los datos del examen en el campo de insertar la respuesta.	
El examen no muestra un diseño correcto que permita delimitar los ejercicios de dicho examen.	
Al terminar el tiempo del examen, no se envía de manera automática	

Como resultado de las pruebas de aceptación, dada la satisfacción del cliente por el sistema se obtuvo la carta de aceptación.

3.5 Conclusiones parciales

En este capítulo fueron desarrolladas las disciplinas de implementación y pruebas, para lo que fue generado un diagrama de componentes que permitió entender la relación de dependencia y uso de la aplicación. Además, se aplicaron pruebas de sistema y de aceptación que permitieron determinar y erradicar las deficiencias encontradas, obteniendo finalmente una solución con un alto nivel de calidad.

CONCLUSIONES

Una vez concluida la presente investigación se puede comprobar el cumplimiento de los objetivos propuestos, arribando a las siguientes conclusiones:

- El análisis de los aspectos teóricos fundamentó la necesidad de desarrollar la aplicación de escritorio para estudiantes del *software* ATcnea utilizando la metodología, tecnologías y herramientas seleccionadas.
- Se desarrolló la versión de escritorio de la aplicación del estudiante del software ATcnea que permitirá extender su uso teniendo en cuenta las condiciones tecnológicas de los centros de educación cubanos.
- Se realizaron pruebas de software a la solución propuesta que permitieron identificar y corregir no conformidades, mejorando así la calidad del producto desarrollado.

RECOMENDACIONES

A partir de los resultados obtenidos con la investigación del autor se proponen las siguientes recomendaciones para futuros trabajos tomando como referencia el actual:

- Añadir como funcionalidad la autenticación de los estudiantes mediante los Sistemas de gestión de aprendizaje (LMS) y los Protocolos Ligeros/Simplificados de Acceso a Directorios (LDAP).
- Añadir la tipología de ensayo dentro del examen, que permite una respuesta de unas pocas frases o párrafos.
- Añadir la tipología de respuestas anidadas dentro del examen, que permite respuestas cortas o numéricas en espacios en blancos situados en un enunciado.

BIBLIOGRAFÍA

Technologies Ltd. Radix. 2015.

Alegsa. 2017. Alegsa.com. *Alegsa.com.* [Online] 2017. http://www.alegsa.com.ar/Dic/javafx.php.

Ander-Egg, E. *El taller: una alternativa de renovación pedagógica. Buenos Aires: Magisterio del Rio de la Plata.*

Avello Martínez, Raidell. (2007). *Exe: útil herramienta para la construcción de contenidos educativos.* s.l. : Quaderns Digitals, (2007). No. 24. ISSN 1575-9393.

Belloch, Ortí. 2014. Las tecnologías de la información y comunicación en el aprendizaje. Universidad de Valencia : s.n., 2014.

Benitez, Gerardo Meneses. 2007. *El proceso de enseñanza – aprendizaje: el acto didáctico.* 2007.

Bernazal, Fernando. Adictos al trabajo.com. [Online] https://www.adictosaltrabajo.com/.

Briano, Fernando. 2008. Sistema de control de versiones distribuido. *picando codigo.* [Online] julio 22, 2008. http://picandocodigo.net/2008/git-sistema-de-control-de-versiones-distribuido/.

Corporation, Oracle. 2011. NetBeans. [Online] 2011. [Cited: 01 15, 2014.] http://netbeans.org/index_es.html.

Dart, S.A. 1987. *Software Development Environments.* s.l. : IEEE Computer, 1987. Vol.20 No.11.

Díaz, Antonia Lozano. 2003. *El aula inteligente: ¿hacia un nuevo paradigma educativo?* s.l. : reseña del libro de Felipe Segovia Olmo, 2003.

Domínguez, Leal. 2012. *La tarea docente. Su contribución al aprendizaje de la historia de Cuba desde una perspectiva interdisciplinaria.* s.l. : Instituto Superior Pedagógico "José Martí"; 2007, 2012.

Edeki, Charles. 2013. *Agile Process Unified.* 2013.

EducaciónIT. 2016. EducaciónIT Blog. [Online] 2016. http://blog.educacionit.com/2013/02/07/que-es-java-hibernate/.

Eggen, P. y Kauchakl, D. 1999. *Estrategias docentes, enseñanza de contenidos curriculares y desarrollo de habilidades de pensamiento .* Buenos Aires : Fondo de Cultura de Argentina, 1999.

Escobar. Mariano Segura Escobar, Carmen Candioti López Pujato y Carlos Javier Medina Bravo Las TIC en la Educación: panorama internacional y situación española. 2007. 19 de noviembre : Centro Nacional de Información y Comunicación Educativa (CNICE), 2007.

Game, The Tech. 2015. The Tech Game. [Online] Marzo 25, 2015. [Cited: junio 15, 2017.] https://www.thetechgame.com/Archives/t=7114829/java-kryonet-networking-tutorial.html.

García, Melchor Gómez. *Estudio sobre aulas digitales para enseñanza presencial.* Universidad Autónoma de Madrid : s.n.

Garrido, MF. 2001. *Formación basada en las Tecnologías de la Información y las Comunicaciones .* 2001.

Gutiérrez, Dra. Ángeles. estudiante, Enfoques y Modelos Educativos Centrados en el. 2008. Mexico : s.n., 2008.

Johnson, Erich Gamma Richard Helm Ralph. 1994. *Desig Patterns.* 1994.

Juristo. 2006. Técnicas de evaluación de software. [Online] GrISE Grupo de Investigación de Ingienerìa de Software Empírica, 2006. [Cited: febrro 23, 2015.] http://www.grise.upm.es/sites/trs/1/pdf/Documentacion_Evaluacion_7.pdf..

Larman, Craig. 2003. UML y Patrones, 2da Edición . s.l. : Prentice Hall, 2003.

Martínez, Ing. Luis Ulices Romo. 2000. Uso de los recursos tecnológicos en las aulas inteligentes que dan los profesores dela preparatoria. s.l. : Tecnologico de Monterrey, 2000.

Martínez, R. A., yLorenzo, I. M. 2008. *El Software Libre en la educacion a distancia. Seleccion de Herramientas.* s.l. : Quaderns digitals , 2008. Revista de Nuevas Tecnologías y Sociedad, 7.

Mayrely Hernandez. educación, Importancia de las Herramientas digitales para la. 2014. 2014.

Müller, T. *Programa de estudio de nivel.* 2010.

Nancy, Carolina. 2011. El proceso de enseñanza aprendizaje con enfoque desarrollador. 2011.

Nanjing Universal Networks. 2014. *Mythware.* 2014.

Ochando. Prof. Dr. Manuel Blázquez Ochando, Fundamentos y Diseño de Base de Datos. 2014. 2014.

Ofelia, Dra. Ángeles Gutiérrez. 2008. Selección del documento “Enfoques y Modelos Educativos Centrados en el estudiante”. [book auth.] Universidad Autónoma Metropolitana Unidad Iztapalapa. Mexico : s.n., 2008.

Ojeda, Ramon. 2014. Granma. [Online] febrero 2, 2014. http://www.granma.cu/cuba/2014-02-02/destaca-informe-de-la-unesco-a-la-calidad-de-la-educacion-cubana.

Oracle Corporation. 2015. Netbeans. [Online] enero 12, 2015. https://netbeans.org/index_es.html.

Oracle. 2015. Java. [Online] Oracle, 2015.

Oré, Alexander. 2010. CalidadySotware.com. *CalidadySotware.com.* [Online] 2010. [Cited: mayo 29, 2017.] http://www.calidadysoftware.com/testing/pruebas_unitarias2.php.

P, VIVAS WHITE. 2009. *Informática: Cuerpo de Profesores de Enseñanza.* 2009.

Peña, Juan Manuel Fernández. 2004. Pruebas de Integración orientadas a objetos. 2004.

Perales, Maritza Del Carmen Rosario Milagros Sánchez. 2014. La mejora de los aprendizajes desde el “Aula . [book auth.] Laura Mercedes Campos Guevara. Puira : s.n., 2014.

Pérez, IC. 2016. Metodología de desarrollo. [Online] mayo 19, 2016. http://www.academia.edu.

Perez, Martin. 2015. Blog Bitix. *JavaFX Scene Builder, editor para crear archivos xml.* [Online] octubre 9, 2015. https://picodotdev.github.io/blog-bitix/2015/10/javafx-scene-builder-editor-para-crear-archivos-fxml/.

Pressman, Roger S. 2010. *Software Engineering: A Practitoner´s Approach. seventh.* New York : McGraw, 2010.

Ramírez, Ivan. 2011. Softonic. NetBeans IDE 7.0.1. [Online] Softonic, 08 09, 2011. [Cited: 01 15, 2014.] http://netbeans-ide.softonic.com/.

Reina, Martha Vazquez. 2011. Herramientas TIC para el aprendizaje. *EROSKI CONSUMER.* 2011.

Sampedro, A., Sariego, R., Martínez, Á., Martínez, R. A., Rodríguez, B. 2007. *Procesos implicados en el desarrollo de Materiales Didácticos reutilizables para el fomento de la Cultura.* s.l. : Científica y Tecnológica. Revista de Educación a Distancia, número monográfico II. Consultado (09/02/2007) en , 2007.

Sanchez, María A. Mendoza. 2004. Informatizate. [Online] 06 07, 2004. [Cited: 03 02, 2014.] http://informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html .

Sánchez, Tamara Rodríguez. 2014. *Metodología de desarrollo para la Actividad productiva de la UCI.* 2014.

Sarmiento. Despliegue, UML Diagrama de. 2013.

Sarmiento, Johna. 2013. UML: Diagrama de Despliegue. *UML: Diagrama de Despliegue.* [Online] 2013. http://umldiagramadespliegue.blogspot.com/.

Segovia Olmo, Felipe. 2003. *El aula inteligente.Nuevas Perspectivas.* Madrid : s.n., 2003.

software.com.ar. 2016. software.com.ar. *software.com.ar.* [Online] 2016. http://www.software.com.ar/p/intellij-idea.

Somerville, Ian. 2005. *Ingienerìa se software.* Madrid : s.n., 2005. 7ma Ediciòn.

Sun Tech. XClass. 2016.

Tena, Juan Ignacio Luca de. 2002. *Java y XML.* Madrid : Anaya, 2002.

Tierno, B. 2008. *“La única pedagogía posible es estimular la curiosidad del educando”.* En Zegovia, C : s.n., 2008.

Tokuhama Espinosa, T En Paci, J. 2012. *"El alumno debe ser el protagonista de las clases, no el maestro".* s.l. : La Nación, 2012.

Universidad de Ciencias Informáticas. 2013 - 2014. Estrategia Marcaria. *Estrategia Marcaria.* [Online] Universidad de Ciencias Informàticas, 2013 - 2014. http://iux.prod.uci.cu/.

Vázquez, Guadalupe. educación., Importancia de las herramientas digitales para la. 2013.

Vázquez, Guadalupe. 2014. Importancia de las herramientas digitales para la educación. 2014.

Visual Paradigm . [Online] [Cited: 03 24, 2014.] http://www.visual-paradigm.com/.

White, P Vivas. 2009. *Informática: Cuerpo de Profesores de Enseñanza.* 2009.

Wilder. 2009. blogger.com. *blogger.com.* [Online] Mayo 18, 2009. [Cited: abril 27, 2017.] http://diagramacomponente.blogspot.com/2009/05/definicion_18.html.

Zappalá, Daniel. 2003. *Serie Propuestas pedagógicas para el aula,Capítulo 1: Las TIC en el aula: estrategias didácticas, Capítulo 2: Propuestas pedagógicas por áreas de aprendizaje y Capítulo 3: Propuestas orientadas a la Formación Laboral.* 2003.