Universidad de las Ciencias Informáticas
Facultad 5

GECMA: Aplicación para la gestión y apoyo al control del Proceso de Enseñanza Aprendizaje de la Matemática Discreta 1

Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor: Yandry Gutierrez Rivero

Tutor: Ing. Yidian Y. Castellanos Sabarí

"Año 57 de la Revolución"
La Habana, Cuba
Junio 2015


“El hombre nunca sabe de lo que es capaz
hasta que lo intenta”
Charles Dickens

Declaración de autoría

Por este medio declaro ser autor de este trabajo y reconozco a la Universidad de las Ciencias Informáticas para que haga el uso que estime pertinente con este trabajo.

Para que así conste firman la presente a los ____ días del mes de _____ del año _____.

_____
Yandry Gutierrez Rivero
Autor

_____
Ing. Yidian Y. Castellanos Sabarí
Tutor

Datos de contacto

Autor: Yandry Gutierrez Rivero

Correo del Autor: ygrivero@estudiantes.uci.cu

Tutor: Ing. Yidian Y. Castellanos Sabarí

Correo del tutor: yycastellanos@uci.cu

Especialidad de graduación: Ingeniero en Ciencias Informáticas

Años de graduado: 4

Laboró durante 2 años en proyectos de conjunto con el Centro de Inmunología Molecular. Es autor de 3 artículos internacionales y 6 de referente nacional. Ha dirigido 4 tesis de pregrado con excelentes resultados. Imparte las asignaturas de Matemática Discreta en la Universidad de las Ciencias Informáticas.

Dedicatoria

Este trabajo de diploma, está dedicado a estas 6 mujeres especiales en mi vida:

Mi mamá: Por estar en todo momento a mi lado y tener siempre una solución para mis problemas.

Mi tía: Por ser para mí una madre más y sentir que me quiere como a un hijo más.

Mi abuela: Por ser fiel seguidora de mis estudios y preocuparse más que yo por mis resultados, dándome siempre el apoyo para asumir nuevas metas.

Mi hermana: Por simplemente ser mi hermana y poder compartir estos años con ella.

Por último y no menos importante, a Keisy y a Lianny por ser la alegría de la familia, las que con un solo gesto te hacen ver lo bonito de la vida.

Agradecimientos

Agradezco a todas aquellas personas que me han apoyado durante estos 5 años de universidad, especialmente:

A mis padres por ayudarme siempre en lo que se me ocurriera.

A mi tutor por disponer de sus conocimientos y tiempo en todo momento.

A mi abuela, mi hermana, mi tía y Pichy.

A Rosabel por hacerme sentir sencillamente feliz.

A los muchachos de mi aula, los de mi apartamento del 87 y a mis compañeros de la FEU.

A los que más que compañeros llegaron a convertirse en amigos: José Félix, Marlon, Roberto, Orson Marin, Erduin, Kiki, Esmaykel y Mirnerys.

Resumen

En la Universidad de las Ciencias Informáticas en el curso 2014-2015 se imparte la asignatura Matemática Discreta 1. Esta asignatura constituye piedra angular en el proceso de formación de ingenieros, sin embargo, los resultados docentes que muestran los alumnos al ser evaluada la misma, evidencian el grado de dificultad con que estos se enfrentan a ella. El profesor desconoce en qué momento el alumno disminuye la aprehensión del conocimiento y por tanto se torna difícil realizar una atención diferenciada en el momento y con los recursos adecuados. A partir de la problemática anterior se plantea la siguiente interrogante ¿Cómo contribuir al control docente en el Proceso de Enseñanza Aprendizaje de la Matemática Discreta 1? Para solventarla se trazó como objetivo: desarrollar una aplicación para apoyar el control del Proceso de Enseñanza Aprendizaje en la asignatura de Matemática Discreta 1, con la gestión del conocimiento como premisa fundamental. El proceso de desarrollo fue guiado por la metodología de desarrollo de software Programación Extrema y para su construcción se utilizaron diferentes herramientas, entre las que se encuentra el IDE Eclipse Luna, el servidor web Apache Tomcat y el gestor de bases de datos PostgreSQL. La implementación de la aplicación GECMA proporcionó una herramienta que evidenció la contribución realizada al control docente del Proceso de Enseñanza Aprendizaje en la Matemática Discreta 1, así como las pruebas realizadas a esta, determinaron la robustez y eficacia del sistema diseñado.

Palabras Claves: Control docente, gestión del conocimiento, Matemática Discreta.

Índice de contenido

Índice de tablas

Índice de figuras

Introducción

Actualmente los docentes utilizan la enseñanza presencial como medio para la gestión del conocimiento y como nexo para la interacción con los alumnos. Cada día aumenta la necesidad del espacio virtual y de una serie de recursos tecnológicos que han penetrado en el ámbito educativo con una fuerza arrolladora. Esta situación conlleva a los educadores a “desaprender para volver a aprender” y desarrollar una serie de competencias que, hasta hace unas décadas eran muy poco avizoradas por el maestro de tiza y pizarra que se dirigía a un grupo de estudiantes en un aula.

La actividad docente es la principal función que se desempeña en la Universidad de las Ciencias Informáticas (UCI), donde el docente es el principal responsable de la calidad de la educación. Los estudiantes tienen el compromiso de aprender a aprender, para ello el docente debe ayudar a desarrollar su potencial intelectual y creativo. El empleo de estrategias innovadoras, de acuerdo con las necesidades e intereses de los estudiantes para promover el aprendizaje significativo, potenciará la meta trazada hoy, de enseñar con un proceso más efectivo.

La Ingeniería en Ciencias Informáticas, carrera única que se imparte en la UCI tiene como objetivo: “el ciclo de vida de un software, con una perspectiva industrial, aplicado a los procesos de tratamiento y gestión de la información y el conocimiento en organizaciones productivas y de servicio” (UCI, 2002b, p.7). El Plan de Estudios de esta carrera tiene una duración de cinco años y la formación se basa en el principio martiano y de Fidel, de vincular el estudio con el trabajo. De esta forma, se estructura en dos ciclos muy bien definidos: un período de integración básico y otro de integración profesional, en los cuales se incluyen 13 disciplinas académicas con la Práctica Profesional como eje central del currículo.

Dicho Plan de Estudio contiene dentro de la disciplina de Matemática las asignaturas de Matemática Discreta (Matemática Discreta 1 (MD1) y Matemática Discreta 2). Al analizar el informe semestral de la universidad en la asignatura MD1 en el curso 2009-2010 (UCI, 2011), se identifican las siguientes deficiencias del claustro de profesores, que aún persisten en el tiempo:

- Existen problemas con la aplicación, en muchos casos aún incierta, de métodos de evaluación efectivos que permitan una retroalimentación apropiada en el proceso de aprendizaje y una comunicación efectiva entre los diferentes actores de la clase.
- Se pone de manifiesto una inadecuada planificación de las clases, donde el tiempo no se distribuye satisfactoriamente para lograr el total cumplimiento de los objetivos, esto también se encuentra afectado por el gran cúmulo de contenidos a impartir en las actividades y el grado de complejidad de los mismos.

La MD1 constituye piedra angular en el proceso de formación de ingenieros informáticos en no pocas universidades del mundo, sin embargo, los resultados docentes que muestran los alumnos al ser evaluada la misma, evidencian el grado de dificultad con que estos se enfrentan a dicha asignatura. En la actualidad los profesores de la universidad objeto de estudio se apoyan básicamente en el registro de asistencia, el Sistema de Gestión Universitaria así como en las evaluaciones frecuentes y parciales para registrar el control sobre la aptitud y las habilidades que van alcanzando cada uno de sus discípulos. De esta manera se convierten dichas herramientas en elementos de vital importancia (cada una por su lado) para comprender la situación de los discentes, sin lograr obtener en el momento adecuado una alerta sobre las causas o consecuencias adversas para vencer los objetivos del año.

Los desfavorables o plausibles resultados que se obtengan en la asignatura pueden estar influenciados por la asistencia, pero esta no es la única causa que incide sobre los resultados, de ahí que el autor asuma que el registro de asistencia y de evaluaciones únicamente no proveen todos los datos que el profesor necesita controlar durante el Proceso de Enseñanza Aprendizaje (PEA) de la MD1. A su vez los resultados docentes que se alcancen están sujetos al modo de estudiar, la incidencia y respuesta de los alumnos al medio social que les rodea, las motivaciones individuales y las que son generadas por la acción del guía y controlador del proceso; solo por citar algunas de las que más afectan hoy la aprehensión del conocimiento. Basta una observación científica adecuada para percibir que los elementos que se han mencionado, en el curso escolar 2014-2015, aun no son controlados de forma eficiente y sinérgica por los docentes. Los resultados que alcanzan los alumnos en los diferentes tipos de evaluación hoy no son registrados en herramientas que permitan la perdurabilidad de estos en cuanto al tiempo, en su mayoría; vale exonerar los cortes evaluativos, las pruebas parciales, exámenes finales y extraordinarios, así como otros datos significativos según el poseedor del

registro. Esta situación trae consigo que muchos profesores en el actual curso académico (2014-2015) confeccionen herramientas a partir del Excel, donde pueden informatizar a un nivel muy primario los datos, y obtener información aún muy básica a partir de lo registrado. Al iniciar el curso, el control docente carece de las herramientas tradicionales, como lo constituye el registro, debido al tiempo en que se hace efectiva la matrícula, cuestión por la que aparecen varias alternativas para este fin.

Cuando el profesor debe entregar un análisis de lo sucedido en un corte evaluativo se evidencia la pobreza de la información, debido a que para esto solo cuenta con herramientas carentes de una descripción que abunde sobre dicha evaluación y sus posibles causas. Se pudo constatar que el plan remedial que el profesor ofrece al que desaprueba, llega tarde, por no haber detectado el/los objetivo(s) que no venció el discente, y generalmente llega a partir de la solicitud de la dirección del Departamento para ser archivado. Detectándose a tiempo las causas que incidan sobre el aprendizaje individual y colectivo, el profesor puede tomar medidas preventivas para lograr la aprensión de los conocimientos. De igual modo es necesario el control frecuente de las evaluaciones de los estudiantes, y más allá de esto poder analizar la situación docente de un grupo, ya sea en el período de un mes o un semestre.

Se desconoce en qué momento el alumno disminuye la aprehensión del conocimiento y por tanto se torna difícil realizar una atención diferenciada en el momento y con los recursos adecuados. Años tras años estas deficiencias se repiten con diferentes actores; sin elementos comparativos para poder estudiar referentes anteriores se comienza a fabricar soluciones, que en varias ocasiones resultan inadecuadas.

El educador debe seleccionar las estrategias a implementar en el proceso de mediación del aprendizaje y promover el desarrollo de habilidades y técnicas para el aprendizaje significativo. Estos conocimientos deben estar orientados a la solución de situaciones prácticas en lo académico y de los problemas cotidianos que se le presenten al aprendiz, es decir, el PEA ha de ser significativo para el estudiante.

A partir de la situación anterior se plantea el siguiente **problema científico:** ¿Cómo contribuir al control docente en el Proceso de Enseñanza Aprendizaje de la Matemática Discreta 1?

Lo antes expuesto permitió definir como **objeto de estudio:** El Proceso de Enseñanza Aprendizaje de la Matemática Discreta 1.

Para resolver el problema planteado se propone como **objetivo general** de la investigación: Desarrollar una aplicación para apoyar el control del Proceso de Enseñanza Aprendizaje de la Matemática Discreta 1, haciendo uso de la gestión del conocimiento como premisa fundamental.

Enmarcado en el **campo de acción** de la investigación: El control del Proceso de Enseñanza Aprendizaje de la Matemática Discreta 1.

Para dar cumplimiento al objetivo general se plantean las siguientes **tareas de investigación**:

- Realización del marco conceptual para precisar los principales conceptos que se emplean en la investigación.
- Elaboración del marco teórico de la investigación a partir del estado del arte existente sobre el tema.
- Selección y justificación de las tecnologías, metodologías y su uso en el desarrollo del software.
- Obtención de requisitos para el diseño e implementación de la aplicación.
- Implementación de una aplicación para apoyar el enriquecimiento del PEA en la asignatura MD 1.
- Realización de pruebas a la aplicación.

Con el cumplimiento de las tareas antes planteadas, se consideran como **posibles resultados**: La elaboración de un informe detallado con la base teórica-práctica sobre la cual se sustenta la solución propuesta, además de una aplicación que apoye el control del PEA en la asignatura de MD 1, tomando como premisa la gestión del conocimiento.

A continuación se detallan los **métodos de investigación científica** empleados en la investigación:

Métodos Teóricos:

Histórico-lógico: Este método se utilizó para realizar el estudio de las tendencias históricas y actuales en cuanto al PEA de la MD 1 con el fin de obtener toda la información relacionada con la gestión del conocimiento y mediante esta, contribuir a apoyar procesos educacionales.

Análisis y síntesis: Se empleó para el análisis del PEA de la asignatura MD 1, realizar una descomposición de este y estudiarlo minuciosamente hasta comprender su funcionamiento y su relación intrínseca, complementándose con la parte sintética.

Métodos Empíricos:

Pruebas al Sistema: Se realizaron diferentes pruebas de aceptación al sistema para determinar si se comportaba según los resultados esperados definidos por el cliente.

Búsqueda Bibliográfica: Se utilizó para la búsqueda de información referente a la MD1, en informes semestrales de la UCI y libros de diversos autores acerca de la asignatura así como la didáctica.

Entrevista: Se empleó para obtener la situación real del uso de aplicaciones similares a la propuesta, teniendo en cuenta los criterios de los profesores tanto de la Universidad de la Habana como de la UCI.

Revisión documental: Se utilizó para analizar los documentos publicados en el espacio creado en el Entorno Virtual de Aprendizaje (EVA) para la MD1 y determinar las características que debe tener la solución propuesta.

El trabajo de diploma está estructurado de la siguiente forma:

- Introducción.
- Tres Capítulos.
- Conclusiones.
- Bibliografía.
- Anexos.

Capítulo 1. Fundamentación Teórica: en este capítulo se define la base teórica de la presente investigación, haciendo énfasis fundamentalmente en la Gestión del Conocimiento y la asignatura Matemática Discreta 1.

Capítulo 2. Características y Diseño del sistema: en este capítulo se realiza una descripción detallada de la solución, donde se definen los requisitos funcionales y además se realizan actividades de análisis y diseño.

Capítulo 3. Implementación y Pruebas: en este capítulo se describe la implementación de la solución, se ejecutan los casos de pruebas y se corrigen las no conformidades encontradas para garantizar el buen funcionamiento de la solución propuesta.

Capítulo 1. Fundamentación teórica

1.1 Introducción

A continuación se detalla la base teórica de la investigación desarrollada, partiendo desde la Gestión del Conocimiento (GC) como premisa fundamental para el desarrollo de GECMA. Se definen las herramientas, espacios de trabajo, lenguajes de programación y metodología a utilizar con el fin de construir una aplicación informática que provea a los docentes de una herramienta que ayude al control del PEA de la MD 1.

1.2 Gestión del conocimiento como vía para mejorar los resultados docentes

En el ámbito educativo la GC no es algo nuevo. Las instituciones educativas son unas de las principales gestoras del conocimiento. “Sus principales agentes -profesores- son por definición trabajadores del conocimiento. Los sujetos del aprendizaje -alumnos- son personas en formación que se encuentran dedicadas a tiempo completo a la tarea noble de aprender, y de aprender a aprender, a lo largo de la vida y a procesar conocimiento” (Carneiro, 2009).

1.2.1 Arquitecturas y herramientas de la Gestión del Conocimiento

Una arquitectura de GC emplea distintas herramientas y servicios que configuran como resultado final una solución de GC completa. Estas herramientas que dan apoyo a la GC dentro de las empresas se diferencian en 3 grupos o conjuntos (BONTIS, N. 1999):

- Grupo 1 - Herramientas de trasmisión inmediata: Son herramientas que permiten transmitir el conocimiento explícito de forma fácil al conjunto de miembros de una misma empresa. Las Wikis son buen ejemplo de este tipo de herramientas o la propia Wikipedia. Estas pertenecerían a una arquitectura principal que podría estar en el grupo 2.

- Grupo 2 - Herramientas y servicios de gestión del conocimiento interno: Son aquellos componentes dentro de una arquitectura que gestionan, analizan, buscan y distribuyen información.

- Grupo 3 - Herramientas y servicios de gestión del conocimiento externo: Al igual que en el grupo 2 son componentes que gestionan, analizan, buscan y distribuyen, pero en este caso también hay que añadir que localizan y extraen, dado que su misión

principal es la localización y extracción de información relacionada con la empresa pero que está en el exterior de ésta (principalmente en Internet o en otros soportes más tradicionales de contenidos) y que por lo tanto en algunas ocasiones la empresa puede ser ajena a esta y no tener conocimiento de su existencia.

La aplicación a desarrollar se sustenta sobre la base del Grupo 2 debido a que la intencionalidad es que mediante dicha herramienta los docentes tengan en sus manos la posibilidad de gestionar el conocimiento de sus estudiantes, a través de la búsqueda y análisis de información educacional que estará en esta. Otras estrategias de gestión de conocimiento incluyen:

- Recompensas (para motivar el intercambio de conocimiento)
- Contar historias (como medio de transferir conocimiento tácito)
- Mapeo de Conocimiento
- Comunidades de Prácticas
- Directorio de Expertos (para ayudar al buscador de conocimientos a llegar a los expertos)
- Evaluación de acciones
- Transferencias de buenas prácticas
- Ferias de Conocimiento
- Gestión de Competencias
- Repositorios de Conocimiento
- Tecnologías Colaborativas
- Agentes de Conocimiento (algunos miembros organizacionales toman la responsabilidad por un "campo" específico y actúan como primera referencia sobre con quién hablar sobre un tema específico)

La solución informática propuesta combina varias de estas estrategias de gestión del conocimiento. Debido a que mediante herramientas colaborativas se logra analizar los repositorios de conocimiento para evaluar las acciones de cada estudiante y constatar el progreso de su aprehensión de conocimiento de los contenidos de la asignatura.

1.2.2 La Gestión del conocimiento en la educación

Según Soto y un grupo de autores que le acompañan (SOTO, 2006), los datos son mediciones objetivas, ya que consisten en números, palabras o imágenes que no necesariamente están organizadas. Al referirse entonces, a un conjunto de datos que han recibido un procesamiento y tienen un significado claro, definido y que están relacionados e interpretados, entonces se menciona la información. Por otro lado, plantean que el conocimiento es la integración de valores, experiencias e información, que implica generar acción con la información que proviene de los datos. Por tanto, el conocimiento se puede crear, producir, medir, administrar, gestionar, compartir, enseñar, transmitir y comunicar. Mediante el conocimiento se puede transformar la realidad circundante, se pueden conocer los sucesos, la esencia, los principios, las relaciones y las consecuencias, en general el conocimiento es lo que le ha permitido al ser humano superarse.

La gestión del conocimiento es un proceso sistémico, organizado, dinámico y continuo, encaminado a aumentar el proceso de aprendizaje de las personas que aprenden y se desarrollan socialmente. Tiene como base que sólo se aprende al cuestionarse lo que se conoce y se hace uso del conocimiento, apoyándose en la integración de conceptos, teorías, métodos, aspectos novedosos y desconocidos que despiertan interés (HERNÁNDEZ LUQUE, 2013).

Se centraliza en tres elementos, el primero dirigido al desarrollo y aprendizaje continuo de las personas que tiene en cuenta la medición de capital intelectual, el segundo a la gestión y almacenamiento de la información y el conocimiento y como último elemento la interrelación con las tecnologías (HERNÁNDEZ LUQUE, 2013).

Como se expresa en el objetivo general de la presente investigación, se asume la GC como premisa fundamental en la implementación de la solución propuesta. Los datos con los cuales se opera en esta herramienta lo constituyen las puntuaciones de las evaluaciones, los datos identificativos de cada alumno y la asistencia, entre otros. La información que se puede extraer de estos datos depende de la creatividad e inteligencia de cada uno de los actores que

interactúen con él. Por su parte el conocimiento será extraído de un conjunto de informaciones propia de los datos relacionados con cada uno de los estudiantes, sus evaluaciones y los remediales que a cada uno de ellos se le apliquen. La GC se hace evidente desde el momento en que se necesite, en cursos posteriores, ante situaciones similares emitir una respuesta adecuada. La compartimentación y socialización del conocimiento tendrán participación fundamental en momentos en que se ha de socializar el corte evaluativo u otras valoraciones referentes a la evolución de un educando. Dichos conocimientos generarán aprendizaje, desarrollo de competencias, creatividad, innovación y valor.

1.3 Papel de las Matemáticas Discretas en la formación profesional

El acelerado desarrollo tecnológico alcanzado en los últimos años en las áreas de la computación y telecomunicaciones ha propiciado sorprendentes avances en las matemáticas, y más específicamente en temas que se encuentran incluidos dentro de la Matemáticas Discretas. Estas asignaturas abarcan temas que proporcionan los conocimientos base, necesarios e indispensables para que los profesionales en formación puedan conocer y adentrarse en estas nuevas tecnologías, con un enfoque práctico, aplicado y computacional, además de un acentuado carácter formativo.

Connotación alcanza su estudio consciente por ser la asignatura que trata sobre el conocimiento y explicación de fenómenos discretos y procesos finitos, que servirán para aplicaciones posteriores dentro de la formación de los futuros ingenieros en informática. Les permite conocer cómo trabajan estas modernas herramientas denominadas computadoras y poder aprovechar su potencialidad. Lo que hace que esta asignatura se plantee como respuesta a una variada serie de problemas de la vida real:

- diseño de bloques
- flujo de redes
- diseño de circuitos
- asignaciones horarias o de tareas

Propicia al alumno posibilidades de búsqueda de modelos matemáticos adecuados para un sinnúmero de situaciones, lo que suele ser muy habitual en el desarrollo profesional. Entre los

principales aportes que brindan las Matemáticas Discretas a través de diversos métodos, se identifica el apoyo a la creación de sistemas de elevada complejidad y que, sin embargo, alcancen los parámetros de eficiencia y eficacia deseados. El uso de métodos no garantiza, a priori, la corrección del software, pero es una buena práctica que permite alcanzar mejores resultados en la construcción de sistemas complejos, al revelar inconsistencias y ambigüedades.

El carácter formativo de la asignatura se debe, no solo al que tienen las Matemáticas en general sino, en concreto, a que el lenguaje y las herramientas que estas usan son los habituales en otras de las asignaturas de la carrera como: Programación e Ingeniería de Software.

1.4 La Matemática Discreta 1 en la UCI

La MD se imparte durante el primer año de la carrera, incluye principalmente los siguientes temas: teoría de conjuntos, relaciones binarias, lógica, circuitos lógicos, técnicas de demostraciones, relaciones de recurrencia, combinatoria y teoría de grafos. Estos contenidos se dividen en dos asignaturas: MD1 y MD2.

La MD 1 es fundamental en el proceso de formación de ingenieros informáticos, sin embargo, los resultados docentes que muestran los alumnos de primer año de la UCI al ser evaluada la misma, evidencian el grado de dificultad con que estos se enfrentan a ella. En la actualidad los profesores de esta universidad se apoyan básicamente en el registro de asistencia y evaluaciones para gestionar estos indicadores, siendo los mismos de vital importancia para comprender la situación de los discentes ya que los malos resultados que se obtengan en la asignatura pueden estar influenciados por problemas de asistencia. Detectándose a tiempo, el profesor podría tomar medidas preventivas para ayudar al estudiante a lograr la aprensión de los conocimientos. De igual modo es necesario el control cotidiano de las evaluaciones de los estudiantes y más allá de eso poder tener una solución que te permita analizar la situación docente de un grupo completo. Debido a la no existencia de esta herramienta, se desconoce en qué momento el alumno disminuye la aprehensión del conocimiento y por tanto se torna difícil realizar una atención diferenciada en el momento y con los recursos adecuados. Años tras años estas deficiencias se repiten con diferentes actores y al no existir una herramienta que permita estudiar referentes anteriores, se comienza a fabricar soluciones, que en varias ocasiones resultan inadecuadas.

El claustro de la asignatura que trabajó durante el primer semestre del curso 2014-2015, está conformado por un total de 12 profesores, ubicados todos en la Facultad Introductoria de Ciencias Informáticas. Para impartir la asignatura se utilizaron principalmente los siguientes métodos activos: aprendizaje basado en problemas, representación activa del conocimiento mediante mapas conceptuales, técnicas de algoritmización y trabajo grupal. Se realizó un trabajo diferenciado con los estudiantes indicándole tareas diferenciadas y atendiendo a sus deficiencias de forma individual. Se implementó además un horario semanal de aclaración de dudas y consultas semanales con los profesores. Aun así los resultados de la asignatura no fueron muy alentadores, a continuación se enuncian algunas situaciones reales que se evidenciaron en la asignatura:

- Hubo un alto número de estudiantes no presentados al examen.
- El plan de la asignatura se cumplió en un 100%, impartiéndose todas las clases con la máxima calidad posible. Los profesores mostraron compromiso y responsabilidad con la tarea.
- Por parte de algunos estudiantes, se ha constatado falta de estudio consciente, organizado y sistemático, dado en gran medida por el desconocimiento de estrategias y estilos de aprendizajes.
- Falta de preparación para enfrentar las actividades prácticas.
- Exceso de actividades docentes durante la semana, lo cual les resta tiempo de auto preparación.
- En algunos casos, reiteradas ausencias a clases por parte de los estudiantes.
- Despreocupación por sus resultados docentes, conformándose con solo aprobar.
- Poca asistencia (aunque no en todos los casos) a las consultas planificadas. La mayor asistencia se logró en los días más cercanos al examen.
- Bajo nivel de comprensión de los conceptos.
- Falta de capacidad analítica, presentando un pensamiento memorístico.

Los resultados alcanzados por los estudiantes en las evaluaciones no han sido satisfactorios. Esto se evidencia en los resultados de las evaluaciones tanto parciales como las frecuentes con la excepción de los resultados de algunos grupos.

Se distingue del gráfico siguiente, sobre la composición de notas, que solo 13 estudiantes obtuvieron entre 4 y 5 puntos en el examen, y que el mayor por ciento del año está suspenso.

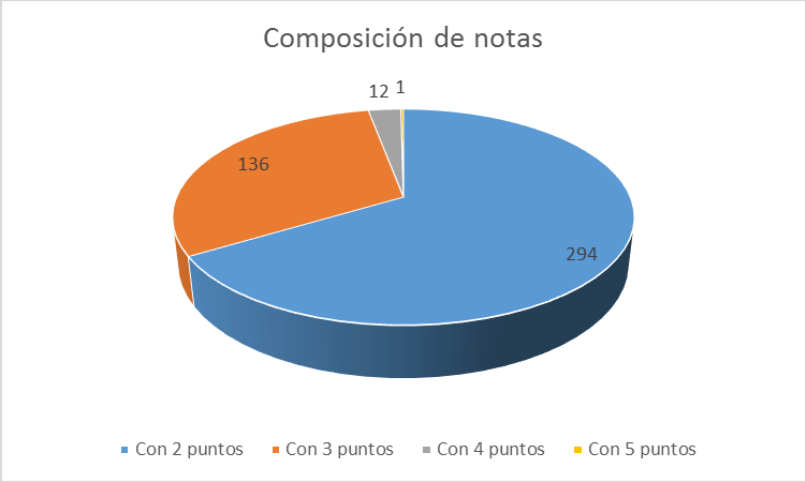

Figura 1. Resultados docentes MD1 curso 2014-2015.

Como principales agravantes a estos resultados fueron señaladas las siguientes:

- Los profesores deben estar pendientes a la atención de los estudiantes cuando se está revisando un ejercicio en la pizarra.
- Prestar atención al desarrollo de los ejercicios en pizarra por parte de los estudiantes para corregir los errores de notación y contenido cometidos.

Las recomendaciones fundamentales de los controles realizados fueron:

- Utilizar otros métodos de aprendizaje y se deben mostrar más ejemplos donde se vincule con otras asignaturas y donde se pueden aplicar los contenidos estudiados.
- Aplicar técnicas como trabajo en equipo para el desarrollo de aquellos ejercicios de mayor complejidad.

- Recordar en clase cuando se imparte un contenido, la existencia de otros modelos o conceptos similares, aunque estos no se utilicen o estudien en la asignatura.
- Realizar más acciones encaminadas a dar cumplimiento a las estrategias curriculares.
- Analizar otras posibles soluciones a los ejercicios, que sean diferentes a las aportadas por los estudiantes o por el profesor (COLECTIVO DE AUTORES, 2015).

1.5 Estudio de aplicaciones análogas

Para el estudio de aplicaciones análogas se confeccionó una entrevista, cuyo modelo se puede observar en los anexos del presente trabajo de diploma.

El Departamento Docente Metodológico (DDM) de la Universidad de La Habana (UH) tiene entre sus funciones dar seguimiento a tesis vinculadas con el PEA. La Msc. Rita Roldán, trabajadora de este, plantea que en el DDM se asesoran muchas tesis, ya sea de pregrado, maestrías o doctorados, destacando el apoyo del departamento a tesis de la UCI. En el diálogo desarrollado con Roldán, refiriéndose al conocimiento de aplicaciones existentes que solventen esta problemática, planteó su desconocimiento acerca de aplicaciones análogas así como que consideraba importante este apoyo para los educadores, principales responsables del PEA de sus educandos.

El Doctor Rey Guerrero Proenza quien se desempeña como subdirector del Centro de Estudios para el Perfeccionamiento de la Educación Superior (CEPES) responde a la interrogante sobre la existencia de aplicaciones análogas a GECMA que en sus años de experiencia nunca había visto una aplicación que hiciera las características descritas. Plantea además que existen muchas herramientas para potenciar la capacidad intelectual de los estudiantes, pero nunca enfocadas a mejorar las condiciones para que los educadores las pongan en práctica.

Según lo expresado por Ricardo Portuondo, Vicedecano Docente de la facultad de Matemática y Computación de la UH, en la facultad no se desarrolla ni se encuentra en proceso de desarrollo una aplicación similar a GECMA, además alega que en la UH -al igual que en la UCI- resulta compleja para los estudiantes la aprehensión de los temas que se imparten. En este periodo la facultad se encuentra en pruebas finales de Matemática Discreta y los resultados han sido poco alentadores, por lo que es necesario este tipo de aplicación que oriente al profesor

mediante informaciones reales de sus estudiantes, por el camino de la toma de decisiones que aumenten los conocimientos de los discentes, siendo el PEA significativo para estos.

En entrevista realizada a la Dra Liliam Domínguez, Directora del Centro de Tecnología Educacional de la UH, se constata la no existencia, en dicho centro, de una aplicación con las características de la solución informática que se propone. En esta casa de altos estudios se utiliza la herramienta Mi Moodle en Casa, para una mejor comprensión de esta se expone inicialmente Moodle:

Moodle: Esta es una aplicación web que permite la creación de cursos, brindando a los estudiantes modos amigables de comprobar sus conocimientos además de archivar los resultados de cada uno de ellos en una base de conocimiento. Solo se pueden almacenar las evaluaciones internas de la aplicación, cualquier otra información ya sea pruebas de diagnóstico, preguntas escritas, registro de asistencia a clases y otras no son almacenables en dicha aplicación debido a que no está concebida para esto. En la opinión de la doctora Liliam la aplicación a desarrollar para el control del PEA tendrá un valor incalculable porque hasta ahora no se cuenta con una herramienta así.

Mi Moodle en Casa: Es una aplicación creada por la UH con funcionalidades similares a las que brinda Moodle. Es una herramienta de escritorio, para facilitar el trabajo de los profesores que no tienen experiencia en el trabajo con la web. Periódicamente se puede exportar un archivo HTML con todos los datos de una asignatura y agregárselos a Moodle. De igual modo que Moodle no satisface la necesidad de poder constatar el avance diario de los estudiantes teniendo en cuenta sus evaluaciones en el aula , su comportamiento y asistencia , además de sus resultados en exámenes y poder compararlo con otros de diferentes etapas.

Entorno Virtual de Aprendizaje (EVA): El EVA es un Moodle personalizado donde se encuentran los cursos que se imparten en la UCI. Los profesores tiene la posibilidad de introducir toda la información necesaria para un curso ya sea bibliografía básica, bibliografía complementaria, conferencia, clases prácticas entre otros. Además, se pueden colocar en él, encuestas, crucigramas y otros, siendo estos evaluados automáticamente por la aplicación y el resultado se aloja en un registro de evaluaciones del propio entorno. Al igual que en los casos anteriores nos satisface la necesidad de registrar las calificaciones de los estudiantes en

actividades en espacios áulicos ya sea pruebas u otro tipo de actividad además que no permite llevar un control de la asistencia de cada asignatura.

Sistema de Gestión Universitaria (SGU): El SGU es una potente herramienta desarrollada en la UCI para llevar un control acerca de las evaluaciones que se le realizan a los estudiantes. Pretende abarcar situaciones de toda la vida universitaria aunque faltan aún módulos por implementar. Brinda al educador grandes facilidades ya que posibilita llevar un control docente del estudiante además de que brinda al discente la posibilidad de observar su estado docente. Es el punto de partida para el análisis de la solución propuesta ya que tiene puntos en común con GECMA. Lograr un nivel más detallado de las evaluaciones y la asistencia, donde el educador pueda desglosar una evaluación en objetivos vencidos, generar cortes evaluativos de los grupos, entre otros, son las principales funciones en la solución propuesta; las cuales hoy no son cubiertas por SGU.

Con el estudio de las herramientas expuestas se pudo constatar su funcionamiento. Moodle y sus especificidades (EVA y Mi Moodle en Casa) por su parte es una herramienta muy potente en cuanto a la gestión de cursos, pero trata las evaluaciones en un nivel secundario y no está concebido para llevar un control de la asistencia propiamente. De igual forma SGU resulta de gran valía para los educadores, permitiendo registrar algunas de las evaluaciones que se les realizan a los estudiantes, pero sin el nivel de detalle que se desea, sin tener en cuenta el uso del plan remedial y sin ofrecer al educador la posibilidad de incidir coherentemente en el control docente. Por lo que se arriba a la conclusión de que no se conoce una aplicación que brinde las funcionalidades que brindará GECMA, ya que las estudiadas están desarrolladas con otros propósitos.

1.6 Metodologías de desarrollo de software

Una metodología de desarrollo de software es un conjunto de pasos y procedimientos que deben seguirse para desarrollar software. En un proceso de desarrollo la experiencia ha demostrado que la clave del éxito de un proyecto de software es la elección correcta de esta, pues puede conducir al programador a desarrollar un buen sistema de software. La elección de la metodología adecuada es más importante que utilizar las mejores y más potentes herramientas. La idea no es tratar de ver cuál es mejor o peor, sino de cuándo usar una u otra, pues esto va de acuerdo al tipo de proyecto, a los recursos con los que se cuentan (tiempo,

dinero, etc.) y a la facilidad de interacción con el usuario real. Las metodologías son el camino a seguir para desarrollar software de una manera sistemática, las metodologías persiguen tres necesidades principales (MADDISON, 1983):

- Mejores aplicaciones, conducentes a una mejor calidad.
- Un proceso de desarrollo controlado.
- Un proceso normalizado en una organización, no dependiente del personal.

Las metodologías de desarrollo de software se clasifican en:

Figura 2. Metodologías Ágiles y Tradicionales.

La elección respecto a la utilización o no de una determinada metodología depende principalmente del grado de predictibilidad que se desee tener en el desarrollo. En el desarrollo de la presente solución informática se selecciona el uso de una metodología de desarrollo ágil, debido a que se cuenta con un solo desarrollador para realizar las tareas, además de que estas permiten simplicidad de sus reglas y prácticas, y son flexibles ante los cambios que puedan surgir en el trascurso de la implementación.

Metodología Ágil	Metodología No Ágil (Tradicional)
Pocos artefactos	Más artefactos
Pocos roles	Más roles
No existe un contrato tradicional o al menos es bastante flexible	Existe un contrato prefijado
El cliente es parte del equipo de desarrollo (además in-situ)	El cliente interactúa con el equipo de desarrollo mediante reuniones
Grupos pequeños (< 10 integrantes) y trabajando en el mismo sitio	Grupos grandes
Menos énfasis en la arquitectura	La arquitectura es esencial

Figura 3. Comparación entre metodologías Ágiles y Tradicionales (METODOLOGIAS, 2012).

1.6.1 Metodologías ágiles de desarrollo

A principios de la década del 90, surgió un enfoque que fue bastante revolucionario para su momento ya que iba en contra de la creencia de que mediante procesos altamente definidos se iba a lograr obtener software en tiempo, costo y con la requerida calidad. Consistía en un entorno de desarrollo altamente productivo, en el que participaban grupos pequeños de programadores utilizando herramientas que generaban código en forma automática tomando como entradas sintaxis de alto nivel. En general, se considera que este fue uno de los primeros hitos en pos de la agilidad en los procesos de desarrollo. Las Metodologías Ágiles o "ligeras" constituyen un nuevo enfoque en el desarrollo de software debido a la simplicidad de sus reglas y prácticas, su orientación a equipos de desarrollo de pequeño tamaño, su flexibilidad ante los cambios y su ideología de colaboración. (HERNÁN, 2004).

Las Metodologías Ágiles valoran:

- Al individuo y las interacciones en el equipo de desarrollo más que a las actividades y las herramientas.
- Desarrollar software que funciona más que conseguir una buena documentación, implica minimalismo respecto del modelado y la documentación del sistema.
- La colaboración con el cliente más que la negociación de un contrato.
- Responder a los cambios más que seguir estrictamente una planificación.

Dentro del grupo de metodologías ágiles se decide utilizar Programación Extrema (XP por sus siglas en inglés), de la cual se abordan a continuación sus beneficios para el proceso de desarrollo del software.

XP: es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en retroalimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico. Este modelo de programación se basa en una serie de metodologías de desarrollo de software en la que se da prioridad a los trabajos que dan un resultado directo y que reducen la burocracia que hay alrededor de la programación. Una de las características principales de este método de programación, es que sus elementos son conocidos desde el principio de la informática. Los autores de XP han seleccionado aquellos que han considerado mejores y han profundizado en sus relaciones y en cómo se refuerzan los unos con los otros. El resultado de esta selección ha sido esta metodología única y compacta. Por esto, aunque no está basada en principios nuevos, sí que el resultado es una nueva manera de ver el desarrollo de software. Entre las principales características de XP sobresalen:

- **Pruebas Unitarias**: Se basa en las pruebas realizadas a los principales procesos, de tal manera que se adelante en algo hacia el futuro, se puedan hacer pruebas de futuras fallas.

- **Re-fabricación:** Se basa en la reutilización de código, para lo cual se crean patrones o modelos estándares, siendo más flexible al cambio.

- **Programación en pares:** Una particularidad de esta metodología es que propone la programación en pares, la cual consiste en que dos desarrolladores participen en un proyecto en una misma estación de trabajo. Cada miembro lleva a cabo la acción que el otro no está haciendo en ese momento.

1.8 Herramientas utilizadas en la implementación de la solución

Una herramienta de desarrollo de software es un programa informático que usa un programador para crear, depurar, gestionar o mantener un programa. Para crear un entorno de trabajo es necesario tener en cuenta las funcionalidades y comodidades que brindan cada una de las herramientas en el desarrollo del software. Se confeccionó el entorno de trabajo para el desarrollo de la solución propuesta con las siguientes herramientas:

Servidor web Apache Tomcat: El Servidor Web Apache Tomcat (versión 7.0.46), es el servidor web por excelencia, su configuración, robustez y estabilidad hacen que millones de servidores reiteren su confianza en este programa. Es uno de los mayores triunfos del software libre (THE NUMBER ONE, 2015).

Entre sus características fundamentales se pueden resaltar:

- Es multiplataforma.
- Es flexible, rápido y eficiente.
- Se desarrolla de forma abierta.
- Es modular, ya que puede ser adaptado a diferentes entornos y necesidades, cuenta con diferentes módulos de apoyo y con la Interfaz de Programación de Aplicaciones (API) de programación de módulos para el desarrollo de módulos específicos.
- Incentiva la realimentación de los usuarios, obteniendo nuevas ideas, informes de fallos y parches para la solución de los mismos.
- Es continuamente actualizado y evoluciona a mayor velocidad.

PrimeFaces (PRIMEFACES, 2015)**:** Es una librería de componentes visuales de código abierto desarrollada y mantenida por Prime Technology, una compañía turca de IT especializada en consultoría ágil, JSF y Java EE. Las principales características de Primefaces son (ENRIQUE LERNE, 2013):

- Soporte nativo de Ajax, incluyendo Push/Comet.
- Kit para crear aplicaciones web para móviles.

- Es compatible con otras librerías de componentes, como JBoss RichFaces.
- Uso de javascript no intrusivo (no aparece en línea dentro de los elementos, sino dentro de un bloque <script>).
- Es un proyecto de código libre, activo y bastante estable entre versiones.

Algunos inconvenientes podrían ser:

- Para utilizar el soporte de Ajax se debe indicar explícitamente, por medio de atributos específicos de cada componente.
- No se puede utilizar el soporte de Ajax de JSF 2 (mediante <f:ajax>) con los componentes de PrimeFaces.

Se decide utilizar para el desarrollo de la aplicación el lenguaje Java por sus innumerables beneficios para el desarrollo web y por las características que a continuación se mencionan:

Java: Es un lenguaje de programación orientado a objetos, creado por Sun Microsystems para poder funcionar en distintos tipos de procesadores. Su sintaxis es muy parecida a la de C o C++, e incorpora como propias algunas características que en otros lenguajes son extensiones: gestión de hilos, ejecución remota, etc. El código Java, una vez compilado, puede llevarse sin modificación alguna sobre cualquier máquina, y ejecutarlo. Esto se debe a que el código se ejecuta sobre una máquina hipotética o virtual, la Máquina Virtual de Java (del inglés Java Virtual Machine, JDK) que se encarga de interpretar el código (ficheros compilados .class) y convertirlo a código particular de la CPU que se esté utilizando (siempre que se soporte dicha máquina virtual). Algunos de los elementos que consiguen que Java sea diferente son los applets: aplicaciones pequeñas, dinámicas, seguras, multiplataforma, activas y en red (Dpto CCIA, 2005).

1.8.1 Herramientas usadas en la modelación visual

Las herramientas CASE (del inglés Computer Aided Software Engineering), utilizan el Lenguaje Unificado de Modelado (del inglés Unified Modeling Language, UML) modelan la información de negocios cuando esta se transfiere entre distintas entidades organizativas en el seno de una compañía. El objetivo primordial de las herramientas de esta categoría consiste en representar

objetos de datos de negocios, sus relaciones, y ayuda a comprender mejor la forma en que fluyen estos objetos de datos entre distintas zonas de negocio.

Luego de un detallado estudio se decidió escoger para el modelado del sistema la herramienta CASE Visual Paradigm (versión 6.4), ya que esta brinda una respuesta rápida y bajos requisitos de memoria del motor de persistencia, solo requiere de una configuración de escritorio, soporta la ingeniería inversa de múltiples formas y permite modelar todos los diagramas de clase. A continuación se ofrecen más detalles sobre la herramienta escogida.

Visual Paradigm (VISUAL-PARADIGM, 2015): Es una herramienta para desarrollo de aplicaciones utilizando modelado UML* 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. Soporta el ciclo de desarrollo completo: análisis y diseño orientado a objeto, construcción, prueba y despliegue, esta herramienta permite una construcción más rápida y eficaz de las aplicaciones, con una calidad elevada y un costo reducido, facilita el trabajo en equipo, fácil de utilizar, es de libre uso, proporciona mayor exactitud, además de facilitar la interoperabilidad con otras herramientas CASE. Permite dibujar todos los diagramas de clase, código inverso, generar código desde diagramas y generar documentación. Proporciona abundantes tutoriales UML, aplicaciones interactivas y proyectos como soporte. Tiene como uno de sus principales factores que es de distribución gratuita.

1.8.2 Selección del Sistema gestor de base de datos para el desarrollo de la aplicación

Un Sistema Gestor de Base de Datos (SGBD) es un conjunto de programas que permiten crear y mantener una base de datos, asegurando su integridad, confidencialidad y seguridad. Por tanto, debe permitir (SGBD, 2015):

- Definir una base de datos: especificar tipos, estructuras y restricciones de datos.
- Construir la base de datos: guardar los datos en algún medio controlado por el mismo SGBD.
- Manipular la base de datos: realizar consultas, actualizarla, generar informes.

En la actualidad existen varios SGBDs, entre los más usados están Oracle, SQL Server de Microsoft, PostgreSQL y MySQL Server.

Se decide utilizar PostgreSQL por ser un SGBD muy potente y el gestor de código abierto más popular y usado en el mundo principalmente por su simpleza, rapidez y robustez. Facilita la exportación e importación de datos. Posee gestión de usuarios y contraseñas, manteniendo un alto nivel de seguridad en los datos, y soporta gran cantidad de tipos de datos para las columnas usadas en la base de datos. A continuación se detallan aspectos fundamentales que facilitar la elección de PostgreSQL como SGBD.

PostgreSQL (POSTGRESQL, 2015): Es sin duda el SGBD de código abierto (gratuito y con código fuente disponible) más avanzado del mundo. Posee las características de los más potentes sistemas comerciales como Oracle o SQL Server. Entre ellas se pueden destacar:

- Completo soporte para transacciones: Una transacción está formada por un conjunto de acciones de forma que o se ejecutan todas ellas o bien ninguna. Utilizando transacciones se asegura la consistencia de los datos.
- Procedimientos almacenados. Código ejecutable que se almacena compilado en el servidor. Entre otras cosas, permite optimizar las aplicaciones evitando transferencias innecesarias a través de la red (aplicaciones con parte cliente y parte servidor). Los procedimientos almacenados se pueden escribir con su propio lenguaje de programación (PL/pgSQL) o bien en Perl o Lenguaje de Herramientas de Comando (del inglés Tool Command Language, TCL)
- Disparadores. Procedimientos almacenados que se lanzan automáticamente bajo determinadas circunstancias como actualizaciones de campos. Permite establecer reglas de integridad y consistencia a nivel del servidor de base de datos.
- Vistas. Conjunto de registros resultado de una consulta que se comportan como una tabla física para facilitar su manejo.
- Orientación a objetos con herencia de tablas.

1.8.3 Entornos de desarrollo

Un Entorno de Desarrollo Integrado (del inglés Integrated Development Environment, IDE) es un programa informático compuesto por un conjunto de herramientas de programación. Los IDE proveen de un marco de trabajo amigable para la mayoría de los lenguajes de programación tales como C++, Python, Java, C#, Delphi, Visual Basic, etc. Es posible que un mismo desarrollo integrado funcione con varios lenguajes de programación (PROGRAMACION Y DESARROLLO, 2013). Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, o sea, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica. Los IDE pueden ser aplicaciones por sí solas o pueden ser parte de aplicaciones existentes. (GPL 34, 2013).

Para la implementación de la solución propuesta se decidió utilizar el IDE Eclipse Luna teniendo en cuenta que es un IDE libre y gratuito sin restricciones de uso, multilenguaje, multiplataforma y permite el desarrollo de aplicaciones completas para la entrega al cliente, permite también la creación de ventanas, menús, barras de herramientas y otros elementos básicos y fundamentales a la hora de la construcción de un proyecto.

Eclipse (ECLIPSE, 2015): es un entorno de desarrollo integrado multiplataforma (utilizado para los lenguajes C, C++, Python y Java entre otros) de código abierto, utilizado en su mayoría para desarrollar otros entornos de desarrollo (como el JDT). Fue originalmente desarrollado por IBM para pasar a la familia de herramientas Visual Ageque que la marca poseía. Sin embargo en la actualidad esta herramienta está siendo desarrollada por la Fundación Eclipse, una organización independiente sin ánimo de lucro que intenta fomentar una comunidad de código abierto así como un conjunto de productos complementarios, capacidades y servicios. La versión existente en la actualidad de esta herramienta ofrece las siguientes características: editor de texto, resaltado de sintaxis, compilación en tiempo real, pruebas unitarias con JUnit, control de versiones con CVS (del inglés Concurrent Versions System), integración con Apache Ant y asistentes para el inicio en algunos de los elementos soportados (como proyectos, clases y test). Además, a través del uso de componentes se pueden utilizar tanto Subversion para realizar el control de versiones como Hibernate para desarrollar las funciones de un motor de persistencia (IDE, 2012). La Plataforma Eclipse está diseñada para afrontar las siguientes necesidades:

- Tecnologías a utilizar.
- Soportar la construcción de gran variedad de herramientas de desarrollo.
- Soportar las herramientas proporcionadas por diferentes fabricantes de software independientes.
- Facilitar una integración transparente entre todas las herramientas y tipos de contenidos sin tener en
- Proporcionar entornos de desarrollo gráfico (GUI) o no gráficos.
- Ejecutarse en una gran variedad de sistemas operativos, incluyendo Windows® y Linux™.
- Hacer hincapié en que el lenguaje de programación sea Java para la construcción de nuevos componentes.

Consideraciones parciales

A lo largo del capítulo se han expuesto los principales conceptos referentes al tema, arribando a las siguientes consideraciones:

- La MD1 reviste importancia en la formación de los futuros profesionales de las Tecnologías de la Informática y las Comunicaciones, puesto que proporciona las bases matemáticas para diferentes aspectos de la informática y desarrolla la capacidad para representar y razonar.
- Existe un deficiente uso de las TIC para el control del PEA de MD 1, dado por la no existencia de una herramienta que permita llevar de la mano el proceso de aprensión de los estudiantes a lo largo de las asignaturas.
- Para el desarrollo de la aplicación GECMA se utilizará: Apache Tomcat 7 como servidor, PrimeFaces en su versión 5.1 para el diseño visual, Eclipse Luna como IDE, PostgreSQL 9.1 como gestor de base de datos. Además se define como metodología de desarrollo XP. El lenguaje de programación a utilizar será Java.

Capítulo 2. Características y Diseño del Sistema

2.1 Introducción

En este capítulo se realiza una valoración de las primeras dos fases del ciclo de vida de la Metodología XP, exploración y planificación. Se establecen las historias de usuarios, las cuales servirán para un mejor entendimiento y conocimiento del software. Estas historias de usuarios son escritas por el cliente, representando en estas las principales necesidades del sistema.

2.2 Descripción de los procesos vinculados al campo de acción

Actualmente la gestión y control del PEA en la MD1, en la UCI, recae directamente en la responsabilidad y profesionalidad del educador que imparte dicha asignatura. Para lo que el docente debe tener un seguimiento estricto de las evaluaciones, asistencia y comportamiento de sus estudiantes, con el propósito de trazar estrategias que mejoren la situación docente de cada uno de sus discípulos.

2.3 Flujo actual del proceso

En la UCI es fundamental la concepción del PEA de todas las materias que se imparten. La MD1 no queda fuera de esto, ya que es una asignatura primordial en la formación de los futuros ingenieros. En cada turno de clase, el profesor debe registrar la asistencia de sus estudiantes, además de esto debe fomentar la participación activa en las clases para conocer el estado docente de estos, para lo que existen diferentes formas de evaluación, mediante las cuales el educador puede saber el estado de los discentes ya sea en el tema que se está impartiendo o en la asignatura en general. Para guardar tanto las evaluaciones diarias como la asistencia, el educador cuenta con un registro, en formato duro, en el cual se preserva esta información. Cuando se desea realizar una búsqueda del estado docente de un estudiante cualquiera, el educador tiene que buscar en dicho registro y manualmente realizar cálculos de las evaluaciones. A la hora de hacer alguna comparación entre estudiantes se torna difícil por el procedimiento antes expuesto.

2.4 Descripción del sistema propuesto

Luego del análisis realizado y la problemática existente se propone desarrollar una aplicación web que permita ayudar al control del PEA de la MD1 para que a través del mismo se pueda acceder a toda la información relacionada con los estudiantes de un grupo dado, facilitando la gestión de la información de cada estudiante en la asignatura. La aplicación está compuesta por

un banner superior, un panel lateral izquierdo y un panel central donde se le brindan al usuario las siguientes opciones:

Autenticarse: La autenticación en la aplicación se realiza mediante la comprobación de la clave introducida por un usuario y la correspondiente a este en la base de datos en la tabla usuarios.

Inicio: Se muestra información referente a la asignatura.

Grupos: Permite visualizar los diferentes grupos que atiende el profesor, mostrando por cada grupo cada uno de sus miembros y una descripción de estos. Permitiendo además la gestión (Insertar, Modificar, Eliminar, Mostrar) de grupos en la aplicación.

Estudiantes: Posibilita la gestión (Insertar, Modificar, Eliminar, Mostrar) de todos los aspectos relacionados con los estudiantes, mostrándose inicialmente una síntesis de los estudiantes de cada grupo que atiende el profesor. De igual forma se puede realizar comparaciones entre estudiantes mediante filtros, tales como resultados docentes y la asistencia.

Asistencia: Permite gestionar (Insertar, Modificar, Eliminar, Mostrar) toda la información estudiantil referente a la asistencia. Muestra una gráfica con la asistencia de los estudiantes en el presente curso.

Evaluaciones: Brinda las opciones de gestión de evaluaciones que se le realicen a los estudiantes.

Ayuda: Se muestra una ayuda para que la navegación del usuario en la aplicación sea sencilla.

2.5 Fase de exploración

En esta fase de la metodología XP es donde los clientes plantean a grandes rasgos las Historias de Usuario (HU) que son de interés para la primera entrega del producto. Al mismo tiempo, el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. La fase de exploración toma de pocas semanas a pocos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnología.

2.5.1 Requerimientos no funcionales del sistema

Los requerimientos no funcionales (RNF) son propiedades o cualidades que debe tener el producto. Estas propiedades son las características que hacen al producto atractivo, usable, rápido o confiable. Entre los requerimientos no funcionales del sistema propuesto se encuentran:

Usabilidad:

- Estética:

  RNF 1: Los botones utilizados en la aplicación deben tener un nombre o descripción entendible para el usuario.

  RNF 2: Debe existir una distribución y categorización de los contenidos que facilita la navegación al usuario.

- Accesibilidad:

  RNF 3: La aplicación tiene como usuario final los educadores que imparten la asignatura de MD1 en la UCI.

  RNF 4: El usuario recibe una ayuda para orientarse en la función que va a realizar sobre el sistema u otra tarea en general.

Eficiencia de desempeño:

- Utilización de recursos:

  RNF 5: Estaciones de trabajo (PC Cliente).

  RNF 6: Se utiliza un SGBD con soporte para grandes volúmenes de datos y alta velocidad de procesamiento. Con un tiempo de respuesta rápido en accesos concurrentes.

- Capacidad:

  RNF 7: Navegador web: Internet Explorer 7.0 o superior, Mozilla Firefox 2.0 o superior, Opera, Safari.

  RNF 8: Servidor Aplicación: Apache 1.3 o superior.

  RNF 9: Versión de Java 1.6 o superior.

  RNF 10: SGBD PostgreSQL 8.2 o superior.

RNF 11: Los servidores proxy, web y de bases de datos deben poseer 512 MB de memoria RAM como mínimo.

- Comportamiento temporal:

RNF 12: El diseño debe tener visibilidad en los principales navegadores.

Portabilidad:

- Adaptabilidad:

RNF 13: Sistema multiplataforma.

Seguridad:

- Confidencialidad:

RNF 14: La información manejada por el sistema está protegida de acceso no autorizado y divulgación.

- Integridad:

RNF 15: Realizar salvas periódicas de la información y BD en otros dispositivos, como solución ante la ocurrencia de problemas.

Fiabilidad:

- Disponibilidad:

RNF 16: El sistema posee un 100% de disponibilidad por lo que podrá ser usado las 24 horas del día por sus clientes.

2.5.2 Historias de usuario

El primer paso de cualquier proyecto que siga XP es definir las HU con el cliente. Las HU tienen la misma finalidad que los casos de uso pero con algunas diferencias: constan de 3 o 4 líneas escritas por el cliente en un lenguaje no técnico sin hacer mucho hincapié en los detalles; no se debe hablar ni de posibles algoritmos para su implementación ni de diseños de base de datos adecuados. Son usadas para estimar tiempos de desarrollo de la parte de la aplicación que describen. También se utilizan en la fase de pruebas, para verificar si el programa cumple con lo que especifica la HU. Cuando llega la hora de implementar una historia de usuario, el cliente y los desarrolladores se reúnen para concretar y detallar lo que tiene que hacer dicha historia. El tiempo de desarrollo ideal para una HU es entre 1 y 3 semanas. Durante el análisis en la fase

de exploración se identificaron 23 HU, cada una de estas, respondiendo a las diferentes funcionalidades solicitadas por el cliente. A continuación se describen las principales HU definidas, las restantes HU se pueden consultar en los anexos del tal al tal del trabajo de diploma.

Tabla 1. HU Insertar estudiante.

Historia de usuario	
Número: 1	**Usuario:** Profesor de Matemática Discreta 1
Nombre historia: Insertar estudiante	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 0.4	**Iteración asignada:** 1
Programador responsable: Yandry Gutierrez Rivero	
Descripción: El flujo comienzo cuando el profesor desea insertar un estudiante en la aplicación. Para ello debe seleccionar la opción ¨Estudiantes¨ del menú principal donde se le muestra un formulario para introducir los datos de dicho estudiante, que serían la información personal, dirección e información de contacto. Luego debe presionar el botón “Insertar” para que el estudiante quede guardado en la aplicación, en caso contrario debe seleccionar la opción “Cancelar” y se reiniciara el formulario.	
Observaciones: Para poder insertar un estudiante es necesario que exista al menos un grupo registrado en la aplicación.	

Tabla 2 HU Eliminar estudiante.

Historia de usuario	
Número: 2	**Usuario:** Profesor de Matemática Discreta 1
Nombre historia: Eliminar estudiante	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 0.4	**Iteración asignada:** 1

Programador responsable: Yandry Gutierrez Rivero
Descripción: El flujo comienza cuando el profesor desea eliminar un estudiante en la aplicación. Para ello debe seleccionar la opción Estudiantes del menú principal y seleccionar el estudiante que desea eliminar.
Observaciones:

Tabla 3 HU Modificar estudiante.

Historia de usuario	
Número: 3	**Usuario:** Profesor de Matemática Discreta 1
Nombre historia: Modificar estudiante	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 0.5	**Iteración asignada:** 1
Programador responsable: Yandry Gutierrez Rivero	
Descripción: El flujo comienza cuando el profesor desea modificar los datos de un estudiante en la aplicación. Para ello debe seleccionar la opción “Estudiantes” del menú principal y seleccionar de la lista de estudiantes que se muestra, el estudiante del cual desea modificar los datos. Introduce los datos a modificar y oprime la opción “Modificar”, en caso contrario debe seleccionar la opción “Cancelar” y se reiniciara el formulario de modificación.	
Observaciones: Para poder modificar los datos de un estudiante debe existir al menos uno al cual realizarle esta acción.	

Tabla 4. HU Mostrar información de estudiante.

Historia de usuario

Número: 21	**Usuario:** Profesor de Matemática Discreta 1
Nombre historia: Mostrar información de estudiante	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 0.2	**Iteración asignada:** 1
Programador responsable: Yandry Gutierrez Rivero	
Descripción: El flujo comienza cuando el profesor desea observar la información de un estudiante al que imparte clases. Para ello debe seleccionar la opción “Estudiantes” del menú principal y seleccionar el estudiante. Se muestra la información personal, dirección, información de contacto, además del estado docente y de asistencia que presenta dicho estudiante.	
Observaciones: Si el estudiante seleccionado no tiene evaluaciones registradas se muestra un mensaje a modo de información sobre esto.	

Tabla 5. HU Graficar estado docente de un estudiante.

	Historia de usuario
Número: 22	**Usuario:** Profesor de Matemática Discreta 1
Nombre historia: Graficar estado docente de un estudiante.	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 0.6	**Iteración asignada:** 3
Programador responsable: Yandry Gutierrez Rivero	
Descripción: El flujo comienza cuando el profesor desea ver el estado docente de un estudiante, para ello debe seleccionar en el menú “Estudiantes” la opción “Estado General”.	
Observaciones: Se muestra una gráfica de barras con las notas y el tipo de evaluación.	

2.6 Fase de planificación

Durante la fase de planificación se realiza una estimación del esfuerzo que costará implementar cada HU, para ello se elaboran las Tareas de Ingeniería (T) necesarias para desarrollar dichas historias, con una estimación de tiempo para cada funcionalidad, además se define el plan de iteraciones que explica qué implementar en cada iteración hasta obtener la versión final del producto.

2.6.1 Tareas de ingeniería

Fueron definidas 26 tareas de ingeniería para la implementación de las historias de usuario. A continuación se describen algunas de estas tareas, las otras que fueron confeccionadas se pueden encontrar en los anexos del trabajo de diploma.

Tabla 6. T Diseño e implementación de la funcionalidad Insertar estudiante.

Tarea	
Número Tarea: 1	**Número Historia:** 1
Nombre tarea: Diseño e implementación de la funcionalidad Insertar estudiante	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha inicio: 10/4/2015	**Fecha fin:** 14/4/2015
Programador responsable: Yandry Gutierrez Rivero	
Descripción: Se diseñó e implementó la funcionalidad para insertar un estudiante en la aplicación y guardar los datos en la BD de la aplicación.	

Tabla 7. T Diseño e implementación de la funcionalidad Eliminar estudiante.

Tarea	
Número Tarea: 2	**Número Historia:** 2
Nombre tarea: Diseño e implementación de la funcionalidad Eliminar estudiante	

Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha inicio:15/4/2015	**Fecha fin:** 19/4/2015
Programador responsable: Yandry Gutierrez Rivero	
Descripción: Se diseñó e implementó una tabla en la cual se mostrara la información de todos los estudiantes, donde se debe seleccionar el estudiante que se desee eliminar de la aplicación.	

Tabla 8. T Diseño e implementación de la funcionalidad Modificar estudiante.

Tarea	
Número Tarea: 3	**Número Historia:** 3
Nombre tarea: Diseño e implementación de la funcionalidad Modificar estudiante	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.5
Fecha inicio: 20/4/2015	**Fecha fin:**25/4/2015
Programador responsable: Yandry Gutierrez Rivero	
Descripción: Se diseñó e implementó una tabla en la que se mostraran los datos de todos los estudiantes, donde se debe seleccionar el estudiante al cual se le deseen modificar sus datos.	

Tabla 9. T Diseño e implementación de la funcionalidad Insertar grupo.

Tarea	
Número Tarea: 4	**Número Historia:** 4
Nombre tarea: Diseño e implementación de la funcionalidad Insertar grupo	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha inicio: 26/4/2015	**Fecha fin:** 30/4/2015

Programador responsable: Yandry Gutierrez Rivero
Descripción: Se diseñó e implementó una interfaz para insertar un grupo en la aplicación y guardar los datos en la BD.

Tabla 9. T Diseño e implementación de la funcionalidad Graficar estado docente de un estudiante.

	Tarea
Número Tarea: 24	**Número Historia:** 22
Nombre tarea: Diseño e implementación de la funcionalidad Graficar estado docente de estudiante	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.6
Fecha inicio: 16/5/2015	**Fecha fin:** 22/5/2015
Programador responsable: Yandry Gutierrez Rivero	
Descripción: Se diseñó e implementó la funcionalidad para graficar el estado docente de un estudiante.	

2.7 Plan de entregas del sistema

Luego de creadas las HU, estas servirán para crear el plan estimado de entregas. Para crear el mismo fue necesario convocar a una reunión con el cliente. El plan de entregas se utiliza para crear los planes de iteración. Con cada HU previamente evaluada en tiempo de desarrollo ideal, el cliente las agrupó en orden de importancia. Un periodo ideal es el tiempo que tomaría implementar dicha historia si no se tiene nada más que hacer. De esta forma se trazó el plan de entregas en función de estos dos parámetros:

- tiempo de desarrollo ideal.
- grado de importancia para el cliente.

Tabla 10. Plan de entregas.

Iteración	Entrega	Fecha
1	Versión 0.4	10 de abril de 2015
2	Versión 0.8	1 de mayo de 2015
3	Versión 1.0	22 de mayo de 2015

2.8 Plan de Iteraciones

El plan de iteración consiste en seleccionar las historias de usuario que, según el plan de entregas, corresponderían a cada iteración. Cada iteración corresponde a un periodo de tiempo de desarrollo del proyecto de entre una y tres semanas. De esta forma, un proyecto, se divide en una docena de iteraciones, más o menos. Al principio de cada iteración se debería convocar una reunión para trazar el plan de iteración correspondiente. Habrá suficiente tiempo para añadir la funcionalidad extra cuando sea realmente importante según el plan de entregas. Se usará la velocidad del proyecto para determinar si una iteración está sobrecargada. La suma de los días que costará desarrollar todas las tareas de la iteración no debería sobrepasar la velocidad del proyecto de la iteración anterior. Si la iteración está sobrecargada, el cliente deberá decidir que historias de usuario retrasar a una iteración posterior. Si, por el contrario, la iteración tiene huecos se rellenará con otras historias de usuario.

Cada historia de usuario se transformará en tareas de desarrollo. Cada tarea de desarrollo corresponderá a un periodo ideal de uno a tres días de desarrollo. Es necesario mantener vigiladas la velocidad del proyecto y el movimiento de historias de usuario. Puede ser necesario volver a calcular las historias de usuario y negociar el plan de entrega cada de tres a cinco iteraciones. Como estaremos siempre implementando las historias de usuario más importantes para el cliente, estaremos haciendo lo máximo posible por nuestro cliente.

2.8.1 Velocidad del proyecto

La velocidad del proyecto es una medida de cuán rápido se está desarrollando. Se usa para determinar cuántas historias de usuario pueden ser implementadas antes de una fecha dada (tiempo), o cuánto tiempo es necesario para llevar a cabo un conjunto de historias (alcance).

Cuando se realiza una planificación por alcance se divide el número total de semanas entre la velocidad de proyecto para determinar cuántas iteraciones estarán disponibles.

Tabla 11. Plan de iteraciones

Iteración	Orden de las historias de usuario a implementar	Duración total de la iteración
Iteración 1	Insertar estudiante. Modificar estudiante. Eliminar estudiante. Insertar grupo. Modificar grupo. Eliminar Grupo. Mostrar información de un estudiante. Mostrar información de un grupo.	3 semanas
Iteración 2	Insertar evaluación. Modificar evaluación. Eliminar evaluación. Insertar registro de asistencia. Modificar registro de asistencia. Eliminar registro de asistencia. Mostrar registro de asistencia de un estudiante. Mostrar evaluaciones de un estudiante.	3 semanas
Iteración 3	Graficar estado de asistencia de un estudiante. Graficar estado de asistencia de un	3 semanas

	grupo. Graficar estado docente de un estudiante. Graficar estado docente de un grupo. Insertar usuario. Modificar usuario. Eliminar usuario.	

2.9 Arquitectura de software

Se entiende por Arquitectura de Software la representación de alto nivel de la estructura de un sistema o aplicación, que describe las partes que la integran, las interacciones entre ellas, los patrones que supervisan su composición y las restricciones a la hora de aplicar esos patrones (SALVERT y PEREZ, 2000). En el campo de la ingeniería de software, el concepto de arquitectura ha sido fundamental como estrategia para enfrentar la complejidad del desarrollo de soluciones informáticas y establecer acuerdos de diseño de alto nivel para estas. En una arquitectura de software confluyen tres elementos fundamentales:

- Los modelos que definen la estructura, topología y dinámica del sistema.
- La trazabilidad o correspondencia de dichos modelos con los requisitos o necesidades establecidas en el contexto que va a operar la solución.
- Las reglas, los principios y justificaciones que rigen la arquitectura y que sustentan las decisiones que se tomaron.

La arquitectura de software es fundamental a la hora de garantizar que la solución cumpla con los criterios de calidad establecidos en los requisitos (ATUESTA, 2007). El arquitecto es quien define las tecnologías y herramientas a utilizar, como frameworks, estilos arquitectónicos, lenguajes de implementación y patrones de diseño (UNADECODIGO, 2007). Así se estará construyendo una base sobre la cual se desarrollará un software fiable, económico y correspondiente a las necesidades y requisitos del cliente.

2.9.1 Modelo Vista Controlador

Modelo Vista Controlador (MVC) es un estilo de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario y la lógica de control en tres componentes distintos. Aunque el patrón MVC se ve frecuentemente en aplicaciones web, donde la vista es la página HTML5 y el código que provee de datos dinámicos a la página, el mismo no está estrechamente vinculado con este tipo de aplicaciones, pues al ser un estilo arquitectónico, propone una vista estructural de alto nivel. El modelo lo constituyen los datos con los que trabaja la aplicación y la lógica de negocio y el controlador es el responsable de recibir los eventos de entrada desde la vista (UNADECODIGO, 2007).

El modelo: Conjunto de clases que representan la información del mundo real que el sistema debe procesar, sin tomar en cuenta ni la forma en la que esa información va a ser mostrada ni los mecanismos que hacen que esos datos estén dentro del modelo. El modelo es el responsable de definir las reglas de negocio y llevar un registro de las vistas y controladores del sistema.

- El Modelo: es el objeto que representa los datos del programa. Maneja los datos y controla todas sus transformaciones. El Modelo no tiene conocimiento específico de los Controladores o de las Vistas, ni siquiera contiene referencias a ellos. Es el propio sistema el que tiene encomendada la responsabilidad de mantener enlaces entre el Modelo y sus Vistas, y notificar a las Vistas cuando cambia el Modelo.

- La Vista: es el objeto que maneja la presentación visual de los datos representados por el Modelo. Genera una representación visual del Modelo y muestra los datos al usuario. Interactúa con el Modelo a través de una referencia al propio Modelo.

- El Controlador: es el objeto que proporciona significado a las órdenes del usuario, actuando sobre los datos representados por el Modelo. Cuando se realiza algún cambio, entra en acción, bien sea por cambios en la información del Modelo o por alteraciones de la Vista. Interactúa con el Modelo a través de una referencia al propio Modelo (MVC, 2015).

Aplicando dicho estilo arquitectónico a la solución propuesta, la arquitectura del software quedó distribuida de la siguiente forma:

Modelo: Paquete DAO, Paquete Modelo

Vista: Paquete Vistas

Controlador: Paquete Beans, Paquete Conversores, Paquete Servicio.

El diagrama que representa esta distribución se puede encontrar en los anexos del trabajo de diploma.

2.10 Patrones de diseño

Un patrón de diseño abarca una idea completa dentro de un programa, y por lo tanto puede también abarcar las fases de análisis y diseño de alto nivel. Sin embargo, dado que un patrón a menudo tiene una implementación directa en código, podría no mostrarse hasta el diseño de bajo nivel o la implementación (Gamma, 1995). Lograr un diseño simple pero a la vez robusto requiere en gran medida del empleo de buenas prácticas. Se tuvieron en cuenta al modelar el sistema los conceptos de bajo acoplamiento para evitar las dependencias excesivas, y la alta cohesión tratando de que cada clase realice labores únicas y bien relacionadas, siempre con la intención de lograr un punto de equilibrio entre ambos. Además, el patrón experto para la realización de las tareas, siendo responsabilidad de la clase que cuenta con los datos involucrados, y el controlador para manejar los eventos del sistema. A continuación se detallan los patrones utilizados durante el desarrollo del sistema.

2.10.1 Patrón Singleton

El patrón Singleton asegura que exista una única instancia de una clase. A primera vista, uno puede pensar que pueden utilizarse clases con miembros estáticos para el mismo fin. Sin embargo, los resultados no son los mismos, ya que en este caso la responsabilidad de tener una única instancia recae en el cliente de la clase. El patrón Singleton hace que la clase sea responsable de su única instancia, quitando así este problema a los clientes.

Adicionalmente, si todos los métodos de esta clase son estáticos, éstos no pueden ser extendidos, desaprovechando así las capacidades polimórficas que nos proveen los entornos orientados a objetos. El funcionamiento de este patrón es muy sencillo y podría reducirse a los siguientes conceptos:

- Ocultar el constructor de la clase Singleton, para que los clientes no puedan crear instancias.

- Declarar en la clase Singleton una variable miembro privada que contenga la referencia a la instancia única que queremos gestionar.
- Proveer en la clase Singleton una función o propiedad que brinde acceso a la única instancia gestionada por el Singleton. Los clientes acceden a la instancia a través de esta función o propiedad.

Estas reglas se cumplen en todas las implementaciones del Singleton, independientemente de los recaudos que deban tomarse para soportar la correcta ejecución en entornos multihilo. El ciclo de vida de los Singleton es un aspecto importante a tener en cuenta (SINGLETON, 2015).

2.10.2 Inyección de dependencias

En informática, Inyección de dependencias es un patrón de diseño orientado a objetos, en el que se suministran objetos a una clase en lugar de ser la propia clase quien cree el objeto. El contenedor inyecta a cada objeto los objetos necesarios según las relaciones plasmadas en un fichero de configuración (Griffin Caprio, 2005). Típicamente este contenedor es implementado por un marco de trabajo externo a la aplicación por lo cual en la aplicación también se utilizará inversión de control al ser el contenedor (almacenado en una biblioteca) quien invoque el código de la aplicación. Esta es la razón por la que los términos de inversión de control e inyección de dependencias se confunden habitualmente entre sí (Martin Flower, 2011). El uso de dicho patrón en la aplicación se evidencia en las clases visuales donde a cada una de ellas para el tratamiento de los datos le será inyectada las dependencias necesarias para realizar dicha función. Se puede tomar como ejemplo el hecho de que para guardar un estudiante, en el formulario que recoge los datos de este se utiliza la inyección de la clase EstudianteBean a la cual se le inyecta un objeto de tipo servicio, el cual es el encargado de tramitar los datos con las clases del paquete DAO (encargadas estas de realizar el acceso a dados).

2.10.3 Patrones GRASP

Los Patrones Generales de Asignación de Responsabilidades (por sus siglas en inglés GRASP) son patrones de diseño son utilizados para la asignación de responsabilidades a una determinada clase. El grupo GRASP está conformado por 5 patrones principales y 4 adicionales aplicables al diseño orientado a objetos. A continuación se refleja la utilización de estos en la solución que se propone:

Experto: Consiste en la asignación de responsabilidades al más competente en información, la clase cuenta con la información necesaria para cumplir la responsabilidad. Es el principio básico de asignación de responsabilidades que suele utilizarse en el diseño Orientado a Objetos.

Utilización: Este patrón se puede observar en la clase GrupoBean ya que se utiliza para realizar todas las acciones que tengan que ver con algún grupo, siendo capaz de validar todas las entradas posibles.

Alta Cohesión: Asignar una responsabilidad de modo que la unión se mantenga a gran escala. Asignar a las clases responsabilidades que trabajen sobre una misma área de aplicación y que no tengan mucha complejidad. Mejoran la claridad y facilidad con que se entiende el diseño.

Utilización: Las clases de Web se encargan de visualizar los datos y manejar las interacciones del usuario, las clases de Controlador se encargan de realizar un modelo estándar de presentación de los datos y las clases de Modelo se encargan de contener los datos.

Bajo Acoplamiento: Asignar una responsabilidad para mantener un engranaje pobre. Es un principio que se debe recordar durante las decisiones de diseño. Soporta el diseño de clases más independientes. Asigna las responsabilidades de forma tal que las clases se comuniquen con el menor número de clases que sea posible

Utilización: Se evidencia en la relación que tienen las clases LoginBean y Login que son las únicas relacionadas con el manejo de los usuarios, siendo además las únicas relacionadas entre sí. Esta estructura permite que un cambio en otro elemento de configuración no afecte a estas clases.

Controlador: Asignar la responsabilidad de controlar el flujo de eventos del sistema a clases específicas. Asigna la responsabilidad del manejo de mensajes de los eventos de un sistema a una clase.

Utilización: Este patrón se evidencia en la clase Servicio, ya que sirve como intermediario entre las clases controladoras. En dependencia del método llamado invoca al algoritmo necesario para cumplir dicha funcionalidad.

Indirección: Asignar la responsabilidad a un objeto intermedio para que medie entre otros componentes o servicios y estos no terminen directamente acoplados.

Utilización: Este patrón se refleja en diferentes clases que actúan como intermediarios entre algunas vistas y los modelos del dominio, como por ejemplo EvaluacionesBean.

Consideraciones parciales

- En el capítulo se detallaron los aspectos fundamentales para el desarrollo de la aplicación: la especificación de los requisitos no funcionales del sistema y la confección de las HU.
- Se definió el Modelo/Vista/Controlador como modelo arquitectónico a utilizar.
- Se detallaron los patrones de diseño utilizados para la asignación de responsabilidades a una determinada clase, GRASP.

Capítulo 3. Diseño, Implementación y prueba del sistema

3.1 Introducción

En el presente capítulo se realiza una valoración de las fases de construcción y prueba de la metodología XP, incidiendo fundamentalmente en las restantes fases de la metodología. Se describen las tarjetas CRC (Contenido, Responsabilidad y Colaboración) para un mejor entendimiento del sistema. Además se exponen las pruebas de aceptación efectuadas al sistema.

3.2 Tarjetas CRC (Cargo o clase, Responsabilidad y Colaboración)

XP estimula el uso de tarjetas CRC como un mecanismo eficaz para pensar en el software en un contexto orientado a objetos. Las tarjetas CRC identifican y organizan las clases orientadas a objetos que son relevantes para el incremento actual de software. Son el único producto de trabajo de diseño que se generan como parte del proceso XP. Estas permiten desprenderse del método de trabajo basado en procedimientos. El nombre de la clase se coloca a modo de título en la tarjeta, las responsabilidades se colocan a la izquierda, y las clases que se implican en cada responsabilidad a la derecha, en la misma línea que su requerimiento correspondiente.

Para poder diseñar el sistema se debe cumplir con tres principios:

- Cargo o Clase
- Responsabilidad
- Colaboración

Las tarjetas determinan el comportamiento de cada actividad. En GECMA cada vista se comporta como un objeto independiente, compuesta por las siguientes clases:

Clases Bean:

- EstudianteBean
- GrupoBean
- EvaluacionBean
- AsistenciaBean

- LoginBean
- UsuarioBean

Clases de dominio:

- Usuario
- Estudiante
- Grupo
- Asistencia
- Evaluacion

Clases del patrón DAO:

- EstudianteDAO
- GrupoDAO
- AsistenciaDAO
- EvaluacionDAO
- UsuarioDAO
- LoginDAO

Clase servicio:

- Servicio

Fueron confeccionadas 18 tarjetas CRC de las cuales se describe a continuación una muestra, las restantes confeccionadas se encuentran en los anexos del trabajo de diploma.

Tabla 12. CRC EstudianteBean.

Clase: EstudianteBean.	
Responsabilidades:	**Colaboradores:**
Guardar	Servicio
Eliminar	EstudianteDAO
Modificar	Estudiante
Actualizar	

Tabla 13. CRC GrupoBean.

Clase: GrupoBean.	
Responsabilidades:	**Colaboradores:**
Guardar	Servicio
Eliminar	GrupoDAO
Modificar	Grupo
Actualizar	

Tabla 14. CRC AsistenciaBean.

Clase: AsistenciaBean.	
Responsabilidades:	**Colaboradores:**
Guardar	Servicio
Eliminar	AsistenciaDAO
Modificar	EstudianteBean
Actualizar	Asistencia

AsistenciaDeEstudiante	

3.3 Validación de la solución propuesta

Uno de los pilares de XP es el proceso de pruebas. XP anima a probar constantemente tanto como sea posible. Esto permite aumentar la calidad de los sistemas reduciendo el número de errores no detectados y disminuyendo el tiempo transcurrido entre la aparición de un error y su detección. También permite aumentar la seguridad de evitar efectos colaterales no deseados a la hora de realizar modificaciones y refactorizaciones. XP divide las pruebas del sistema en dos grupos:

- Pruebas unitarias: encargadas de verificar el código y diseñada por los programadores.
- Pruebas de aceptación o pruebas funcionales: destinadas a evaluar si al final de una iteración se consiguió la funcionalidad requerida diseñadas por el cliente final.

Las pruebas de aceptación son de gran importancia dado que significan la satisfacción del cliente con el producto desarrollado y el final de una iteración y el comienzo de la siguiente por esto, el cliente es la persona adecuada para diseñar las pruebas de aceptación(PRUEBAS, 2015).

3.3.1 Pruebas de caja negra mediante pruebas de aceptación

El objetivo de estas pruebas es verificar los requisitos, por este motivo, los propios requisitos del sistema son la principal fuente de información a la hora de construir los Casos de Pruebas de Aceptación (CPA). Las pruebas de aceptación son creadas a partir de las historias de usuario. Durante una iteración la historia de usuario seleccionada en la planificación de iteraciones generará pruebas de aceptación. El cliente o usuario especifica los aspectos que serán objetos de prueba cuando una historia de usuario ha sido correctamente implementada. Una historia de usuario puede tener más de una prueba de aceptación, tantas como sean necesarias para garantizar su correcto funcionamiento y no se considera completa hasta que no supera estas.

Una prueba de aceptación es como una caja negra. Cada una de ellas representa una salida esperada del sistema. Es responsabilidad del cliente verificar la corrección de las pruebas de

aceptación y tomar decisiones acerca de las mismas. La garantía de calidad es una parte esencial en el proceso de XP. La realización de este tipo de pruebas y la publicación de los resultados debe ser lo más rápido posible para que los desarrolladores puedan realizar con la mayor rapidez los cambios que sean necesarios.

En la ejecución de dichas pruebas a la aplicación se realizaron 3 iteraciones, en las cuales se solucionaron las no conformidades detectadas. En la primera iteración se detectaron 4 no conformidades, luego en la segunda iteración surgió una no conformidad y en la tercera iteración no se obtuvieron no conformidades por lo que fueron evaluados todos los casos de prueba de satisfactorio. Las gráficas que reflejan estos resultados se encuentran en los anexos del trabajo de diploma al igual que los otros casos de pruebas realizados.

Tabla 15. CPA Insertar estudiante.

Caso de prueba de aceptación
Historia de usuario: Insertar estudiante
Nombre: Insertar estudiante
Responsable: Yandry Gutierrez Rivero
Condiciones de ejecución: Para insertar un estudiante debe existir un grupo al cual asignarlo.
Entradas/Pasos de ejecución: Seleccionar en el menú lateral la opción, Estudiantes, llenar el formulario para insertar un estudiante y presionar el botón Guardar.
Resultado esperado: Se inserta un estudiante en la BD.
Evaluación de la prueba: Satisfactoria

Tabla 16. CPA Modificar estudiante.

Caso de prueba de aceptación
Historia de usuario: Modificar estudiante
Nombre: Modificar estudiante

Responsable: Yandry Gutierrez Rivero
Condiciones de ejecución: Para modificar un estudiante este debe existir en la BD.
Entradas/Pasos de ejecución: Seleccionar en el menú lateral la opción, Estudiantes, aparece una tabla con el listado de estudiantes donde se debe seleccionar el que se le deseen modificar sus datos haciendo clic derecho sobre él y seleccionar la opción Modificar.
Resultado esperado: Se modifican los datos de un estudiante en la BD.
Evaluación de la prueba: Satisfactoria

Tabla 17. CPA Eliminar estudiante.

Caso de prueba de aceptación
Historia de usuario: Eliminar estudiante
Nombre: Eliminar estudiante
Responsable: Yandry Gutierrez Rivero
Condiciones de ejecución: Para eliminar un estudiante debe existir este en la BD.
Entradas/Pasos de ejecución: Seleccionar en el menú lateral la opción, “Estudiantes”, aparecerá una tabla con todos los estudiantes donde se debe seleccionar el estudiante a eliminar.
Resultado esperado: Se elimina un estudiante en la BD.
Evaluación de la prueba: Satisfactoria

Tabla 18. CPA Mostrar registro de asistencia de un estudiante.

Caso de prueba de aceptación
Historia de usuario: Mostrar registro de asistencia de un estudiante.

Nombre: Mostrar registro de asistencia de un estudiante.
Responsable: Yandry Gutierrez Rivero
Condiciones de ejecución: Debe existir al menos un estudiante en la BD que tenga reflejado al menos un registro de asistencia.
Entradas/Pasos de ejecución: Seleccionar en el menú lateral la opción, Asistencia, aparecerá una tabla con todos los registros de un estudiante donde haciendo clic derecho sobre él se debe seleccionar la opción Ver registro de asistencia.
Resultado esperado: Se muestra una tabla donde se muestran todos los registros de asistencia de un estudiante.
Evaluación de la prueba: Satisfactoria

Tabla 19. CPA Mostrar evaluaciones de un estudiante.

Caso de prueba de aceptación
Historia de usuario: Mostrar evaluaciones de un estudiante.
Nombre: Mostrar evaluaciones de un estudiante.
Responsable: Yandry Gutierrez Rivero
Condiciones de ejecución: Debe existir al menos un estudiante en la BD que tenga reflejada al menos una evaluación.
Entradas/Pasos de ejecución: Seleccionar en el menú lateral la opción, Evaluaciones, aparecerá una tabla con todas las evaluaciones de un estudiante donde haciendo clic derecho sobre él se debe seleccionar la opción Ver evaluación.
Resultado esperado: Se muestra una tabla donde se observan todas las evaluaciones de un estudiante.
Evaluación de la prueba: Satisfactoria

Consideraciones parciales

- Se realizó la construcción de una versión de la solución propuesta que corresponde con el 95% de la aplicación final.
- Se desarrollaron las fases de diseño y prueba de la metodología XP.
- Para la comprobación del funcionamiento del software se realizaron pruebas de aceptación donde luego de la tercera iteración se obtuvo un 100% de satisfacción.

Conclusiones

- A partir del análisis realizado a la documentación referente al objeto de estudio, se pudo precisar la connotación que alcanza el uso de las tecnologías de la informática y la comunicación en aras de perfeccionar el control sobre el proceso docente educativo en la UCI, así como la inexistencia entre las herramientas estudiadas de una que sea capaz de darle solución a la problemática planteada.

- Mediante el uso de las tecnologías y lenguaje de programación seleccionados para el desarrollo de la aplicación, se obtuvo un producto con un diseño simple e intuitivo que responden a las necesidades alcanzadas a través de la captura de los requisitos funcionales y no funcionales.

- La implementación de la aplicación GECMA proporcionó una herramienta que evidenció la contribución realizada al control docente del PEA en la MD1 así como las pruebas realizadas a esta determinaron la robustez y eficacia del sistema diseñado.

Recomendaciones

- Realizar la obtención de datos de los estudiantes mediante la utilización de un Protocolo Ligero de Acceso a Directorios (LDAP por sus siglas en inglés).
- Agregar secciones a la aplicación para que los estudiantes puedan acceder a ella y observar sus resultados docentes.
- Enriquecer el software con la utilización de foros, chat y otras herramientas colaborativas para permitir la interacción de los estudiantes y contribuir en su desempeño docente.

Bibliografía referenciada

- COLECTIVO DE AUTORES, UCI, 2002ª. Proyección Estratégica de la Universidad de las Ciencias Informáticas. Documentos de la Rectoría. UCI. La Habana. Cuba. 2002ª.
- COLECTIVO DE AUTORES, UCI, 2002b. Modelo del profesional de la UCI. http://intranet2.ci.cu/sites/default/files/pdf_formacion/Modelo%20del%20Profesional%20y%20Objetivos%20Generales.pdf.
- UCI, 2011. Informe semestral de Matemática Discreta 1. Curso 2010-2011. Habana: s.n.
- CARNEIRO R. y otros, 2009. Los Desafíos de las Tics para el cambio educativo. España.
- MINAKATA ARCEO A, 2009. Gestión del conocimiento en educación y transformación de la escuela. Notas para un campo en construcción. México. Sináptica, revista electrónica de educación. http://www.oei.es/mx43.htm
- FALOH BEJERANO. A, 2001. Tendencias del futuro. Nueva Empresa, I, 32-34.
- CITMA, 2001. Ministerio de Ciencia Tecnología y Medio Ambiente. Política para la introducción de la gestión del conocimiento en Cuba. La Habana.
- MARTÍN RODRÍGUEZ E, 2000. Educación a distancia y nuevas tecnologías. Publicado en el libro de SAPERAS, E. y otros.
- ISAEL HERRERA ,2012. Conjunto de Objetos de Aprendizaje para la Matemática Discreta. Tesis de grado, facultad 1.
- THE NUMBER ONE, 2015. http://httpd.apache.org/.
- SINGLETON, 2015. Consultado el 5 de junio de 2015 https://msdn.microsoft.com/es-es/library/bb972272.aspx.
- COLECTIVO DE AUTORES, UCI, 2015. Informe de la asignatura MD1 curso 2014-2015.
- PRUEBAS, 2015 Consultado el 15 de junio de 2015 http://www.lsi.us.es/~javierj/investigacion_ficheros/PSISEXTREMA.pdf
- ENRIQUE VIÑE LERMA, 2013. Consultor tecnológico de desarrollo de proyectos informáticos. Ingeniero Técnico en Informática por la Universidad Politécnica de Madrid.
- FUENTES, MOMPIÉ y CISNERO, IYATNE y CHACÓN, CLARITZA. 2009. “Módulo de objetos interactivos de aprendizaje para apoyar el proceso de enseñanza-aprendizaje en la disciplina de Matemática Aplicada. La Habana: s.n., 2009.
- LUJÁN MORA, SERGIO, 2010. s/f. C++ Paso a Paso. s/f.
- PHP, 2011. PHP. [En línea]. [Citado el: 11 de 12 de 2011.] http://php.net/manual/es/intro-

whatis.php.

- METODOLOGIAS, 2012. http://www.monografias.com/trabajos67/metodologia-desarrollo software / metodologia-desarrollo-softwares2.shtml.
- BONTIS, N. 1999. Managing an Organizational Learning System by Aligning Stocks and Flows of Knowledge: an Empirical Examination of Intellectual Capital and Knowledge. Universidad de Ontario Oeste.
- DPTO. CCIA, 2005. Lenguaje Java y Entorno de Desarrollo. 2005.
- PROGRAMACION Y DESARROLLO, 2013. http://programaciondesarrollo.es/ que-es-un-entorno-de-desarrollo-integrado-ide/.
- GPL 34, 2013. Grupo de Programación del Plantel 34. http://infowebc.net46.net/entornos-de-desarrollo.html.
- DOSIDEAS, 2013. http://www.dosideas.com/wiki/NetBeans.
- PRIMEFACES, 2015. Sitio oficial de Primefaces consultado el 15 de marzo de 2015 http://primefaces.org/.
- VISUAL-PARADIGM, 2015. Sitio oficial de Visual Paradigm consultado el 20 de mayo de 2015 www.visual-paradigm.com.
- SGBD, 2015. Consultado el 26 de mayo de 2015 http://fundametosbdunipanamericana.blogspot.com/2010/11/caracteristicas-de-los-sgbd.html.
- POSTGRESQL, 2015. Sitio oficial de PostgreSQL consultado el 2 de junio de 2015 www.postgresql.org.
- ECLIPSE, 2015. Sitio oficial de Eclipse consultado el 5 de junio de 2015 https://eclipse.org.
- IDE, 2012. Entornos de Desarrollo Integrado .2012. http://petra.euitio.uniovi.es /~i1667065/HD/documentos/Entornos%20de%20Desarrollo%20Integrado.pdf.
- RAMOS y SALABERT, 2000. Isidro Ramos Salavert, María Dolores Lozano Pérez .Ingeniería Del Software Y Bases de Datos: Tendencias Actuales. 2000. pág. 62.

- ATUESTA, 2007. Atuesta, Claudia Maria Zea Restrepo y Maria del Rosario. Hacia Una Comunidad Educativa Interactiva. Medellin: s.n., 2007. pág. 97.
- UNADECODIGO, 2007. El Paradigma Modelo Vista Controlador. [En línea] http://www.unadecodigo.com/2007/05/30/el-paradigma-modelo-vistacontrolador-tutorial-ror-ii.
- CENCOMED, 2007. http://cencomed.sld.cu/socbio2007/trabajos/pdf/t072.pdf.
- PATRON, 2012. http://www.proactivacalidad.com/java/patrones/mvc.html.
- SEBASTIAN, 2013. Ing. Juan. Comusoft. [En línea] 2013. http://www.comusoft.com/modelo-vista-controlador-definicion-y-caracteristicas.
- SOTO, 2006. Gestión del conocimiento. Parte I. Revisión crítica del estado del arte.
- HERNÁNDEZ LUQUE EYLÍN, 2013. Tesis presentada en opción al título de Máster en Gestión de Proyectos Informáticos.