UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 3

SISTEMA INFORMÁTICO DE GESTIÓN DE INDICADORES PARA LA FORMACIÓN DE ESTUDIANTES POTENCIALMENTE TALENTOSOS EN LA UCI

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

Autor: Ivaniet Díaz Romeu

Tutoras: MSc. Dayana Caridad Tejera Hernández

Ing. Olga Yarisbel Rojas Grass

LA HABANA, 30 DE JUNIO DE 2015

DEDICATORIA

A mi familia por las infinitas muestras de amor, sacrificio y entrega incondicional. Por guiarme en cada paso que he dado en la vida, por contribuir en mi educación y superación profesional.

AGRADECIMIENTOS

Hoy es el día más importante de mi vida como estudiante, hoy he conquistado un gran sueño que no hubiese sido posible alcanzar sin el inmenso amor e incondicional apoyo de mi familia, la entrega de profesores y la ayuda de amigos. Agradezco desde lo más profundo de mí:

A Dios por darme fuerzas para seguir adelante y acompañarme en los momentos más difíciles.

A mi mamá Katia por ser tan especial para mí, por sus noches de desvelos y entera dedicación, por educarme y conducirme en la vida.

A mi tío Leonel por todo el apoyo brindado y su cariño, por confiar en mí y demostrarme que siempre se puede un poco más e influir tanto en mi educación.

A mi abuelas Juana e Hilda y mis abuelos Enrique y Diego por todo el amor y cariño que me han dado, por todo el apoyo brindado y quererme como a un hijo.

A mis hermanas Keity y Beatriz por ser tan importantes en mi vida.

A William por toda su ayuda, cariño y apoyo ante cada paso que he dado.

A mi papá por querer siempre lo mejor para mí.

A mis tías Niurka, Maricelys, Ileana e Iniola por todo su amor y ayuda brindada y mi tío Arnaldo por todo su cariño y apoyo.

A mi primas Ariannys, Arialys, Lisandra, Yusnabis, Yusladis, Ibettys, Ismary y Sandra por todo su cariño, ayuda brindada y preocupación.

A mi primo Eyvis y a Yaima por toda la ayuda brindada y su cariño.

A mi bisabuela Ofelia por toda su ayuda y sus consejos, mis tías Celina, La Niña, Celia y Mella por toda la ayuda brindada y tíos Mario, Nolberto, El Chino y Orestes por toda la ayuda y apoyo.

A mima Adelfa y toda la familia de Villa Clara por todas las muestras de amor y cariño que me han dado y todo su apoyo.

A mis tutoras Dayana y Olga Yarisbel por todo tiempo dedicado a guiarme en el desarrollo del trabajo de diploma, por su experiencia y conocimiento brindado y a la profesora Olga por la ayuda ofrecida aun cuando no tenía esa responsabilidad.

A los profesores del tribunal Nemury, Miguel Ángel, Yisel, Yaniris y Katia por todos los señalamientos realizados para guiarme en el desarrollo de la investigación.

A Oneisy, Dannelis, Leydis, Irianny, Darianis, Greter y Yanna por su amistad, amor y cariño, por toda la ayuda brindada en estos años de universidad, por toda su preocupación y apoyo.

AGRADECIMIENTOS

A Rey Manuel, José Javier, Luis Ángel, Leosbel, Sandy, Pablo, Alejandro y Yoan Antonio por toda la ayuda desinteresada, y todo su apoyo en el desarrollo de este trabajo.

A mis compañeros de aula por todos los momentos que hemos compartido juntos.

A Raimer, por su ayuda desinteresada, sacrificio y apoyo; a Yuri y Nilson por toda la ayuda y apoyo brindado.

A Mercy por quererme como a un hijo, por toda la ayuda que me ha dado, por su contante dedicación y preocupación.

A Edisvel, Ámbar y Mercedes, por toda la ayuda brindada, apoyo y preocupación.

A las instructoras María Luisa, Olga Lidia, Aneth y Annia por toda la ayuda brindada en estos años; a Marieta y las psicopedagogas Yolanda, Yaisel, Nidia, Dorita y Dagmara por su amistad, su ayuda, su cariño y ayuda desinteresada y Alier y Daimara por su ayuda y preocupación.

A los profesores Sonia Amada, Arismayda, Amauri, Marianela, Álida por toda la ayuda brindada y por su profesionalidad.

A mis vecinos por su preocupación y apoyo.

A todas las personas que han contribuido en mi formación y que han estado presentes en todos estos años.

Cualquier palabra es poca para expresar mis agradecimientos, a todos:

Muchas gracias

DECLARACIÓN JURADA DE AUTORÍA

Declaro ser autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firman la presente a los 30 días del mes de junio del año 2015.

_____

Ivaniet Díaz Romeu
Autor

_____

MSc. Dayana Caridad Tejera Hernández
Tutora

_____

Ing. Olga Yarisbel Rojas Grass
Tutora

RESUMEN

En el desarrollo de las fases identificación, estimulación, orientación y gestión del proceso de formación de estudiantes potencialmente talentosos intervienen un conjunto de indicadores que contribuyen a medir las potencialidades de los educandos. La gestión y el seguimiento de estos indicadores permiten sustentar y orientar la labor educativa hacia el desarrollo de las habilidades de los estudiantes potencialmente talentosos. En la Facultad 5 de la Universidad de las Ciencias Informáticas radica el proyecto Talenmático, que tiene como objetivo desarrollar y aplicar un modelo de intervención para la atención educativa a los estudiantes potencialmente talentosos de esta universidad. Una de las deficiencias que presenta este proyecto para el cumplimiento de su objetivo es la gestión de indicadores de medición del talento para el seguimiento del desarrollo de las habilidades de los estudiantes potencialmente talentosos, ya que el proceso es engorroso y difícil de realizar de forma manual y personalizada por la cantidad de indicadores que se deben evaluar.

El objetivo del presente trabajo de diploma es desarrollar un sistema informático que permita agilizar la gestión de indicadores de medición del talento para el seguimiento del desarrollo de las habilidades de los estudiantes potencialmente talentosos de la universidad. Para guiar el proceso de desarrollo del sistema se utilizó la metodología de software Variación de AUP para la universidad y en la implementación se emplearon tecnologías y herramientas de software libre. El sistema elaborado permite agilizar el proceso de seguimiento de indicadores que miden el desarrollo de las habilidades de los estudiantes talentosos.

Palabras clave: indicadores, formación, habilidades, talento

ÍNDICE DE CONTENIDO

INTRODUCCIÓN 1

CAPÍTULO I: Fundamentación teórica sobre el seguimiento de indicadores para la formación del talento 5

1.1 Introducción 5

1.2 Definición de conceptos 5

1.2.1 Definiciones de talento 5

1.2.2 Definición de talento informático 5

1.2.3 Definición de agilizar 6

1.3 Proceso de formación de estudiantes potencialmente talentosos 6

1.4 Sistemas informáticos para la gestión de indicadores de medición que evalúen el desarrollo de las habilidades de los estudiantes 7

1.4.1 DocCF, Software de Gestión Escolar 8

1.4.2 Portal Colegios Colombia 8

1.4.3 Entorno Virtual de Aprendizaje 9

1.4.4 Sistemas informáticos desarrollados en la UCI para el desarrollo de habilidades en estudiantes potencialmente talentosos 10

1.4.5 Análisis realizado a los sistemas informáticos 11

1.5 Metodología de desarrollo de software 13

1.5.1 Metodología de desarrollo de software Variación de AUP para la UCI 13

1.6 Lenguaje de modelado de software 15

1.6.1 Lenguaje Unificado de Modelado 2.0 15

1.7 Herramientas para el modelado del sistema 15

1.7.1 Visual Paradigm 8.0 15

1.8 Lenguajes de programación 16

1.8.1 HTML 5 16

1.8.2 CSS 3.0 16

1.8.3 Javascript 1.6 17

1.8.4 PHP 5.5 17
1.9 IDE de programación 17
1.9.1 NetBeans 8.0 17
1.10 Marco de trabajo 18
1.10.1 Symfony 2 18
1.10.2 Bootstrap 3.0 18
1.11 Servidor de aplicaciones web 19
1.11.1 Apache 2.0 19
1.12 ORM 19
1.12.1 Doctrine 2 20
1.13 Servidor de base de datos 20
1.13.1 PostgreSQL 9.1 20
1.14 PgAdmin III 1.14.0 20
1.15 Conclusiones parciales del capítulo 21
CAPÍTULO II: Descripción de la propuesta de solución 22
2.1 Introducción 22
2.2 Modelado del negocio 22
2.2.1 Modelo conceptual 22
2.3 Requisitos 24
2.3.1 Fuentes para la obtención de requisitos y técnicas de identificación de requisitos 24
2.3.2 Especificación de requisitos de software 25
2.3.3 Validación de requisitos 30
2.3.4 Administración de requisitos 31
2.4 Análisis y diseño 32
2.4.1 Diseño arquitectónico 33
2.4.2 Modelo de datos 34
2.4.3 Modelado del diseño 35
2.5 Implementación 41

2.5.1 Modelo de implementación 41
2.5.2 Estándares de codificación 45
2.6 Interfaz del sistema SIAGEPTI 46
2.7 Conclusiones del capítulo 47
CAPÍTULO III: Validación del sistema 48
3.1 Introducción 48
3.2 Validación del diseño del sistema 48
3.2.1 Métrica Tamaño operacional de clase (TOC) 48
3.2.2 Métrica Relaciones entre clases (RC) 50
3.3 Pruebas de software 52
3.3.1 Pruebas internas 52
3.3.2 Pruebas de liberación 52
3.3.3 Pruebas de aceptación 52
3.4 Métodos y técnicas de pruebas 53
3.5 Aplicación y resultados de las pruebas realizadas al sistema 53
3.5.1 Pruebas internas 53
3.5.2 Pruebas de liberación 56
3.5.3 Aplicación y resultados de las pruebas de aceptación 59
3.6 Validación del objetivo de la investigación 59
3.7 Conclusiones del capítulo 61
CONCLUSIONES 62
RECOMENDACIONES 63
REFERENCIAS BIBLIOGRÁFICAS 64

ÍNDICE DE FIGURAS

Figura 1. Metodología de desarrollo de software Variación de AUP para la UCI 14

Figura 2. Modelo conceptual para la gestión de indicadores de medición 23

Figura 3. Descripción del requisito funcional Adicionar tarea 29

Figura 4. Prototipo del requisito funcional Adicionar tarea 30

Figura 5. Fragmento de la Matriz de trazabilidad Requisitos – Modelo conceptual 32

Figura 6. Patrón arquitectónico MVC (Fuente: [45]) 33

Figura 7. Aplicación del patrón MVC en el SIAGEPTI 34

Figura 8. Modelo de datos del SIAGEPTI 35

Figura 9. Fragmento del Diagrama de clases del diseño de la agrupación Adicionar tarea 36

Figura 10. Ejemplo del uso de los patrones GRASP en el SIAGEPTI 38

Figura 11. Ejemplo de uso del patrón Decorador en el SIAGEPTI 40

Figura 12. Diagrama de componentes del SIAGEPTI 42

Figura 13. Diagrama de despliegue del SIAGEPTI 44

Figura 14. Ejemplificación del uso de los estándares de codificación 46

Figura 15. Interfaz de la sesión del profesor con el rol de administración para la gestión de indicadores 47

Figura 16. Resultados de la aplicación de la métrica TOC 49

Figura 17. Resultados de la aplicación de la métrica RC 51

Figura 18. Código de la implementación del requisito funcional Modificar indicador 54

Figura 19. Grafo de flujo del código de implementación del requisito funcional Modificar indicador 54

Figura 20. Resultados de las pruebas de liberación 58

ÍNDICE DE TABLAS

Tabla 1. Resultados del estudio realizado a los sistemas informáticos 11

Tabla 2. Aportes y limitaciones de los sistemas informáticos analizados 12

Tabla 3. Listado de los requisitos funcionales de la propuesta de solución 26

Tabla 4. Resultados de la aplicación de la métrica TOC 49

Tabla 5. Resultados de la aplicación de la métrica RC. 51

Tabla 6. Diseño de caso de prueba para el camino 2 55

Tabla 7. Escenarios de prueba para el requisito funcional Adicionar tarea 57

Tabla 8. Casos válidos y no válidos del requisito funcional Adicionar tarea 58

Tabla 9. Cuadro lógico de Iadov 59

Tabla 10. Resultados de las escalas de satisfacción 61

INTRODUCCIÓN

La sociedad del presente se caracteriza por ser cada día más dependiente del conocimiento y la tecnología, y de convivir bajo la presencia de una imperante globalización y competitividad a escala mundial. Estos factores han provocado que las organizaciones centren sus esfuerzos inversores en las personas y el conocimiento, con el objetivo de utilizar esos potenciales en su actividad productiva e investigativa, para hacer de ellas entidades más rentables y eficientes económicamente. En consecuencia, la sociedad moderna viene demandando una modificación del rol de las instituciones educativas, entre ellas las universitarias, las que deben diseñar y desarrollar políticas de calidad orientadas a potenciar la preparación general de sus estudiantes e incentivar la actividad investigadora y creativa [1].

En las universidades existen estudiantes con diferentes características, capacidades, intereses, motivaciones y necesidades en el aprendizaje. Unos poseen aptitudes, facultades y habilidades superiores y otros presentan deficiencias y más bajo rendimiento académico. Pero el proceso docente educativo existente en estas instituciones ha priorizado la atención a los más desventajados, lo que ha contribuido a la pérdida del talento universitario en múltiples ocasiones [2].

La comunidad científica internacional cada vez le presta una mayor atención a esta problemática por la importancia que tiene para la obtención de un personal laboral más capacitado entre los estudios realizados están: [3], [4], [5], [6]. En Cuba se han llevado a cabo diversas investigaciones entre ellas [7], [8], [9], orientadas a fundamentar e instrumentar alternativas para atender a estudiantes potencialmente talentosos. Sin embargo las investigaciones cubanas se han dirigido fundamentalmente hacia la enseñanza primaria y media mientras que en el sector universitario el estado actual del estudio del talento revela déficit [10].

En la enseñanza superior cubana, la atención a las diferencias individuales de los estudiantes talentosos no constituye una prioridad ya que las respuestas educativas que proveen los sistemas de educación son uniformes y no estimulan al máximo el desarrollo de todas las potencialidades de estos estudiantes [2]. Siendo las causas principales el desconocimiento de sus características, la falta de información sobre las áreas de formación en las que estos se desenvuelven y fallas en su identificación, orientación y motivación en el contexto educacional [2]. El desarrollo de las habilidades de los estudiantes universitarios talentosos contribuye a formar un potencial humano altamente calificado que impulse la productividad y promueva el desarrollo económico, social y cultural en Cuba, garantizando la formación de profesionales que busquen soluciones a problemas, que mejoren la educación y que impulsen la productividad [10].

La Universidad de las Ciencias Informáticas (UCI) surge en el 2002 con el objetivo de formar profesionales para el apoyo al proceso de informatización de la sociedad cubana. La UCI está dividida en siete facultades, en la Facultad 5 radica el proyecto Talenmático, que tiene como objetivo desarrollar y aplicar un modelo de intervención para la atención educativa a los estudiantes potencialmente talentosos de la universidad.

Para conocer el estado actual de la proyección de la institución hacia los estudiantes potencialmente talentosos y la significación que tiene para la universidad el desarrollo de las habilidades de los mismos, se realizó una encuesta (ver anexo 1) a profesores del proyecto Talenmático. El proyecto está constituido por doce profesores, se realizó una selección de forma aleatoria aplicándole la encuesta a una muestra de ocho profesores que representan el 66,7 % de la población. Los resultados arrojaron que el 100 % de los encuestados consideran que el desarrollo de las habilidades de los estudiantes potencialmente talentosos es necesario en la universidad. Además reflejan los beneficios que traería consigo el desarrollo de las habilidades de estos estudiantes, contribuyendo al incremento de la participación en eventos científicos, al aumento de la calidad de los trabajos investigativos que se presenten y al crecimiento del desarrollo de artículos científicos.

A partir de la encuesta, los 8 profesores plantean que en la UCI no se han sistematizado actividades dirigidas a desarrollar las habilidades de los estudiantes potencialmente talentosos. El 100% considera que las principales deficiencias presentes en la universidad que afectan la atención educativa de los profesores a estos estudiantes son: falta de conocimientos sobre las habilidades, necesidades, motivaciones e intereses de los estudiantes potencialmente talentosos y que el proceso de seguimiento del desarrollo de sus habilidades es difícil de realizar de forma manual y personalizado, debido a la cantidad de indicadores a evaluarles. Además el 87,5 % concuerda que la comunicación y retroalimentación entre profesores y estudiantes potencialmente talentosos es insuficiente, así como la ausencia de una estrategia de gestión del talento informático en correspondencia con los intereses de la universidad.

En la universidad se utiliza un Entorno Virtual de Aprendizaje como apoyo al proceso de formación. El 100 % de los encuestados afirman que esta plataforma no permite definir tareas basadas en indicadores que son necesarios para el seguimiento del desarrollo de las habilidades de los estudiantes potencialmente talentosos.

La situación descrita anteriormente conduce al siguiente **problema a resolver:** ¿Cómo agilizar el proceso de gestión de indicadores de medición del talento para el seguimiento del desarrollo de las habilidades de los estudiantes potencialmente talentosos en la UCI?

Se plantea como **objeto de estudio:** Proceso de formación de estudiantes potencialmente talentosos. Delimitando como **campo de acción:** Sistemas de gestión de indicadores para el seguimiento de habilidades en estudiantes.

Para dar solución al problema de investigación planteado se define como **objetivo general:** Desarrollar un sistema informático que permita agilizar la gestión de indicadores de medición del talento para el seguimiento del desarrollo de las habilidades de los estudiantes potencialmente talentosos de la UCI.

Para darle cumplimiento al objetivo general se definieron los siguientes **objetivos específicos:**

1. Elaborar el marco teórico de la investigación mediante el estudio y el análisis de los principales referentes teóricos acerca de la gestión de indicadores para el seguimiento de habilidades de estudiantes talentosos.
2. Desarrollar un sistema informático que permita la gestión de indicadores de medición del talento para el seguimiento del desarrollo de las habilidades de los estudiantes potencialmente talentosos de la UCI.
3. Validar el objetivo de la investigación mediante la aplicación de técnicas, métricas y pruebas.

Idea a defender:

Si se desarrolla un sistema informático para la gestión de indicadores de medición del talento entonces se agiliza el seguimiento del desarrollo de las habilidades de los estudiantes potencialmente talentosos de la UCI.

En el desarrollo del trabajo de diploma se utilizaron los siguientes **métodos de investigación científica:**

Métodos Teóricos:

- El histórico-lógico: permite conocer los elementos que han caracterizado el proceso de formación de estudiantes potencialmente talentosos, así como los antecedentes y tendencias actuales de sistemas de gestión de indicadores para el seguimiento de habilidades en estudiantes.
- El analítico-sintético facilita el estudio de las fuentes bibliográficas utilizadas en la investigación, con el objetivo de elaborar el marco teórico, identificar los elementos relacionados con la solución propuesta y brindar un análisis de los sistemas relacionados al campo de acción.
- La modelación brinda una abstracción de la solución ya que representa mediante modelos y diagramas los diferentes elementos que la componen.

Métodos Empíricos:

- La encuesta se realiza para obtener información sobre las deficiencias existentes para el desarrollo de habilidades de estudiantes potencialmente talentosos y los elementos a tener en

cuenta en el desarrollo de la solución.

Métodos Estadísticos:

- Análisis porcentual se emplea para el análisis estadístico de la información a procesar de las encuestas a aplicar a los profesores del proyecto Talenmático.

Estructura del trabajo de diploma:

El presente documento está estructurado en: introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas y anexos.

Capítulo 1: Fundamentación teórica

En este capítulo el lector podrá encontrar información sobre los principales conceptos abordados en la investigación. Se realiza un estudio sobre el proceso de formación de estudiantes potencialmente talentosos y sistemas informáticos de gestión de indicadores para el seguimiento de las habilidades de los estudiantes potencialmente talentosos. Además se define la metodología de desarrollo de software a emplear y propone los lenguajes de programación, las herramientas y tecnologías a utilizar en el desarrollo de la propuesta de solución.

Capítulo 2: Propuesta de solución

En el capítulo se presenta la propuesta de solución al problema de la investigación. Se obtiene los productos de trabajo resultantes del desarrollo de las actividades ejecutadas en las disciplinas de Modelado del negocio, Requisitos, Análisis y diseño e Implementación que se definen en la metodología.

Capítulo 3: Validación de la Solución

El capítulo contiene los elementos que forman parte de la validación de la solución. En él se exponen las pruebas realizadas a la solución para validar las funcionalidades del sistema desarrollado, así como los resultados obtenidos en la evaluación del objetivo de la investigación.

CAPÍTULO I: Fundamentación teórica sobre el seguimiento de indicadores para la formación del talento

1.1 Introducción

El contexto educativo diverso en el que se desarrolla la universidad del presente está influenciado por una alta demanda de conocimientos y habilidades tecnológicas. La introducción de las tecnologías de la información y las comunicaciones en el sector educacional ha revolucionado la forma en la que se llevan a cabo los procesos de enseñanza – aprendizaje en las instituciones. La explotación de aplicaciones informáticas como vía de comunicación, transmisión y adquisición de conocimientos y la formación de destrezas en los estudiantes, son componentes esenciales para la formación de profesionales más preparados integralmente [11].

El presente capítulo aborda conceptos relacionados con el proceso de formación de estudiantes potencialmente talentosos. Contiene un estudio sobre la utilización de diversas aplicaciones informáticas en el sector educacional, para la gestión de indicadores de medición de actividades docentes. Realiza una descripción de la metodología de desarrollo de software, así como del lenguaje de modelado para la representación gráfica de los productos de trabajo que formarán parte de la solución. Se definen las herramientas y tecnologías para la implementación de la propuesta de solución.

1.2 Definición de conceptos

1.2.1 Definiciones de talento

El talento es: “una configuración psicológica de la personalidad de naturaleza cognitivo-afectiva, que integra de manera dinámica las capacidades generales y especiales del individuo, con una fuerte energía motivacional manifiesta en los planos intra e interpersonal, expresando un alto nivel de desempeño creativo en un área (s) de interés” [12].

El talento también se puede definir como: “el resultado de la integración funcional de la inteligencia y las capacidades especiales del ser humano con el desarrollo de fuertes intereses en áreas donde se halla profundamente comprometido en lo emocional” [13].

El análisis de las definiciones anteriores permitió definir al talento como: un fenómeno social presente en individuos con capacidades y habilidades cognitivas, emocionales y psicosociales ampliamente desarrolladas.

1.2.2 Definición de talento informático

“Formación psicológica compleja sustentada en una elevada motivación profesional, que le permite al estudiante potencialmente talentoso desarrollar altos niveles de creatividad a partir del compromiso e

interacción con el ambiente social-tecnológico en que se desenvuelve, obteniendo resultados relevantes socialmente válidos en una, o varias áreas de la informática" [14].

En la presente investigación se asume el concepto de talento informático definido por el proyecto Talenmático.

1.2.3 Definición de agilizar

Agilizar: hacer ágil, dar rapidez y facilidad al desarrollo de un proceso o a la realización de algo [15].

1.3 Proceso de formación de estudiantes potencialmente talentosos

Los estudiantes con altas potencialidades y talentos constituyen una parte de la diversidad de estudiantes que asisten a los centros educacionales y requieren de un apoyo y una atención diferenciada, especialmente a garantizar el pleno desarrollo de sus potencialidades. La atención educativa a los estudiantes potencialmente talentosos ha ganado interés a nivel mundial. En Cuba la formación de estudiantes con altas capacidades en el contexto educacional es un campo de investigación e intervención educativa con poca sistematización teórica y metodológica aunque ya se han dado algunos pasos y se han definido estrategias educativas para la atención diferenciada de estos estudiantes [13].

El proceso de formación de estudiantes potencialmente talentosos transcurre por las fases de identificación, estimulación, orientación y gestión de las altas potencialidades, contribuyendo así a la sistematización de su atención educativa en el contexto docente. En el desarrollo de estas fases intervienen un conjunto de indicadores que contribuyen a medir las potencialidades de los educandos. La gestión y el seguimiento de estos indicadores permiten sustentar y orientar la labor educativa hacia el desarrollo de las habilidades de los estudiantes potencialmente talentosos [13].

En Cuba se han publicado artículos y tesis que abordan la formación de talentos universitarios citados anteriormente, definiendo a ese potencial humano formado, capaz de impulsar la productividad y promover el desarrollo económico y social. Este proceso de formación en la UCI actualmente presenta déficit, ya que no se ha sistematizado una atención educativa orientada al trabajo con los estudiantes potencialmente talentosos. En consecuencia, el proyecto Talenmático de la Facultad 5 de la universidad está desarrollando un modelo de intervención para la atención educativa a los estudiantes potencialmente talentosos en informática, que estimule su desempeño académico en correspondencia con sus potencialidades y acorde a las necesidades de la UCI. En la elaboración de este modelo se han realizado las siguientes acciones [16]:

- Valoración de las estrategias existentes para la atención educativa a estudiantes en el nivel superior, especialmente en el área de la informática.

- Valorar la atención educativa que actualmente brinda la UCI, a los estudiantes potencialmente talentosos en informática.
- Elaborar un sistema de gestión de la información para sistematizar y difundir la información científico-técnica existente sobre el tema que permita construir una base científica (teórica y metodológica) que tribute a la realización del trabajo, la capacitación de los docentes y directivos y la búsqueda de apoyo institucional.
- Contextualizar los conceptos de talento, talento informático, y las estrategias para la atención educativa a los estudiantes potencialmente talentosos en el nivel superior, que se emplearán en la investigación.
- Elaborar un sistema de identificación, estimulación y orientación de estudiantes potencialmente talentosos en informática en correspondencia con las necesidades de la UCI.
- Elaborar una estrategia general para la atención educativa a los estudiantes identificados como potencialmente talentos en informática en la UCI.
- Elaborar una estrategia de capacitación a los docentes para la atención educativa a los estudiantes potencialmente talentosos en informática en correspondencia con el modelo del profesional actual que egresa la UCI.

Se han elaborado dos sistemas informáticos orientados al desarrollo de las habilidades de este tipo de estudiantes en el área académica [17], [18] para apoyar el proceso de formación de los estudiantes potencialmente talentosos de la universidad.

Para conocer las características que poseen los sistemas informáticos orientados a la gestión de indicadores que permiten evaluar el proceso de formación de estudiantes se realizó una búsqueda bibliográfica. En el epígrafe 1.4 se relaciona los criterios definidos para el estudio de estos sistemas y los resultados obtenidos.

1.4 Sistemas informáticos para la gestión de indicadores de medición que evalúen el desarrollo de las habilidades de los estudiantes

El proyecto Talenmático está desarrollando una plataforma web para brindarle soporte a todas las aplicaciones informáticas desarrolladas para la atención educativa de estudiantes potencialmente talentosos. En el proceso de búsqueda de sistemas informáticos existentes para la gestión de indicadores que evalúen el desarrollo de habilidades de los estudiantes, se definieron los siguientes criterios: sistemas basados en tecnología web, centralización de la información, definición de indicadores o medidores de evaluación y atención educativa personalizada. Estos criterios permitieron identificar sistemas informáticos con características similares a las necesidades del proyecto Talenmático, permitiendo conocer el objetivo, el ambiente de trabajo y las funcionalidades

que poseen dichos sistemas. A continuación se relacionan los sistemas que por sus características forman parte del estudio.

1.4.1 DocCF, Software de Gestión Escolar

DocCF, Software de Gestión Escolar, es una aplicación de licencia privativa basada en tecnología web, disponible para el sistema operativo Windows. Desarrollada en Colombia por el Grupo CF Developer para sistematizar y automatizar los procesos escolares, académicos y administrativos en las instituciones educativas. Es totalmente configurable ya que se puede configurar valores por defecto a través de parámetros definidos por el usuario en procesos como matrículas, calificaciones, boletines académicos, biblioteca, evaluaciones de docentes, conceptos de cobro, pagos y venta de artículos. Ofrece seguridad y confidencialidad de los datos ante intrusiones no autorizadas. Puede definir diferentes perfiles (tantos como se desee) para docentes, personal administrativo, directivos, padres de familia y alumnos y de este modo restringir o permitir el acceso total o sólo lectura a ciertas funciones del programa [19].

DocCF posee diferentes módulos, de ellos los orientados a la actividad educativa de los estudiantes son: Gestión de docentes y alumnos, Gestión escolar web y Gestión académica. Este último permite la evaluación de los estudiantes, eligiendo un sistema de evaluación entre cinco tipos o personalizándolo de acuerdo a las necesidades y procedimientos de calificación de una institución. Uno de esos módulos es el de evaluación por logros e indicadores, mediante el cual el docente puede definir un conjunto de indicadores para evaluar a los estudiantes [19].

El análisis realizado al sistema DocCF permitió conocer sus características y funcionalidades. Esta aplicación aunque define indicadores para evaluar a los estudiantes no ofrece la posibilidad de diseñar y emitir tareas a los estudiantes así como controlar las soluciones que estos le proveen a dichas tareas, por lo que no se ajusta a las necesidades del cliente. El sistema solo tiene soporte en el sistema operativo Windows y su licencia comercial es privativa.

1.4.2 Portal Colegios Colombia

Portal Colegios Colombia, es una plataforma web de gestión educativa que se realiza a la medida para las escuelas. Su objetivo es facilitar la comunicación de las instituciones educativas con toda la comunidad (docentes, alumnos, padres y personal administrativo). Provee un soporte interactivo para apoyar el proceso educativo. Cuenta con tres módulos principales: Académico, Administrativo y Comunicativo. El módulo Académico tiene como objetivo gestionar la actividad académica de la institución y cuenta con diferentes módulos, entre ellos: Tareas, Consolidación de notas o evaluaciones, Observador del alumno, Boletines dinámicos y configurables en línea y “Logros del curso, indicadores e indicadores por período” [20].

En el módulo Tareas, los profesores pueden colocar tareas a sus estudiantes, acompañadas de documentos, recursos y vínculos a páginas web. Podrá definir si la tarea es calificable o no y especificar el peso de la misma dentro de las notas del período, así como el tipo de tarea. Con el módulo “Logros del curso, indicadores e indicadores por período” el docente tiene la posibilidad de determinar un conjunto de indicadores de gestión académica para evaluar la actividad educativa de sus estudiantes [20].

El sistema antes mencionado a pesar de que evalúa tareas no permite la definición y evaluación a través de indicadores de medición enfocadas al desarrollo de habilidades por lo que no se ajusta a las necesidades del cliente.

1.4.3 Entorno Virtual de Aprendizaje

El Entorno Virtual de Aprendizaje orientado al proceso docente educativo de la UCI, está basado en la plataforma educativa Moodle (Modular Object-Oriented Dynamic Learning Environment, Entorno de Aprendizaje Dinámico Modular Orientado a Objetos). Es una aplicación que pertenece al grupo de los Gestores de Contenidos Educativos (LMS, Learning Management Systems), también conocidos como Entornos de Aprendizaje Virtuales (VLE, Virtual Learning Managements) [21].

Es una aplicación web gratuita que los educadores pueden utilizar para crear sitios de aprendizaje en línea. Se basa en un núcleo del sistema que provee los mecanismos necesarios para que una gran cantidad de conectores o extensiones le brinden las funcionalidades específicas a la plataforma y sus componentes principales son los siguientes: categorías, cursos, usuarios, grupos, permisos y roles [22].

Las categorías son los contenedores de información de más alto nivel, están formadas por cursos, los que contienen una secuencia de actividades y recursos agrupados en secciones. Los usuarios son diferenciados en la aplicación a través de un sistema de roles en función de los siguientes permisos [23]:

- Administrador, puede “hacer todo” en el sitio.
- Profesor editor, puede gestionar y añadir contenidos a los cursos.
- Profesor no editor, puede calificar dentro de los cursos, pero no puede editarlos.
- Estudiante, puede acceder y participar en los cursos e Invitado, puede ver los cursos, pero no participa.

El análisis realizado al EVA utilizado en el proceso de formación en la universidad posibilitó conocer las características y los recursos que este posee, permitiendo concluir que los módulos de actividades didácticas que posee para el soporte de los cursos en este entorno son: Tarea, Cuestionario, Glosario,

Taller, Wiki. Los módulos que poseen características similares a las necesidades del cliente se relacionan a continuación:

- El módulo Tarea: permite la definición de tareas a un grupo de estudiantes sin hacer diferenciaciones individuales y emite una sola evaluación a la respuesta de cada estudiante, impidiendo que el profesor exponga varias evaluaciones relacionadas a diferentes indicadores que se le pudieran medir para el desarrollo de habilidades.
- El módulo Taller: permite definir indicadores pero estos varían para cada taller por lo que no se ajusta al objetivo de la investigación.

Los módulos Tarea y Taller no permiten realizar un seguimiento a los indicadores de medición definidos por el proyecto Talenmático para el desarrollo de las habilidades de los estudiantes potencialmente talentosos de la UCI. Además no cumplen con los elementos necesarios para el diseño, evaluación y control de tareas que respondan a dichos indicadores.

Otros elementos a tener en cuenta detectados en el estudio realizado al EVA son:

- La estructura definida en este entorno para el perfil de los estudiantes no permite que estos definan información relacionada con sus habilidades, intereses y motivaciones.
- El sistema de reportes que posee no cumple con los requisitos del negocio ya que estos están enfocados a: evaluaciones generales, actividades realizadas y fecha y hora del acceso a los recursos disponibles impidiendo la visualización de las evaluaciones por indicadores.

1.4.4 Sistemas informáticos desarrollados en la UCI para el desarrollo de habilidades en estudiantes potencialmente talentosos

En el proyecto Talenmático se ha desarrollado dos aplicaciones informáticas como resultado de dos tesis de grado orientadas al desarrollo de habilidades en estudiantes potencialmente talentosos en el área académica, los mismos se relacionan a continuación.

Sistema informático para la estimulación del aprendizaje del Cálculo Diferencial e Integral en estudiantes potencialmente talentosos

Este sistema consiste en una aplicación web que permite el control del proceso de asignación, resolución y evaluación de problemas orientados a desarrollar el talento en estudiantes de primer año de la UCI. El mismo consume el servicio de autenticación de la universidad, garantizando así la seguridad y la confiabilidad de la información. Permite la gestión de cada uno de los profesores que imparten la asignatura y a su vez están relacionados con el sistema, así como la creación de ejercicios, valoración automática de las respuestas dadas por los estudiantes, generación de un ranking con todas las evaluaciones obtenidas por cada repuesta acertada y la consulta previa o posterior de la documentación referente al tema por parte de los estudiantes contribuyendo a

estimular la autopreparación de los mismos. La aplicación posibilita la conexión con el asistente matemático MatLab para la comprobación de las soluciones dadas por los estudiantes. Brinda la opción de realizar encuestas y un diagnóstico a los estudiantes para el apoyo a los profesores en el proceso de identificar y medir el talento en los estudiantes, de esta manera dar un tratamiento diferenciado a cada uno [18].

Este sistema no abarca todas áreas del contexto docente en las que se desenvuelven los estudiantes potencialmente talentosos de la universidad, solamente contiene ejercicios relacionados con el área académica y no permite definir indicadores para medir el desarrollo de las habilidades de estos estudiantes, por lo que no se ajusta a las necesidades del cliente.

Sistema Web para la Identificación de Potenciales Talentos Informáticos en la Programación Competitiva

El sistema consiste en una aplicación web para gestionar la identificación de potenciales talentos informáticos en los estudiantes de la UCI en la programación competitiva. Realiza la autenticación por roles brindando los servicios en dependencia al tipo de usuario autenticado, esto permite mantener la seguridad de los datos que se manejan. Además propone una evaluación, mediante la utilización de técnicas de la lógica difusa, para definir el rango o nivel del estudiante, el cual se manifiesta en potencialmente talentoso, talentoso y no talentoso [17].

Este sistema está orientado al área académica del proceso de formación de la universidad por lo que no permite realizar un seguimiento de las habilidades que desarrollan los estudiantes potencialmente talentosos en otras áreas de este proceso de formación. Además en el sistema de evaluación que propone para medir los conocimientos de estos estudiantes no permite definir indicadores, por lo que su utilización no responde a las necesidades del cliente.

1.4.5 Análisis realizado a los sistemas informáticos

A partir de los criterios definidos en el epígrafe 1.4 y la revisión bibliográfica realizada a los sistemas anteriores se pudo conocer el estado actual de aplicaciones informáticas que permiten la gestión de indicadores que están orientados al desarrollo del proceso docente - educativo en diferentes centros educacionales. Los resultados de este estudio se resumen en la Tabla 1.

Tabla 1. Resultados del estudio realizado a los sistemas informáticos

Criterios de búsqueda	**Sistemas estudiados**			
	DocCF	Portal Colegios Colombia	EVA	Sistemas desarrollados en la UCI

Sistemas basados en tecnología web	Sí	Sí	Sí	Sí
Centralización de la información	Sí	Sí	Sí	Sí
Definición de indicadores o medidores de evaluación	Sí	No	No	No
Atención educativa personalizada	No	Sí	No	No

El estudio realizado permitió definir un conjunto de aportes y limitaciones, visibles en la Tabla 2, que fueron tomados en cuenta para el diseño de la propuesta de solución.

Tabla 2. Aportes y limitaciones de los sistemas informáticos analizados

Sistemas	**Aportes**	**Limitaciones**
DocCF	Estructura definida para el registro de evaluaciones y los reportes académicos.	No permite diseñar y asignar tareas a los estudiantes así como controlar las soluciones que estos emiten.
Portal Colegios Colombia	Permite la redacción asignación de las tareas a los estudiantes y el control y evaluación de las respuestas. En el diseño de esta tarea los docentes pueden adjuntar documentos, recursos y vínculos a páginas web para complementar las orientaciones contenidas en ella.	No permite evaluar indicadores por tareas.
EVA	El sistema de permisos a los usuarios está orientado a roles. La estructura que define para el diseño de tareas.	No permite definir indicadores de medición para las tareas impidiendo la visualización del progreso individual de los estudiantes. Los reportes que emite no se ajustan a las necesidades del proyecto Talenmático.
Sistemas desarrollados en la UCI	El sistema de autenticación que implementan cumple con las políticas de seguridad.	Desarrollan habilidades solamente en el área académica. No definen indicadores para el

		seguimiento de las habilidades de los estudiantes potencialmente talentosos.

1.5 Metodología de desarrollo de software

Una metodología de desarrollo de software "es un conjunto integrado de técnicas y métodos que permite abordar de forma homogénea y abierta cada una de las actividades del ciclo de vida de un proyecto de desarrollo. Es un proceso de software detallado y completo. Es un marco de trabajo que se usa para estructurar, planificar y controlar el proceso de desarrollo de un sistema informático" [24].
Las metodologías de desarrollo de software "son un conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a los desarrolladores a realizar un nuevo software. Puede seguir uno o varios modelos de ciclo de vida, es decir, el ciclo de vida indica qué es lo que hay que obtener a lo largo del desarrollo del proyecto pero no cómo hacerlo. La metodología indica cómo hay que obtener los distintos productos parciales y finales" [25].

1.5.1 Metodología de desarrollo de software Variación de AUP para la UCI

En el desarrollo de la aplicación se utilizará la metodología de desarrollo de software Variación de AUP para la UCI, es una variante realizada por la Universidad de las Ciencias Informáticas a la metodología ágil AUP y está definida por la universidad como el documento rector de la actividad productiva.

Fases de Variación de AUP para la UCI

La metodología Variación de AUP para la UCI está forma por tres fases, (Inicio, Ejecución y Cierre) para el ciclo de vida de los proyectos de la universidad, las cuales contienen las características de las cuatro fases (Inicio, Elaboración, Construcción y Transición) propuestas en AUP. Las características de las fases de la metodología de la universidad son [26]:

- Inicio: Durante el inicio del proyecto se llevan a cabo las actividades relacionadas con la planeación del proyecto. En esta fase se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y costo y decidir si se ejecuta o no el proyecto.
- Ejecución: En esta fase se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura. Durante el desarrollo se modela el negocio, se obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto. Durante esta fase el software es transferido al ambiente de los usuarios finales o entregado al cliente junto con la documentación. Además, en esta transición se capacita a los usuarios finales sobre la utilización de la aplicación.

- Cierre: En el cierre se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto.

Disciplinas de Variación de AUP para la UCI

AUP propone 7 disciplinas: Modelo, Implementación, Prueba, Despliegue, Gestión de configuración, Gestión de proyecto y Entorno; para el ciclo de vida de los proyectos de la UCI se decidió contar con 8 disciplinas, pero a un nivel más atómico que el definido en AUP, ellas son: Modelado de negocio, Requisitos, Análisis y diseño, Implementación, Pruebas interna, Pruebas de liberación, Pruebas de Aceptación y Despliegue [26]. En la Figura 1 se presenta las fases, disciplinas y principales tareas que conforman la metodología.

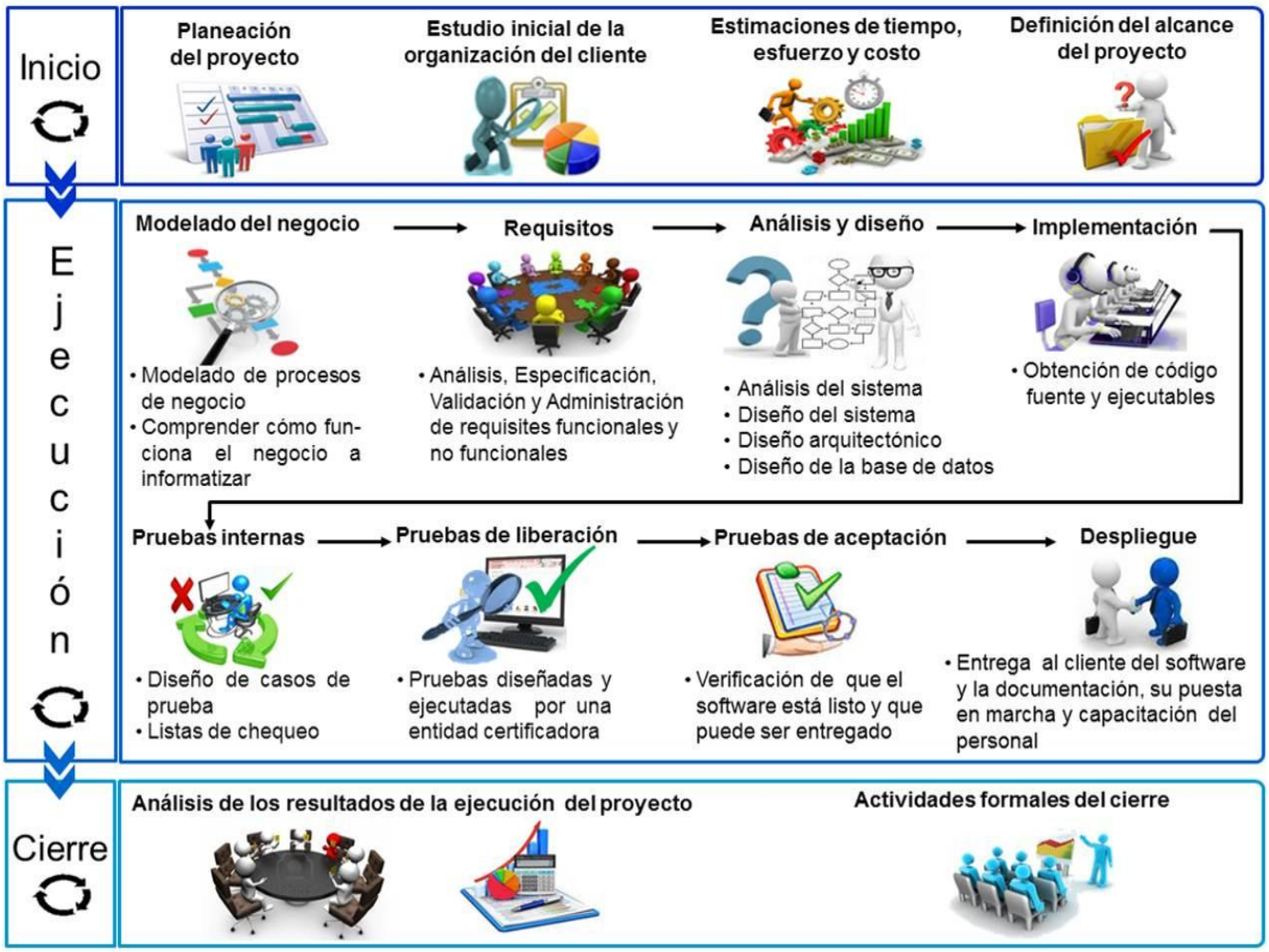

Figura 1. Metodología de desarrollo de software Variación de AUP para la UCI

El desarrollo del sistema de gestión de indicadores de medición para el desarrollo de habilidades de los estudiantes potencialmente talentosos de la UCI, se centrará en la fase de Ejecución de la metodología. Además se generarán los productos de trabajo definidos para las disciplinas Modelado

del negocio, Requisitos, Análisis y diseño, Implementación, Pruebas internas, Pruebas de liberación y de Aceptación.

1.6 Lenguaje de modelado de software

La utilización de los lenguajes de modelado le permite al equipo de desarrollo de software obtener modelos para tener una “representación conceptual o física a escala de un proceso o sistema, con el fin de analizar su naturaleza, desarrollar o comprobar hipótesis o supuestos y permitir una mejor comprensión del fenómeno real al cual el modelo representa y permitir así perfeccionar los diseños, antes de iniciar la construcción de objetos reales” y ayudar a los usuarios a visualizar el producto final [27]. El lenguaje utilizado para el modelado de la aplicación es UML.

1.6.1 Lenguaje Unificado de Modelado 2.0

El Lenguaje Unificado de Modelado (UML, Unified Modeling Language) es un lenguaje de modelado visual que se utiliza para especificar, visualizar, construir y documentar artefactos de un sistema de software. Está pensado para usarse con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios y pretende unificar diferentes técnicas de modelado.

UML incluye conceptos semánticos, notación, y principios generales de un sistema además de que permite representar decisiones de implementación y entorno. Contiene además construcciones organizativas para agrupar los modelos en paquetes, lo que permite dividir grandes sistemas en piezas de trabajo más simples [28].

1.7 Herramientas para el modelado del sistema

La Ingeniería de Software Asistida por Computadoras (CASE, Computer Aided Software Engineering) es la aplicación de tecnología informática a las actividades, las técnicas y las metodologías propias del desarrollo de software, su objetivo es acelerar el proceso de una o más fases del ciclo de vida del desarrollo de sistemas [29].

1.7.1 Visual Paradigm 8.0

Visual Paradigm for UML es una herramienta CASE que soporta el modelado mediante UML y proporciona asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software. Permite integrarse con otras aplicaciones, como herramientas ofimáticas, lo cual aumenta la productividad. Brinda la posibilidad de generar código de forma automática, reduciendo los tiempos de desarrollo y evitando errores en la codificación del software así como obtener diversos informes a partir de la información introducida en la herramienta [30].

Esta herramienta se selecciona para el modelado del sistema por la importancia del uso del software libre en Cuba, la existencia de una licencia otorgada a la universidad para su utilización y las características que posee.

1.8 Lenguajes de programación

Un lenguaje de programación es un lenguaje diseñado para describir el conjunto de acciones consecutivas que un equipo debe ejecutar. Por lo tanto, un lenguaje de programación es un modo práctico para que los seres humanos puedan dar instrucciones a un equipo [31].

Es un sistema estructurado y diseñado principalmente para que las máquinas y computadoras se entiendan entre sí y con los humanos. Contiene un conjunto de acciones consecutivas que el ordenador debe ejecutar [32].

1.8.1 HTML 5

HTML (HyperText Markup Language, Lenguaje de Macado de Hipertextos) es el lenguaje que se utiliza para crear todas las páginas web de Internet. Es “un lenguaje reconocido universalmente y que permite publicar información de forma global”. (W3C, 2015.) “Define una estructura básica y un código HTML para la definición de contenido de una página web, como texto, imágenes, entre otros” y se basa en la referenciación por hipertextos o enlaces entre páginas [33].

HTML 5 no es una nueva versión del antiguo lenguaje de etiquetas, ni siquiera una mejora de lenguaje, sino un nuevo concepto para la construcción de sitios web y aplicaciones donde se combinan las tecnologías HTML, para la estructura y organización de las páginas; CSS, para presentar el contenido a través de diferentes estilos de diseño y Javascript para proveer determinadas funcionalidades útiles para la web [34].

1.8.2 CSS 3.0

CSS (Cascading Style Sheets, Hojas de Estilo en Cascada) es un lenguaje para controlar la presentación de los documentos electrónicos definidos con HTML y XHTML. CSS es la mejor forma de separar los contenidos y su presentación y es imprescindible para la creación de páginas web complejas”. (…) “Mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes” [35].

Se utiliza para definir el aspecto de todos los contenidos, como: el color, tamaño y tipo de letra de los párrafos de texto, la separación entre titulares y párrafos, la tabulación con la que se muestran los elementos de una lista [35]. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos. Funciona a base de reglas para las declaraciones sobre el estilo de uno o más elementos [36].

1.8.3 Javascript 1.6

Javascript es un lenguaje de programación web en el lado del cliente que se utiliza principalmente para crear páginas web dinámicas. Es un lenguaje interpretado, por lo que no es necesario compilar los programas para ejecutarlos, es decir, los programas escritos con Javascript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios [37].

"Está diseñado para el desarrollo de aplicaciones cliente-servidor a través de Internet" [38]. Permite incorporar funciones para el control de la información. Evita cargas en las transferencias de datos entre el cliente y el servidor ya que el navegador de usuario es el responsable de asumir toda la carga de procesamiento [34].

1.8.4 PHP 5.5

PHP (HyperText Preprocessor, Preprocesador de Hipertexto) es un lenguaje de programación de uso general de código abierto y del lado del servidor, resulta muy útil para diseñar de forma rápida y eficaz aplicaciones web dirigidas a bases de datos, permite la generación dinámica de contenidos en un servidor web y puede ser embebido en páginas HTML. Contiene técnicas de programación orientada a objetos y posee bibliotecas con funciones predefinidas. Es un lenguaje multiplataforma con capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad, tiene capacidad de expandir su potencial utilizando la enorme cantidad de módulos, llamados extensiones [39].

1.9 IDE de programación

Un IDE (Integrated Development Environment, Entorno de Desarrollo Integrado), es una aplicación de software compuesto por herramientas informáticas que proporciona servicios integrales para que los programadores de software desarrollen aplicaciones. Están diseñados para maximizar la productividad de los desarrolladores, proporcionando componentes muy unidos con interfaces de usuario similares. Presentan un único programa en el que se lleva a cabo todo el desarrollo de una aplicación y poseen múltiples características para la creación, la modificación, la compilación, implementación y depuración de software. Generalmente se componen de un editor de código, un compilador, un depurador y una interfaz gráfica [40].

1.9.1 NetBeans 8.0

NetBeans es un reconocido entorno de desarrollo integrado, de código y sin restricciones. Fue hecho principalmente para el lenguaje de programación Java, pero soporta otros lenguajes como: PHP, JavaScript, C/C++ y Ruby. Está disponible para Windows, Mac, Linux y Solaris. Ha tenido gran éxito en la comunidad de desarrolladores del software libre ya que brinda una plataforma de aplicación que

permite a los desarrolladores crear aplicaciones web, empresariales, de escritorio y móviles de forma rápida [40].

Esta herramienta se seleccionó para el desarrollo de la propuesta de solución ya que responde a la política de soberanía tecnológica de la universidad.

1.10 Marco de trabajo

Un marco de trabajo del inglés framework es un conjunto de componentes físicos y lógicos estructurados de tal forma que permiten ser reutilizados en el diseño y desarrollo de nuevos sistemas de información. Es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado, a partir de una estructura software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación. Contienen patrones y buenas prácticas que apoyan el desarrollo de un producto y un proceso con calidad [41]. Es una aplicación reusable y semicompleta que puede ser utilizada para producir aplicaciones personalizadas o específicas y proporciona una plantilla extensible para el desarrollo de aplicaciones [42].

1.10.1 Symfony 2

Symfony es un marco de trabajo diseñado para optimizar el desarrollo de las aplicaciones web. Esta plataforma de desarrollo garantiza la separación de responsabilidades, es decir, separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación. Esto permite mantener el código organizado y que la aplicación evolucione fácilmente, evitando mezclar llamadas a la base de datos, etiquetas HTML y el código de lógica del negocio en un mismo archivo [43]. “Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación” [44].

Symfony 2 utiliza el lenguaje de programación PHP. Posee una arquitectura interna completamente desacoplada, lo que permite reemplazar o eliminar fácilmente aquellas partes que no encajan en un proyecto de desarrollo de aplicaciones web [45]. Emplea el patrón de diseño MVC (Modelo – Vista – Controlador) para separar las distintas partes que forman un sistema [46].

1.10.2 Bootstrap 3.0

Bootstrap es un marco de trabajo que posee un conjunto de herramientas de software libre para desarrollar aplicaciones web, aportando principalmente, funcionalidades para diseñar el estilo de una página web, haciendo uso de HTML, CSS y Javascript y proporcionando una homogeneidad en el estilo de estas páginas. Permite desarrollar este tipo aplicaciones con una interfaz adecuada para dispositivos de diversos tamaños de forma sencilla [47]. Contiene varias librerías para el desarrollo de

interfaces de usuario, provee un sistema de cuadrícula ajustable a todo tipo de tamaños de pantalla y es totalmente personalizable [48].

Estas tecnologías se seleccionaron para el desarrollo de la propuesta de solución porque responden a la política de soberanía tecnológica de la universidad.

1.11 Servidor de aplicaciones web

Un servidor de aplicaciones web es un software que gestiona el procesamiento de aplicaciones web a través de toda la infraestructura necesaria para ello. Se basa en la arquitectura cliente-servidor, utilizando el protocolo de comunicaciones HTTP para atender las solicitudes de información de un programa cliente, generalmente un navegador web, obtener la información solicitada y enviarla al programa cliente para su visualización por el usuario [49].

Los servidores de aplicaciones se distinguen por el uso extensivo de contenido dinámico y por su frecuente integración con bases de datos. Dichos servidores gestionan la mayor parte de las funciones de lógica de negocio y acceso a los datos de las aplicaciones, además que funcionan como un intermediario para la seguridad y el mantenimiento de las mismas. Presenta funciones multiproceso para atender a las distintas peticiones, además de enviar solicitudes a diferentes equipos en función de la carga y la disponibilidad [50].**Apache 2.0**

El servidor Apache es un servidor web de tecnología Open Source (código abierto) para uso comercial. Opera en diferentes sistemas operativos y es altamente configurable. Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor ya que puede configurarse para que ejecute un determinado script (archivo de órdenes de procesamiento) cuando ocurra un error en concreto. Posee una alta configurabilidad en la creación y gestión de logs (registro), posibilitando la creación de ficheros de log a medida del administrador, de este modo puedes tener un mayor control sobre lo que sucede en el servidor [51].

1.12 ORM

El ORM (Object-Relational Mapping, Mapeo de Objeto relacional) “es una técnica de programación que permite convertir datos entre el lenguaje de programación orientado a objetos y el sistema de base de datos relacional” [52] utilizada en el desarrollo de una aplicación. Esta técnica posibilita convertir de forma automática los objetos de datos de las aplicaciones orientadas a objeto en registros de datos primitivos de las bases de datos relacionales y viceversa [53].

Brinda a los desarrolladores: rapidez en el desarrollo de aplicaciones ya que la interacción con la base de datos es más simple, reutilización al poderse utilizar los métodos de un objeto de datos desde distintas zonas de la aplicación, incluso desde aplicaciones distintas, seguridad para evitar tipos de ataques como las inyecciones SQL (Structured Query Language, Lenguaje de Consulta

Estructurado) y mantenimiento del código debido a la correcta ordenación de la capa de datos, haciendo que este sea mucho más sencillo [53].

1.12.1 Doctrine 2

Doctrine es un ORM para PHP 5.3.0 (y otras versiones superiores) que proporciona persistencia transparente de objetos PHP. Brinda una capa de abstracción de la base de datos y es una librería muy completa y configurable. Permite generar de forma automática la base de datos basándose en el modelo relacional entre las clases. Una de sus características más importantes es la posibilidad de acceder a la base de datos a través de un lenguaje integrado y propio llamado Doctrine Query Language (DQL, Lenguaje de Consultas Doctrine), permitiendo escribir consultas de una manera sencilla y flexible [54].

1.13 Servidor de base de datos

Los Sistemas de gestión de bases de datos (SGBD) son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Permiten describir los elementos de datos con su estructura, interrelaciones y validaciones. El propósito general de los sistemas de gestión de bases de datos es el de manejar de manera clara, sencilla y ordenada un conjunto de datos que posteriormente se convertirán en información relevante para una organización. Tienen como características principales: la abstracción de la información, la independencia, la consistencia, la seguridad, el manejo de transacciones y el tiempo de respuesta [55].

1.13.1 PostgreSQL 9.1

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, multiusuario, centralizado y de propósito general. Es ampliamente considerado como el sistema gestor de bases de datos de código abierto más avanzado del mundo. Permite el control de concurrencia multiversión, gestión de transacciones y puntos de salvas [56].

1.14 PgAdmin III 1.14.0

PgAdmin III es una aplicación con interfaz gráfica para gestionar bases de datos PostgreSQL, siendo la más completa y popular con licencia de código abierto. Es libremente disponible bajo los términos de la Licencia de PostgreSQL y puede ser redistribuido siempre que los términos de la licencia sean cumplidos. Se puede utilizar en los sistemas operativos Linux, Solaris, Mac OS X y Windows. Soporta todas las características de PostgreSQL y está diseñado para darle respuesta a las necesidades de la mayoría de los usuarios, desde la escritura de consultas simples en SQL (Structured Query Language, Lenguaje de Consulta Estructurado) hasta el desarrollo de bases de datos complejas. Posee un editor SQL con resaltado de sintaxis, un editor de código de la parte del servidor y un agente para lanzar

scripts programados y ofrece una alta seguridad en la transferencia de información entre el cliente y el servidor de base de datos [57].

Estas herramientas se seleccionaron para el desarrollo de la propuesta de solución ya que responden a la política de soberanía tecnológica de la universidad.

1.15 Conclusiones parciales del capítulo

La revisión bibliográfica a los referentes teóricos que sustenta la formación de estudiantes potencialmente talentosos permitió caracterizar los elementos necesarios para el desarrollo del proceso de formación de talentos universitarios, en el que se tiene en cuenta un conjunto de indicadores que permiten medir la evolución de las habilidades desarrolladas en estos estudiantes y enfocar la labor del docente hacia este propósito. El estudio realizado a diferentes sistemas informáticos orientadas a la gestión de indicadores en el contexto educativo, posibilitó conocer que no satisfacen las necesidades existentes en la UCI para el desarrollo de habilidades en estudiantes potencialmente talentosos, por lo que surge la necesidad de desarrollar un sistema informático que permita agilizar la gestión de indicadores de medición de los estudiantes potencialmente talentos. El sistema se desarrollará con tecnologías y herramientas de código abierto garantizando la soberanía tecnológica y se utilizará la metodología Variación de AUP para la UCI para guiar el proceso de desarrollo.

CAPÍTULO II: Descripción de la propuesta de solución

2.1 Introducción

El presente capítulo contiene la información relacionada con el análisis, el diseño y la implementación de la propuesta de solución. En él se evidencia la aplicación práctica del conocimiento científico de la ingeniería de software. Se describen las acciones llevadas a cabo para la construcción de la aplicación, guiadas por la metodología de desarrollo de software seleccionada para el desarrollo de la investigación. Se reflejan las necesidades del cliente, los requisitos que el sistema debe cumplir, se define el comportamiento del sistema así como las restricciones del diseño, concebidas para la construcción de la aplicación y los estándares de codificación a seguir durante la implementación de sus funcionalidades. El nombre que recibe el sistema a desarrollar es SIAGEPTI, Sistema Informático de Apoyo a la Gestión de Estudiantes Potencialmente Talentosos en Informática.

2.2 Modelado del negocio

La modelación del negocio consiste en comprender los procesos de negocio de una organización que se desean informatizar, con el objetivo de garantizar que el software desarrollado va a responder a las necesidades del cliente. En este proceso el equipo de desarrollo de software debe tener un contacto directo con el cliente y se puede auxiliar de su documentación como: la misión, visión, objetivos, políticas, estrategias de la entidad, sus procesos de negocio, su estructura, organización [58].

Para comprender el contexto del negocio a informatizar en la aplicación SIAGEPTI, el autor de la tesis elaboró un modelo conceptual, en el que plasmó los conceptos, sus características y las relaciones existentes entre ellos.

2.2.1 Modelo conceptual

El Modelo conceptual es una representación visual de los objetos y conceptos (o partes esenciales) que se requieren en la modelación de un problema determinado y las relaciones que subsisten entre ellos [59]. En la Figura 2 se muestra el Modelo conceptual que representa los conceptos del contexto del negocio referente a la gestión de indicadores de medición para el desarrollo de habilidades de los estudiantes potencialmente talentosos.

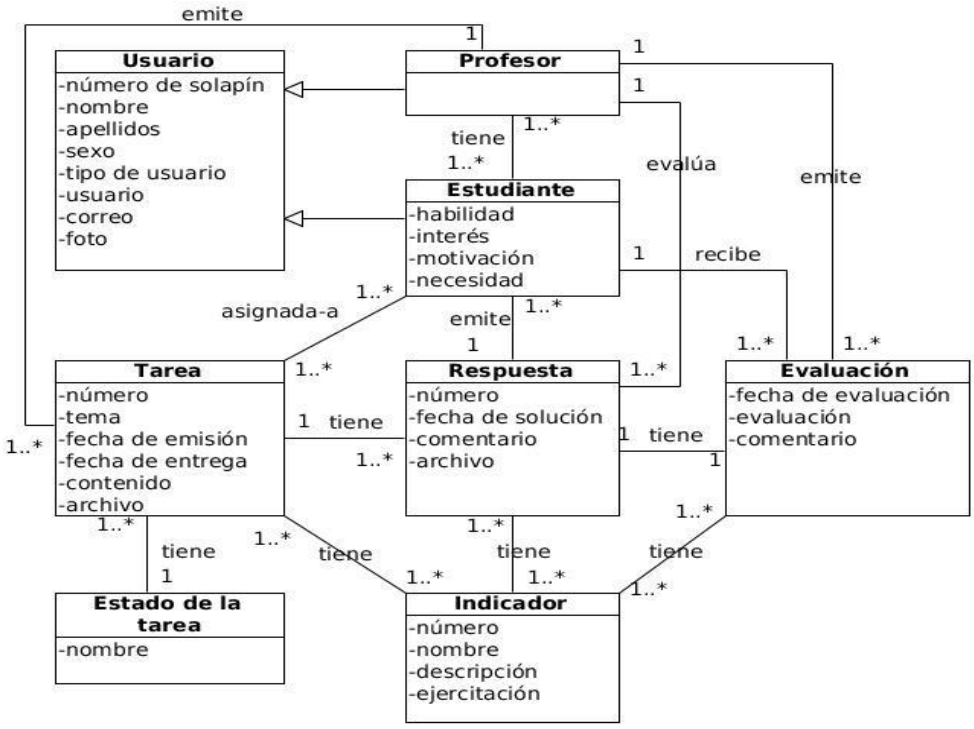

Figura 2. Modelo conceptual para la gestión de indicadores de medición

Los conceptos del contexto del negocio anteriormente modelado son los siguientes:

- Estado de la tarea: contiene los estados por el que puede transitar una tarea.
- Estudiante: es un tipo de usuario que tiene información propia.
- Evaluación de tarea: es la evaluación emitida por un profesor al revisar una respuesta dada por un estudiante.
- Indicador: son los indicadores que se utilizan para medir el desarrollo de las habilidades de los estudiantes potencialmente talentosos
- Profesor: es un tipo de usuario.
- Reporte: contiene las evaluaciones obtenidas por un estudiante
- Tarea: contiene la información de las tareas que emite los profesores y que son resueltas por los estudiantes.
- Usuario: contiene la información general de un los usuarios.

Para comprender el contexto del negocio se elaboró el modelo conceptual, en el que se definen los conceptos asociados a la gestión de indicadores de medición para el desarrollo de habilidades de los estudiantes potencialmente talentosos, así como las características de estos conceptos y las relaciones existentes entre ellos. El modelo conceptual elaborado se utiliza como base para la definición de las funcionalidades y restricciones que debe cumplir el sistema. Estos requisitos

indispensables para la satisfacción de las necesidades del proyecto Talenmático serán abordados en la disciplina Requisitos de la metodología de desarrollo.

2.3 Requisitos

El esfuerzo principal en la disciplina Requisitos es desarrollar un modelo del sistema que se va a construir y comprende la administración de los requisitos funcionales y no funcionales del producto [26]. Permite mejorar el entendimiento, análisis, validación, especificación y gestión de las necesidades, funcionalidades y restricciones que debe cumplir un sistema [60]. A continuación se presenta las actividades desarrolladas en esta disciplina así como los productos de trabajo elaborados.

2.3.1 Fuentes para la obtención de requisitos y técnicas de identificación de requisitos

Las fuentes de obtención de requisitos a consultar para la definición de los requisitos del sistema se relacionan a continuación:

- profesores del proyecto Talenmático
- soluciones informáticas existentes con propósitos similares a los deseados
- Modelo conceptual

Técnicas de identificación de requisitos

Las técnicas de identificación de requisitos de software permiten identificar las necesidades de negocio de clientes y usuarios. Son mecanismos que se utilizan para recolectar la información necesaria en la obtención de los requisitos funcionales y no funcionales de una aplicación [61].

Existen diferentes técnicas de identificación de requisitos dentro de las que se encuentran: entrevistas, encuestas, escenarios, tormentas de ideas, etnografía y construcción de prototipos [62].

- **Entrevista**: es una técnica ampliamente utilizada para la obtención de requisitos. Se basa en la acción de desarrollar una conversación con los clientes con el objetivo de obtener una comprensión general de lo que hace el cliente. Las entrevistas pueden ser de dos tipos: cerradas, donde los entrevistados responden a un conjunto predeterminado de preguntas y abiertas, donde no hay un programa de preguntas predefinido [62].
- **Encuesta:** es una de las técnicas más comunes para el levantamiento de requisitos, consiste en realizar una serie de preguntas a cada uno de los encargados de los procesos de negocio que se desarrollan en una entidad para determinar las necesidades de cada área de trabajo y de la entidad en general. Estas están catalogadas en: estructuradas, que consiste en la elaboración de un mismo tipo de preguntas para todos los encargados de los procesos de negocio y no estructuradas, la cual es realizada sin una guía patrón o lista de preguntas para la obtención de la información [63].

- **Escenarios:** consiste en describir generalmente de forma informal las características del sistema a desarrollar mediante una secuencia de pasos que contiene la interacción del cliente con el sistema desde su perspectiva [62].
- **Tormenta de ideas:** consiste en reunir a los diferentes encargados de los procesos de negocio para que en una reunión informal generen ideas sobre los diferentes problemas que cada uno en su área posea. Como resultado de estas sesiones se obtiene la mayor cantidad de soluciones posibles desde el punto de vista de cada uno de los involucrados en los procesos de la organización y puede generar salidas que beneficien a la organización [63].
- **Etnografía:** es una técnica de observación que se utiliza para entender los requerimientos sociales y organizacionales. Consiste en estudiar el entorno laboral donde se utilizará el sistema a desarrollar [62].
- **Construcción de prototipos:** los prototipos suelen consistir en vistas reducidas de la aplicación a desarrollar. Permite que el usuario cuente con una visión general de lo que desea y que puedan mejorar las especificaciones de los requerimientos [62].

En la identificación de los requisitos del sistema se utilizó la encuesta y la tormenta de ideas. Para ello el autor del trabajo de diploma asistió a tres reuniones del proyecto Talenmático. En la primera reunión se realizó una tormenta de ideas para comprender de forma general las necesidades del cliente y el contexto en el que se desarrolla el problema a solucionar. En una segunda reunión se aplicó una encuesta, cuyos resultados permitieron definir las características de la problemática existente, las posibles funcionalidades a tener en cuenta en el desarrollo de la solución y los beneficios que aportaría el sistema propuesto. En la tercera reunión efectuada de aplicó una tormenta de ideas donde se explicó la idea de la propuesta de solución a informatizar sometiéndose al juicio de los profesores del proyecto con el objetivo de definir las funcionalidades a implementar. En el epígrafe 2.3.2 se relaciona los requisitos funcionales y no funcionales definidos para el sistema informático a desarrollar.

2.3.2 Especificación de requisitos de software

La especificación de requisitos de software tiene como objetivo principal servir como medio de comunicación entre clientes, usuarios y equipo de desarrollo. Contiene un conjunto de información que ayuda a los desarrolladores de software a analizar y entender todos los requisitos que el cliente desea así como los requisitos que debe cumplir el sistema software a desarrollar para satisfacer dichas necesidades [64].

Requisitos funcionales

Los requisitos funcionales de un software describen lo que este debe hacer. Dependen del tipo de software que se desarrolle, de los posibles usuarios del software y de las necesidades que posea el

cliente. Son las declaraciones de los servicios que el sistema debe proporcionar, de la manera en que debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones determinadas [62]. La Tabla 3 muestra los requisitos funcionales definidos para el desarrollo de la aplicación que se propone y se han especificado a partir de las características y funcionalidades definidas con el cliente.

Tabla 3. Listado de los requisitos funcionales de la propuesta de solución

Agrupación de requisitos	RF	Nombre	Complejidad
Gestionar usuario	RF. 1	Adicionar usuario	Baja
	RF. 2	Modificar usuario	Baja
	RF. 3	Eliminar usuario	Baja
	RF. 4	Buscar usuario	Media
	RF. 5	Listar usuarios	Baja
Gestionar indicador	RF. 6	Adicionar indicador	Baja
	RF. 7	Modificar indicador	Baja
	RF. 8	Eliminar indicador	Baja
	RF. 9	Buscar indicador	Media
	RF. 10	Listar indicador	Baja
Gestionar tarea	RF. 11	Adicionar tarea	Media
	RF. 12	Modificar tarea	Media
	RF. 13	Eliminar tarea	Media
	RF. 14	Consultar tarea	Baja
	RF. 15	Buscar tarea	Media
	RF. 16	Listar tarea	Media
Gestionar evaluación	RF. 17	Adicionar evaluación	Media
	RF. 18	Modificar evaluación	Media
	RF. 19	Eliminar evaluación	Media
	RF. 20	Consultar evaluación	Media
	RF. 21	Buscar evaluación	Media
Mostrar respuestas de estudiantes	RF. 22	Listar respuestas de estudiantes	Media
	RF. 23	Consultar respuesta de estudiante	Baja
Gestionar estudiante	RF. 24	Adicionar estudiante al registro del profesor	Baja
	RF. 25	Eliminar estudiante del registro del profesor	Baja
	RF. 26	Listar estudiantes del profesor	Media
	RF. 27	Buscar estudiantes	Media
Gestionar reporte	RF. 28	Listar evaluaciones de estudiantes por indicadores	Alta
	RF. 29	Buscar evaluaciones de estudiante por indicadores	Media

	RF. 30	Consultar evaluaciones de estudiante por indicadores	Baja
Gestionar respuesta	RF. 31	Adicionar respuesta	Media
	RF. 32	Modificar respuesta	Media
	RF. 33	Eliminar respuesta	Media
	RF. 34	Consultar respuesta	Baja
	RF. 35	Buscar respuesta	Media
	RF. 36	Listar respuestas	Media
Mostrar tareas de estudiantes	RF. 37	Listar tareas del estudiante	Media
	RF. 38	Consultar tarea del estudiante	Baja
Gestionar perfil	RF. 39	Consultar perfil	Baja
	RF. 40	Actualizar perfil	Baja
Mostrar perfiles de estudiantes	RF. 41	Listar perfiles de estudiante	Media
	RF. 42	Consultar perfil de estudiante	Baja
Archivo	RF. 43	Cargar archivo de la tarea	Baja
	RF. 44	Abrir archivo de la tarea	Baja
	RF. 45	Cargar archivo de respuesta	Baja
	RF. 46	Abrir archivo de respuesta	Baja
Notificaciones	RF. 47	Notificar asignación de tarea	Baja
	RF. 48	Notificar respuesta	Baja
	RF. 49	Notificar evaluación de la tarea	Baja
Listar evaluaciones de estudiante	RF. 50	Listar evaluaciones por indicadores del estudiante	Alta
Autenticar usuario	RF. 51	Autenticar usuario	Baja

Requisitos no funcionales

Los requisitos no funcionales son restricciones de las funciones o servicios ofrecidos por el sistema. No se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de este. Definen restricciones sobre el sistema y el proceso de desarrollo de software [62]. Los requisitos no funcionales definidos para el desarrollo de la aplicación propuesta se listan a continuación:

RNF. 1 Usabilidad

RNF. 1.1 El idioma de todas las interfaces de la aplicación será español. Los errores cometidos por el usuario les serán notificados. El sistema expondrá el menú general desde cualquiera de sus páginas.

RNF. 2 Confiabilidad

RNF. 2.1 El sistema estará disponible durante 24 horas, los 7 días de la semana, los 365 días del año.

RNF. 2.2 Ante el fallo de una funcionalidad del sistema, el resto de las funcionalidades que no dependen de esta deberán seguir funcionando.

RNF. 2.3 El sistema impondrá campos obligatorios para garantizar la integridad de la información que se introduce por el usuario y no permitirá la entrada de datos incorrectos.

RNF. 3 Mantenibilidad

RNF. 3.1 La codificación del sistema será estándar y las funcionalidades serán comentadas. Además el sistema debe permitir adicionar o modificar funcionalidades.

RNF. 4 Portabilidad

RNF. 4.1 El sistema deberá adaptarse al entorno operativo siempre y cuando este cumpla con las características de la aplicación y debe coexistir con otros sistemas sin dificultad.

RNF.5 Hardware

RNF. 5.1 Las estaciones de trabajo deberán poseer como mínimo un procesador Pentium IV a 2.20 GHZ, una RAM de 512 MB (recomendado 1024 GB) y una tarjeta de red.

RNF. 5.2 El servidor de aplicaciones deberá poseer como mínimo un Core2 2.33 GHz, una RAM de 4 GB, disco duro 250 GB, una tarjeta de red y una fuente de alimentación de corriente.

RNF. 5.3 El servidor de base de datos deberá poseer como mínimo un Core2 2.33 GHz, una RAM de 4 GB, disco duro 500 GB, una tarjeta de red y una fuente de alimentación de corriente.

RNF.6 Software

RNF. 6.1 Las estaciones de trabajo deberán poseer el navegador web Mozilla Firefox 34.0 o una versión superior y el sistema operativo Linux.

RNF. 6.2 El servidor de aplicaciones web deberá ser Apache 2.0 y el sistema operativo Linux.

RNF. 6.3 El servidor de base de datos deberá ser PostgreSQL 9.1 y el sistema operativo Linux.

RNF.7 Rendimiento

RNF. 7.1 El sistema en las operaciones de modificación o búsqueda de datos, deberá tener respuestas a peticiones del cliente en un tiempo máximo de 5 segundos (esta cifra no incluye los retardos por concepto de tráfico de red).

Descripción de requisitos

La descripción de requisitos describe con mayor detalle lo que el requisito debe proporcionar en el sistema. Contiene los pasos o las actividades que se deberán realizar para darle cumplimiento al requisito, así como las restricciones a las que está sujeto y las posibles respuestas del sistema. Es una herramienta ampliamente utilizada en el análisis y diseño de un sistema informático [62]. A continuación se muestra la descripción del requisito funcional Adicionar tarea.

Precondiciones	Debe existir al menos un profesor para poder adicionar una tarea. Debe existir al menos un estudiante para asignarle la tarea. Debe existir al menos un indicador.
Flujo de eventos	
Flujo básico Adicionar tarea.	
1.	El profesor selecciona Adicionar en el menú de la página Gestionar Tareas.
2.	El sistema muestra el formulario Adicionar tarea y carga automáticamente el siguiente atributo del formulario: Estado
3.	El profesor introduce los datos de los siguientes campos que aparecen en el formulario: Número Archivo Fecha de entrega Tema Contenido Indicadores Estudiantes
4.	El profesor selecciona la opción Aceptar.
5.	El sistema valida (ver validación 1) los datos introducidos.
Pos-condiciones	
1.	Se ha adicionado una nueva tarea. La tarea se encuentra en estado de "Creada".
Flujos alternativos	
Flujo alternativo 5.a Datos incorrecto en el formulario.	
1	El sistema informa sobre la existencia de errores.
2	El profesor introduce los datos.
3	Volver al paso 4 del flujo básico.
Pos-condiciones	
1	N/A
Flujo alternativo *.a El usuario cancela la acción.	
1	El sistema cierra los formularios
2	Concluye el requisito.
Pos-condiciones	
1	N/A
Validaciones	
1	Se validan los datos según lo establecido en el Modelo conceptual.

Relaciones		**Requisitos Incluidos**	N/A
		Extensiones	N/A
Conceptos	**Tarea**	**Visibles en la interfaz:** Número Estado Archivo Fecha de entrega Tema Contenido Indicadores Estudiantes **Utilizados internamente:** id profesor fechaEmision	
Requisitos especiales	N/A		
Asuntos pendientes	N/A		

Figura 3. Descripción del requisito funcional Adicionar tarea

La descripción de los requisitos que componen la aplicación permitió definir los flujos de trabajo que se deben seguir, así como a las restricciones que deben cumplir y las respuestas que debe dar el sistema para cada uno de los pasos definidos en las funcionalidades.

En la Figura 4 se muestra el prototipo de interfaz de usuario del requisito funcional Adicionar tarea.

Figura 4. Prototipo del requisito funcional Adicionar tarea

2.3.3 Validación de requisitos

El objetivo principal de la validación requisitos es comprobar que los requisitos funcionales y no funcionales definidos para el sistema se corresponden con las necesidades del negocio del cliente y usuarios, obteniendo su aprobación y permitiendo generar una línea base de los requisitos en la actividad [65]. Existen diferentes técnicas para la validación de requisitos de software como: Técnica de prototipado, Diseño de casos de prueba y Revisiones Técnicas Formales.

- **Técnica de prototipado:** el prototipado de interfaz de usuario es una técnica de representación aproximada de la interfaz de usuario de un sistema software que permite a clientes y usuarios entender fácilmente la propuesta de solución para resolver sus problemas de negocio [65].

- **Diseño de casos de prueba:** el objetivo principal del diseño de casos de prueba es obtener un conjunto de pruebas que tengan la mayor probabilidad de descubrir los defectos del software [61].
- **Revisiones Técnicas Formales:** es el filtro más efectivo desde el punto de vista de garantía de calidad. El objetivo fundamental de estas revisiones es encontrar errores durante el proceso de desarrollo de software. Se deben realizar desde el principio del proyecto para evitar la propagación de los errores a las etapas siguientes del proceso de software [61].

La validación de los requisitos de software de la propuesta de solución se desarrolló a través de las técnicas de prototipado y diseño de casos de prueba. En el proceso de validación de los requisitos a se diseñaron los prototipos de interfaces y se presentaron a diferentes profesores del proyecto Talenmático para su valoración. Esto permitió que el cliente tuviera una noción de la propuesta de solución y se pudieran hacer las correcciones necesarias antes del diseño e implementación de las funcionalidades del sistema. La generación de casos de prueba permitió especificar los posibles datos de entrada así como las salidas del sistema generándose un diseño de caso de prueba para cada requisito funcional.

2.3.4 Administración de requisitos

La administración de requisitos es el proceso de comprender y controlar los cambios en los requisitos del sistema. Incluye las actividades de planificación de la administración de requisitos, en la que se identifican los requisitos cambiantes y los no cambiantes, se establecen políticas y procedimientos para el seguimiento de la relación entre los requisitos y de estos con el sistema. Además abarca la gestión de cambio en la que se analiza los cambios propuestos en los requisitos y se evalúa el impacto que provoca [62].

En la propuesta de solución se utilizaron las matrices de trazabilidad para la administración de los requisitos del sistema desarrollado, permitiendo seguir la evolución de los requisitos durante el ciclo de vida del proyecto. Para ello se definieron los siguientes elementos de seguimiento para la trazabilidad de requisitos:

- Modelo conceptual
- Requisitos
- Modelo de datos
- Diagrama de clases del diseño
- Diagrama de componentes
- Diseños de casos de prueba

Las matrices de trazabilidad de los requisitos funcionales del sistema realizadas son las siguientes: Requisitos - Modelo conceptual, Requisitos - Modelo de datos, Requisitos - Diagrama de clases del diseño, Requisitos - Diagrama de componentes y Requisitos - Diseños de casos de prueba. En la Figura 5 se muestra un fragmento de la Matriz de trazabilidad Requisitos - Modelo conceptual, de las restantes se presenta un fragmento en los anexos 2, 3, 4 y 5.

(51) Requirement / By: Transitor / (8) Class	Estado de la tarea	Estudiante	Evaluación	Indicador	Profesor	Respuesta	Tarea	Usuario
Abrir archivo de la tarea							✓	
Abrir archivo de respuesta						✓		
Actualizar perfil		✓						
Adicionar estudiante al registro del profesor		✓			✓			
Adicionar evaluación			✓	✓		✓		
Adicionar indicador				✓				
Adicionar respuesta		✓				✓	✓	
Adicionar tarea	✓	✓		✓			✓	
Adicionar usuario		✓			✓			✓
Buscar estudiantes		✓						✓
Buscar evaluaciones de estudiante por indicadores			✓	✓				
Buscar evaluación			✓					
Buscar indicador				✓				
Buscar respuesta						✓		
Buscar tarea							✓	
Buscar usuario								✓
Cargar archivo de la tarea							✓	
Cargar archivo de respuesta						✓		

Figura 5. Fragmento de la Matriz de trazabilidad Requisitos – Modelo conceptual

2.4 Análisis y diseño

En esta disciplina los requisitos pueden ser refinados y estructurados para conseguir una comprensión más precisa de estos, y una descripción que sea fácil de mantener y ayude a la estructuración del sistema [26].

2.4.1 Diseño arquitectónico

La arquitectura de software de un sistema se refiere a sus diferentes componentes, las interacciones entre ellos y la configuración del sistema. El diseño arquitectónico es una actividad que permite estructurar de forma general al sistema y su objetivo es tener una visión clara del software a construir. En él se identifican los subsistemas en los que se divide una aplicación y se establece un marco de control y comunicación entre ellos [66]. En el diseño arquitectónico del sistema propuesto se utiliza el patrón Modelo – Vista – Controlador (MVC), el mismo se describe a continuación.

Patrón arquitectónico MVC

Los patrones arquitectónicos son patrones de software que ofrecen soluciones a problemas de arquitectura de software, adaptabilidad a requerimientos cambiantes, rendimiento, modularidad y acoplamiento. Especifican un conjunto predefinido de subsistemas con sus responsabilidades y una serie de recomendaciones para organizar los distintos componentes de un sistema. La solución que plantean es la creación de patrones de llamadas entre objetos y el empaquetado de funcionalidades [67].

El patrón MVC utilizado para definir la arquitectura del sistema separa los datos de la aplicación (modelo), la interfaz de usuario (vista), y la lógica de negocio (controlador) en tres componentes distintos. En la Figura 6 se presenta la estructura de separación que propone el patrón.

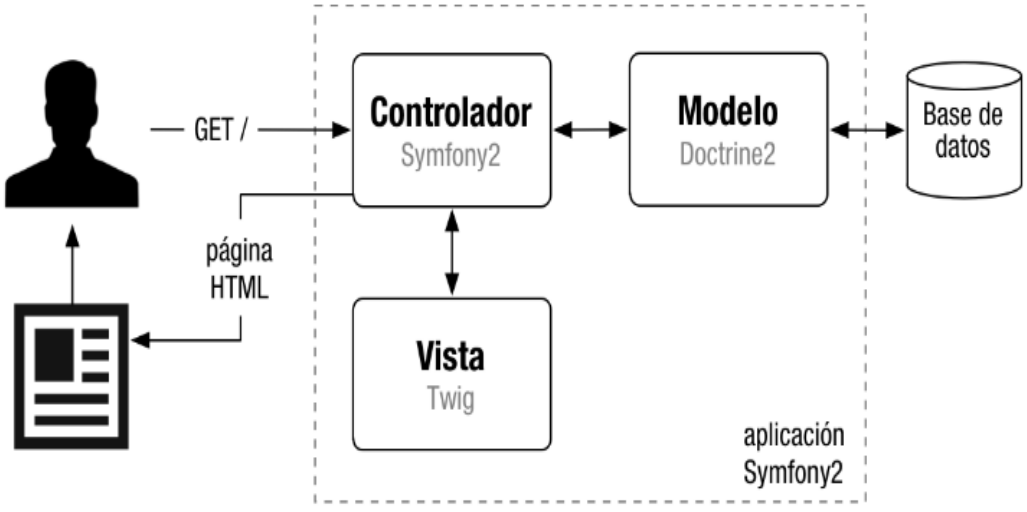

Figura 6. Patrón arquitectónico MVC (Fuente: [45])

La Figura 7 representa la separación de las clases de la aplicación según el marco de trabajo Symfony 2. En ella se visualiza la utilización del patrón MVC ya que en la carpeta *Controller* se encuentran las clases controladoras, en la carpeta *Entity* las clases del modelo y en la carpeta *views* las clases que forman parte de la vista de la aplicación.

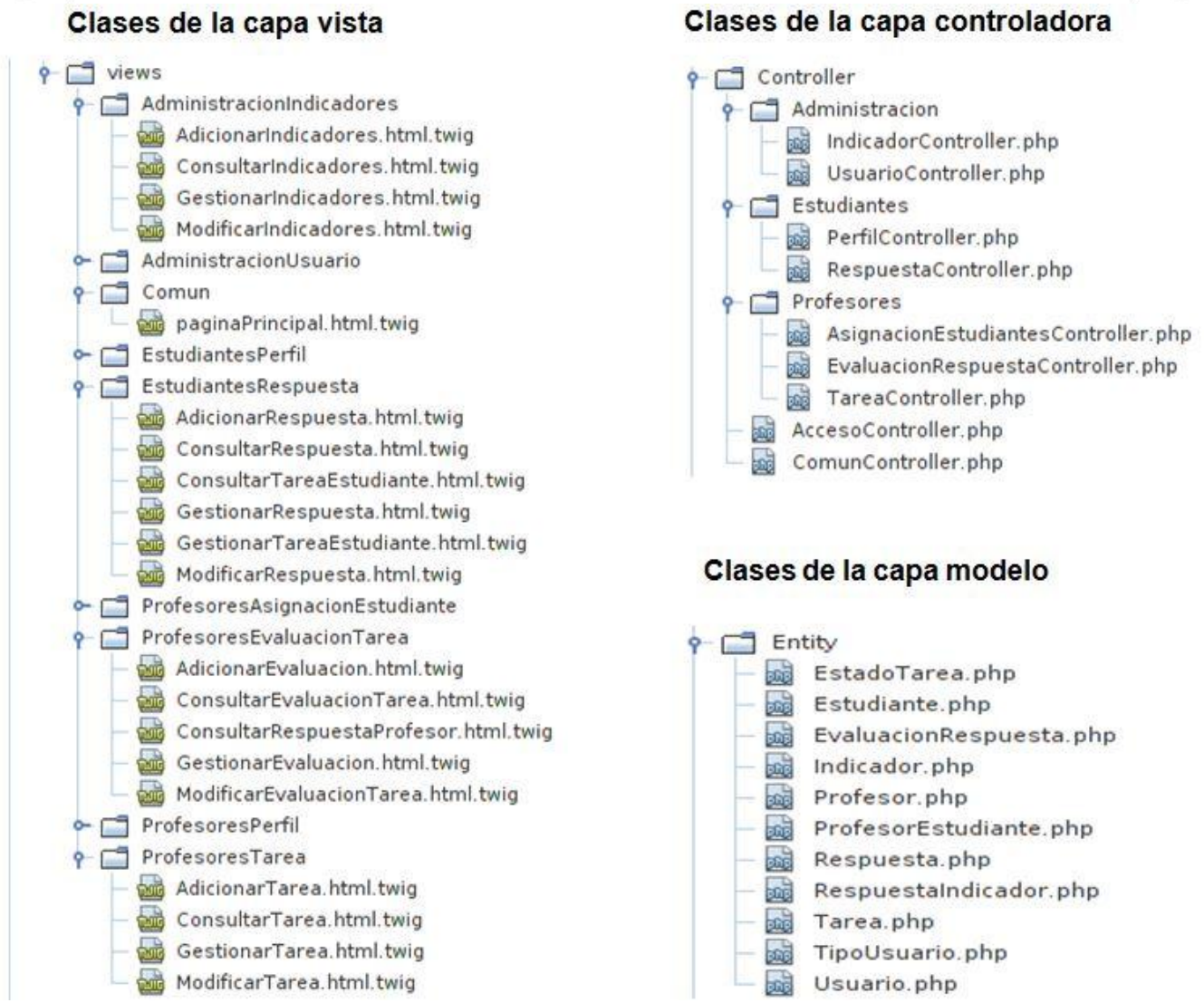

Figura 7. Aplicación del patrón MVC en el SIAGEPTI

2.4.2 Modelo de datos

Un modelo de datos describe los datos que apoyan los procesos de una organización e ilustra gráficamente las personas, lugares o cosas acerca de la información capturada y cómo se relaciona entre sí. Refleja con exactitud cómo serán guardados los datos en la base de datos [68].

La Figura 8 muestra el modelo de datos del SIAGEPTI. Contiene cada una de las tablas de la base de datos de la aplicación, así como sus atributos y relaciones. El modelo se encuentra normalizado en Tercera Forma Normal, de las seis formas normales existentes para la normalización de las bases de datos relacionales, por lo que todos los atributos de las tablas del modelo, dependen únicamente de la llave primaria.

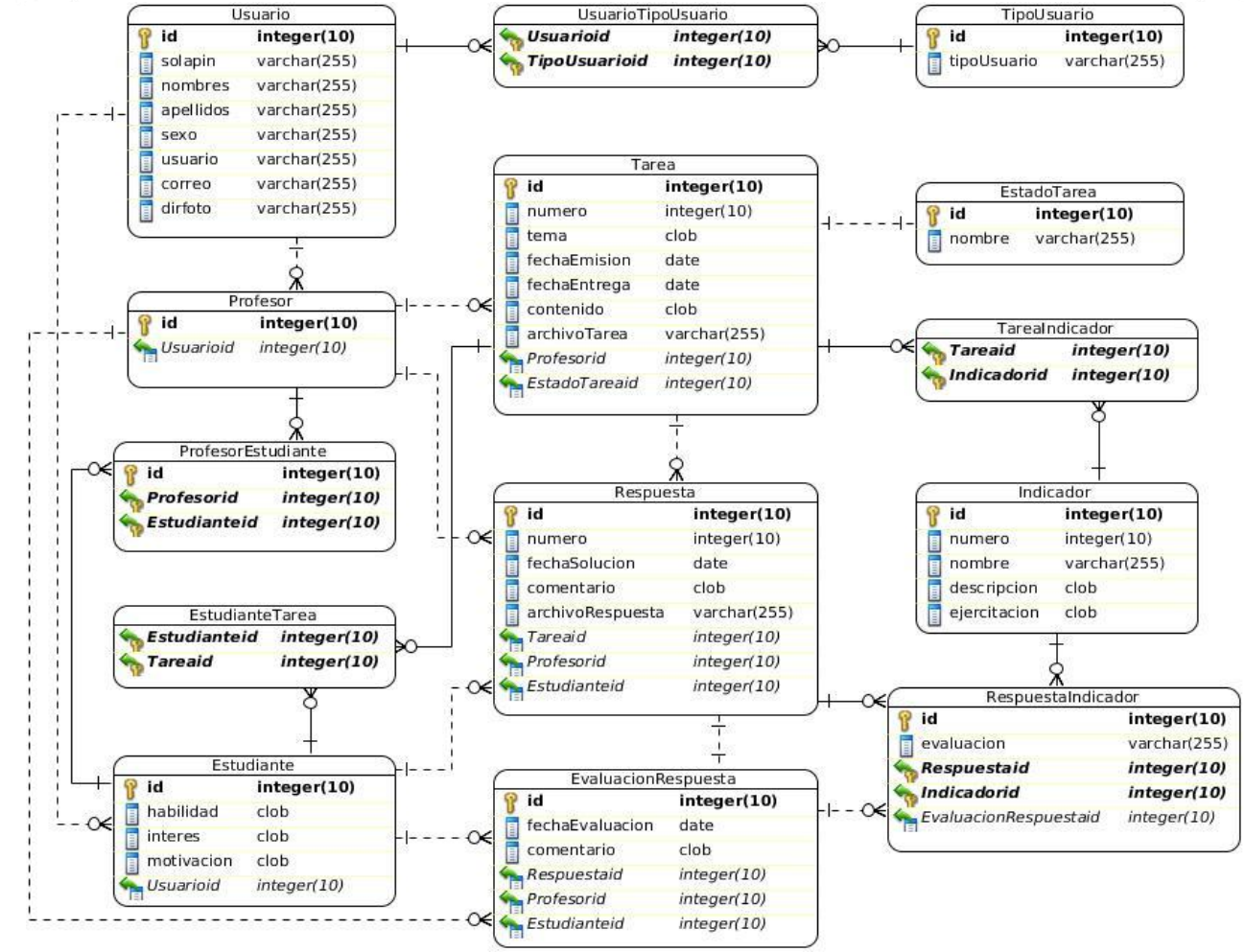

Figura 8. Modelo de datos del SIAGEPTI

2.4.3 Modelado del diseño

El diseño es la forma exacta en la que un requisito del cliente se puede convertir en un sistema o producto de software terminado. Crea una representación o modelo del software de forma detallada sobre la estructura de datos, la arquitectura, las interfaces y los componentes del software que son necesarios para implementar el sistema e influye directamente en la calidad del producto informático [61].

Diagrama de clases del diseño

Un diagrama de clases es un tipo de diagrama estático que describe la estructura de un sistema mostrando sus clases, atributos y las relaciones entre ellos [69]. El diagrama de clases del diseño es el resultado de un refinamiento del diagrama de clases del análisis, y muestra la estructura del sistema a un nivel de detalles de diseño que permitirá la implementación de dichas clases [61]. La Figura 9 muestra un fragmento del diagrama de clases del diseño de la agrupación de requisitos Adicionar tarea.

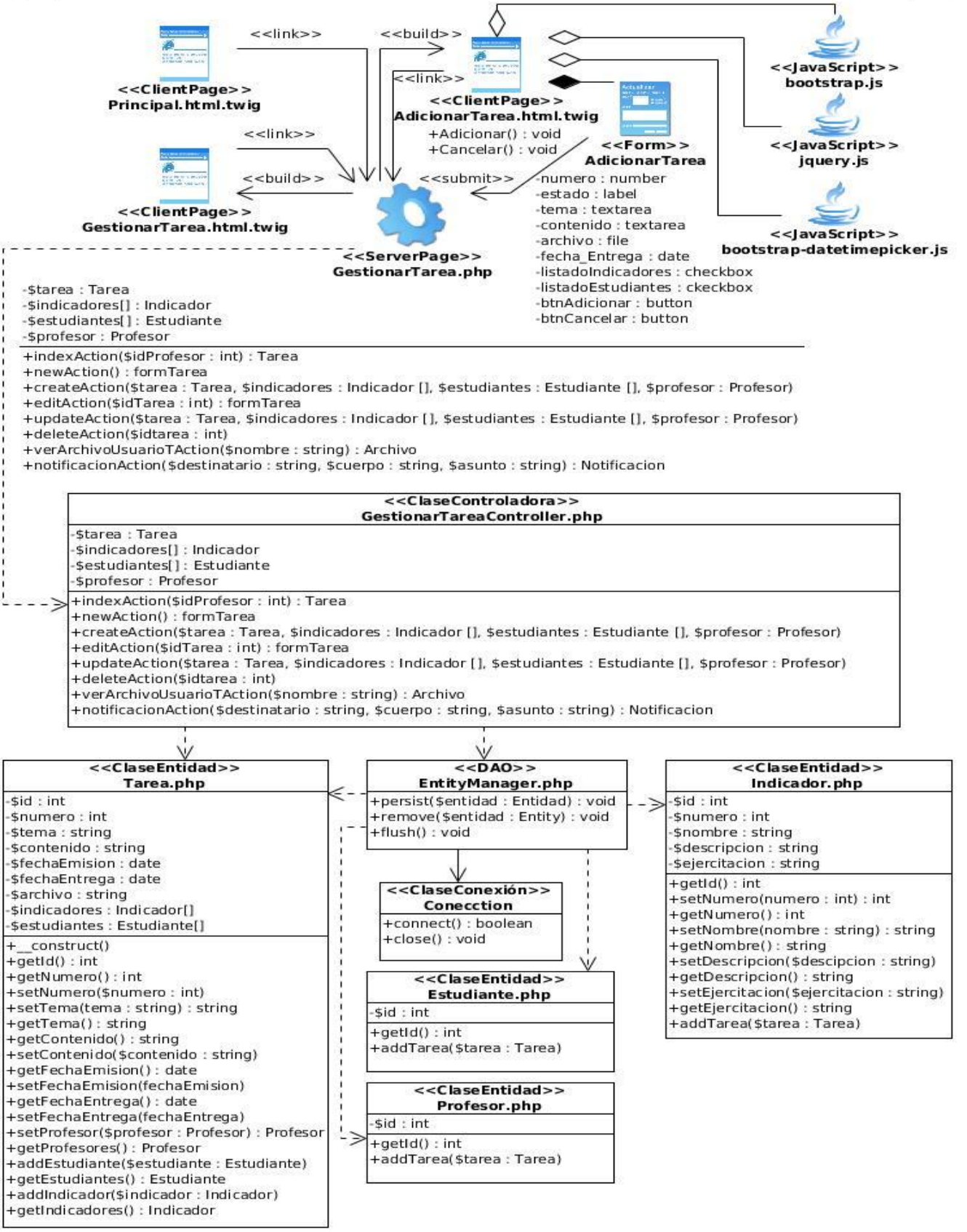

Figura 9. Fragmento del Diagrama de clases del diseño de la agrupación Adicionar tarea

El diagrama de clases del diseño contiene diferentes tipos de clases que lo componen y las relaciones existentes entre ellas:

- Clases de interfaz: definen todas las abstracciones necesarias para la interacción entre los usuarios y la aplicación. Están formadas por las páginas clientes: *Pincipal.html.twig*, *GestionarTarea.html.twig* y *AdicionarTarea.html.twig*.
- Clase controladora: permiten encapsular las funcionalidades necesarias para interactuar con las clases interfaces, las clases entidades y de acceso a los datos. Agrupa la lógica del negocio ya que responde a las peticiones HTTP provenientes del cliente e invoca peticiones a la base de datos. En este caso es la clase *GestionarTareasController.php*.
- Clase de acceso a datos: permite la comunicación entre la clase controladora del negocio de la aplicación y la base de datos. En el diagrama están representadas por las clases *EntityManager.php* y *Connection.php*.
- Clases persistentes: representan el almacenamiento de datos que persistirá más allá de la ejecución del sistema. Están representadas por las clases de entidades: *Tarea.php*, *Estudiante.php*, *Profesor.php* e *Indicador.php*.

Patrones del diseño de software

Un patrón es “un problema que tiene una solución asociada que se puede adaptar a nuevos contextos y que proporciona consejos acerca de cómo puede aplicarse a nuevas situaciones”. (Pérez, 2013)

Los patrones de diseño brindan soluciones a una serie de problemas comunes que se presentan en el desarrollo de software. Facilitan la reutilización y la capacidad de expansión del software, reducen la complejidad del código y del acoplamiento y facilitan el mantenimiento. (Montenegro, Rodríguez y Salazar, 2012) En el trabajo de diploma se hace uso de los patrones de diseño de software GRASP y GoF.

Patrones GRASP

Los patrones GRASP (General Responsibility Assignment Software Patterns, Patrones Generales de Software para la Asignación de Responsabilidades) fomentan una serie de buenas prácticas para el diseño de software. El objetivo de estos patrones es describir los principios fundamentales del diseño de objetos y la asignación de responsabilidades. Los principales patrones GRASP son: Experto, Creador, Alta cohesión, Bajo acoplamiento y Controlador. Cada uno describe un problema, una solución y los beneficios de implementarlos [70].

- Experto: plantea que las responsabilidades se deben asignar al objeto experto en información.
- Creador: consiste en determinar el objeto que debe ser responsable de crear una nueva instancia de la clase.

- Alta cohesión: asegura la necesidad de mantener el sistema con un bajo nivel de complejidad, es decir, una clase tiene una responsabilidad moderada en un área funcional y colabora con otras clases para llevar a cabo las tareas.
- Bajo acoplamiento: permite mantener bajas dependencias, bajo impacto al cambio e incrementar la reutilización.
- Controlador: asigna la responsabilidad de gestionar mensajes en el sistema a una clase que represente al sistema global, a un subsistema, manejador de sesión o de eventos de un escenario de caso de uso.

La Figura 10 presenta el uso de los patrones GRASP en SIAGEPTI.

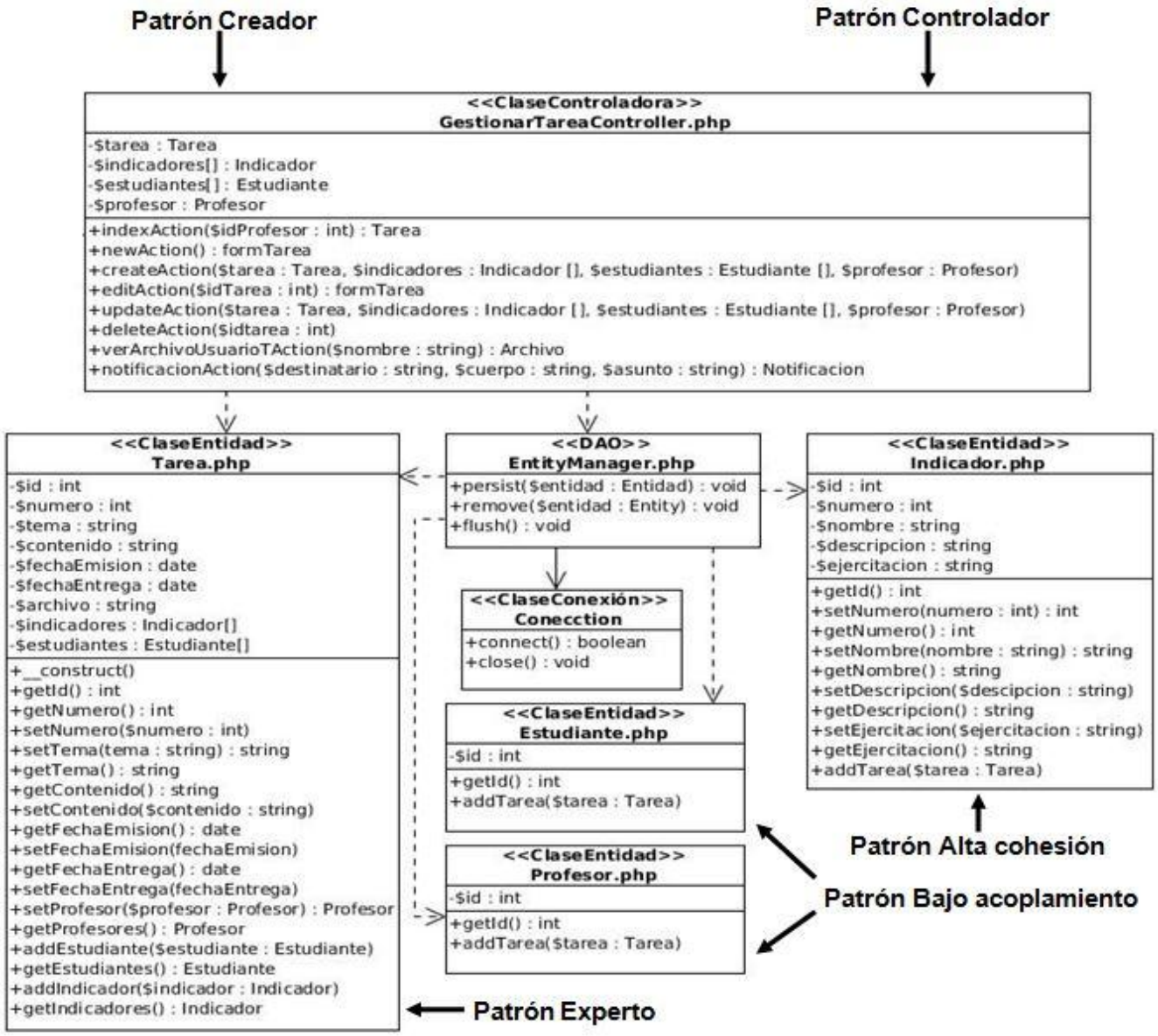

Figura 10. Ejemplo del uso de los patrones GRASP en el SIAGEPTI

- En la clase *Tarea.php* se evidencia el uso del patrón Experto ya que es la que contiene las funcionalidades necesarias para acceder a la información de una tarea.
- La clase *GestionarTareaController.php* hace uso del patrón Creador ya que es la única la única clase que posee una funcionalidad para crear una tarea, en este caso el método *createAction()*.
- El patrón Controlador está presente en la clase *GestionarTareaController.php* ya que es la responsable de gestionar la información relacionada a una tarea, provenientes del cliente.
- En la clase *EntityManager.php* se utiliza el patrón Alta cohesión ya que está formada por varias funcionalidades que están estrechamente relacionadas, siendo la misma la responsable de definir las acciones de persistencia de los datos en la base de datos y colaborar con otras clases para cumplir con ese propósito.
- El patrón bajo acoplamiento está presente en las clases *Estudiante.php*, *Profesor.php* e *Indicador.php* ya que para cumplir con sus objetivos se relaciones solamente con la clase *EntityManager.php*, proporcionando una baja dependencia entre las clases.

Patrones GoF

Los patrones GoF (The Gang of Four, o La banda de los cuatro), conocidos así por las cuatro personas que lo propusieron, son patrones de diseño de software que se utilizan para proveer soluciones comprobadas a problemas comunes que se presentan en el desarrollo de software. Estos patrones se clasifican en tres categorías: creacionales, estructurales y de comportamiento. Los patrones creacionales se ocupan del proceso de creación de clases y objetos. Los estructurales tratan la composición de clases y objetos, se ocupan de cómo estos se agrupan, para formar estructuras más grandes, permitiendo que los cambios en los requisitos de la aplicación no ocasionen cambios en las relaciones entre los objetos. Los de comportamiento caracterizan las formas en que las clases o los objetos interactúan y distribuyen la responsabilidad [71].

En el diseño de la solución se utilizó el patrón Decorador, contenido dentro de la categoría de los patrones estructurales de GoF. El propósito de este patrón es añadir responsabilidades adicionales a un objeto dinámicamente, proporcionando una alternativa flexible a la especialización mediante herencia, cuando se trata de añadir funcionalidades. Aporta una mayor flexibilidad que la herencia estática, permitiendo añadir una funcionalidad dos o más veces [72].

El patrón está implementado en la clase abstracta *View* de Symfony utilizada en la creación de plantillas para las páginas *html.twig*. Permite agregar funcionalidades de forma dinámica ya que a través de la plantilla base o global que almacena el código HTML común a todas las páginas de la aplicación, las demás vistas pueden heredar el código que esta posee y redefinir su contenido basada en esa estructura heredada. La Figura 11 muestra la utilización del patrón Decorador implementado

en las vistas *Base.html.twig,* la plantilla base del sistema y *GestionarTarea.html.twig*, una página cliente que hereda el código definido en la plantilla base.

Fragmento de código de la clase Base.html.twig

```
<!DOCTYPE html>
<html lang="es">
    <head>
        <title>SIAGEPTI | {% block title %} {% endblock %}</title>
        {% block stylesheets %}
            {# Contiene todos los css #}
            <link href="{{ asset('bundles/assets/css/bootstrap.css') }}" rel="stylesheet">
            <link href="{{ asset('bundles/assets/css/docs.css') }}" rel="stylesheet">
            <link href="{{ asset('bundles/assets/css/modern-business.css') }}" rel="stylesheet">
        {% endblock %}

        {% block javascripts %}
            {# Contiene todos los javascript #}
            <script src="{{ asset('bundles/assets/js/jquery.js') }}"></script>
            <script src="{{ asset('bundles/assets/js/bootstrap.js') }}"></script>
            <script src="{{ asset('bundles/assets/js/modern-business.js') }}"></script>
        {% endblock %}
    </head>
```

Fragmento de código de la clase GestionarTarea.html.twig

```
{% extends "::base.html.twig" %}
{% block title %}  <title>Tareas</title> {% endblock %}
{# Hereda todo el contenido especificado en la plantilla Base.html.twig #}
{% block stylesheets %}
    {{ parent() }}
{% endblock %}

{% block javascripts %}
    {{ parent() }}
{% endblock %}
```

Figura 11. Ejemplo de uso del patrón Decorador en el SIAGEPTI

Otros patrones GoF implementados en el marco de trabajo Symfony 2 son:

- Fábrica: pertenece a la categoría de patrones creacionales de GoF y permite crear diferentes familias de objetos. En Symfony 2 se emplea para la creación de objetos que pueden ser utilizados como servicios para otras clases, indicándole al contenedor de servicios que llame a un método del objeto creado y no directamente una instancia del objeto [43].
- Observador: pertenece a la categoría de patrones de comportamiento de GoF. Define una dependencia “uno a muchos” entre objetos, para que, cuando uno de ellos cambie su estado, todos los que dependan de él sean avisados y puedan actualizarse convenientemente [72].

Symfony 2 implementa este patrón en el despachador de eventos para la gestión de los eventos ocurridos en Symfony [45].

La utilización de estos patrones no es visible al desarrollador ya que son implementados y utilizados internamente en el marco de trabajo Symfony 2.

2.5 Implementación

La implementación del sistema consiste “en la traducción del diseño en una plataforma tecnológica (lenguaje de programación, servidores web, protocolos de comunicación). El resultado es un producto de software con las características descritas en la actividad anterior, en esta actividad se obtiene un producto de software que puede ser utilizado por el cliente” [66].

2.5.1 Modelo de implementación

El modelo de implementación describe cómo los elementos del modelo del diseño se implementan en términos de componentes. Este modelo refiere también cómo se organizan los componentes de acuerdo con los mecanismos de estructuración y modulación disponibles en el entorno de implementación y en el lenguaje de programación empleado y cómo dependen los componentes unos de otros [73].

Diagrama de componentes

El diagrama de componentes muestra las organizaciones y dependencias lógicas entre componentes software, sean éstos de código fuente, binarios o ejecutables. Desde el punto de vista del diagrama de componentes se tienen en consideración los requisitos relacionados con la facilidad de desarrollo, la gestión del software, la reutilización, y las restricciones impuestas por los lenguajes de programación y las herramientas utilizadas en el desarrollo [74].

La siguiente figura presenta el diagrama de componentes de la aplicación propuesta, en él se observan los componentes unidad física de implementación con interfaces bien definidas y pensada para ser utilizada como parte reemplazable de un sistema e incorpora la implementación de ciertas clases del diseño del software y las relaciones existentes entre ellos [28]. En la Figura 12 se presenta el diagrama de componentes del SIAGEPTI.

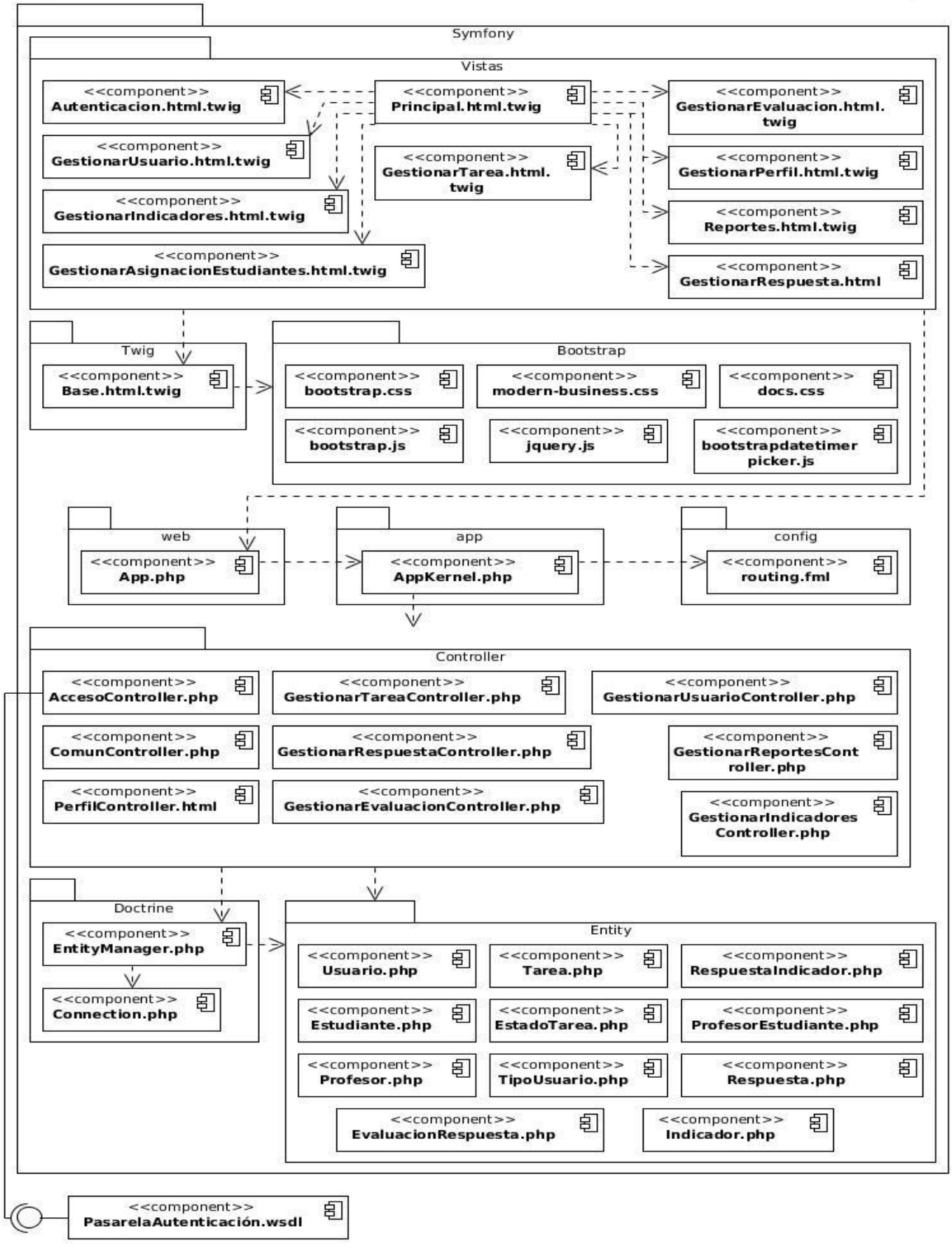

Figura 12. Diagrama de componentes del SIAGEPTI

Los componentes contenidos en el diagrama anterior son:

- Los componentes contenidos en Vistas: representan las páginas clientes que permiten la comunicación de los usuarios con el sistema.
- *Base.html.twig*: representa la plantilla base para la organización de la estructura de las páginas *html.twig.*
- *bootstrap.css, modern-business.css y docs.css*: representan los archivos de estilos de formato visual (CSS) de las páginas *html.twig.*
- *bootstrap.js, jquery.js y bootstrapdatetimerpicker.js:* representan los archivos de validación de Javascript (JS).
- *App.php:* es el componente que representa al controlador frontal y es el único punto de entrada a la aplicación.
- *AppKernel.php:* representa a la clase responsable de cargar el archivo de configuración del controlador que se utilice. Interactúa con el componente *routing.yml* para gestionar las rutas de las clases que contienen las funcionalidades que responden a las peticiones del cliente y con las clases controladoras.
- *routing.yml:* representa el archivo que contiene las rutas de las funcionalidades del sistema.
- Los componentes contenidos en *Controller*: representan a las clases controladoras que gestionan el flujo de información en el sistema.
- *EntityManager.php*: es la clase que gestiona la manipulación de la información de Doctrine2 como: buscar, crear, modificar y borrar registros en las tablas de la base de datos.
- *Connection.php:* es la clase responsable de establecer la comunicación con la base de datos.
- Los componentes contenidos en *Entity*: representan a las clases entidades que contienen la información persistente del sistema.
- *PasarelaAutenticacion.wsdl:* permite la autenticación y obtención de los datos de los usuarios del sistema, mediante un usuario y contraseña del dominio UCI.

Diagrama de despliegue

Este diagrama se utiliza al momento de modelar aspectos estáticos de nodos físicos y sus relaciones. Con este diagrama, es posible modelar muchos de los aspectos hardware de un sistema con el nivel de detalle adecuado para que se pueda especificar la plataforma sobre la que se ejecutará, de tal forma que la frontera software - hardware del sistema sea mejor manejada [75].

El diagrama de despliegue del SIAGEPTI presentado en la Figura 13, muestra los nodos (recursos de hardware necesarios para la ejecución de la aplicación, en este caso computadoras), así como los componentes asociados a cada uno de ellos y las relaciones de comunicación existentes entre los nodos a través de protocolos.

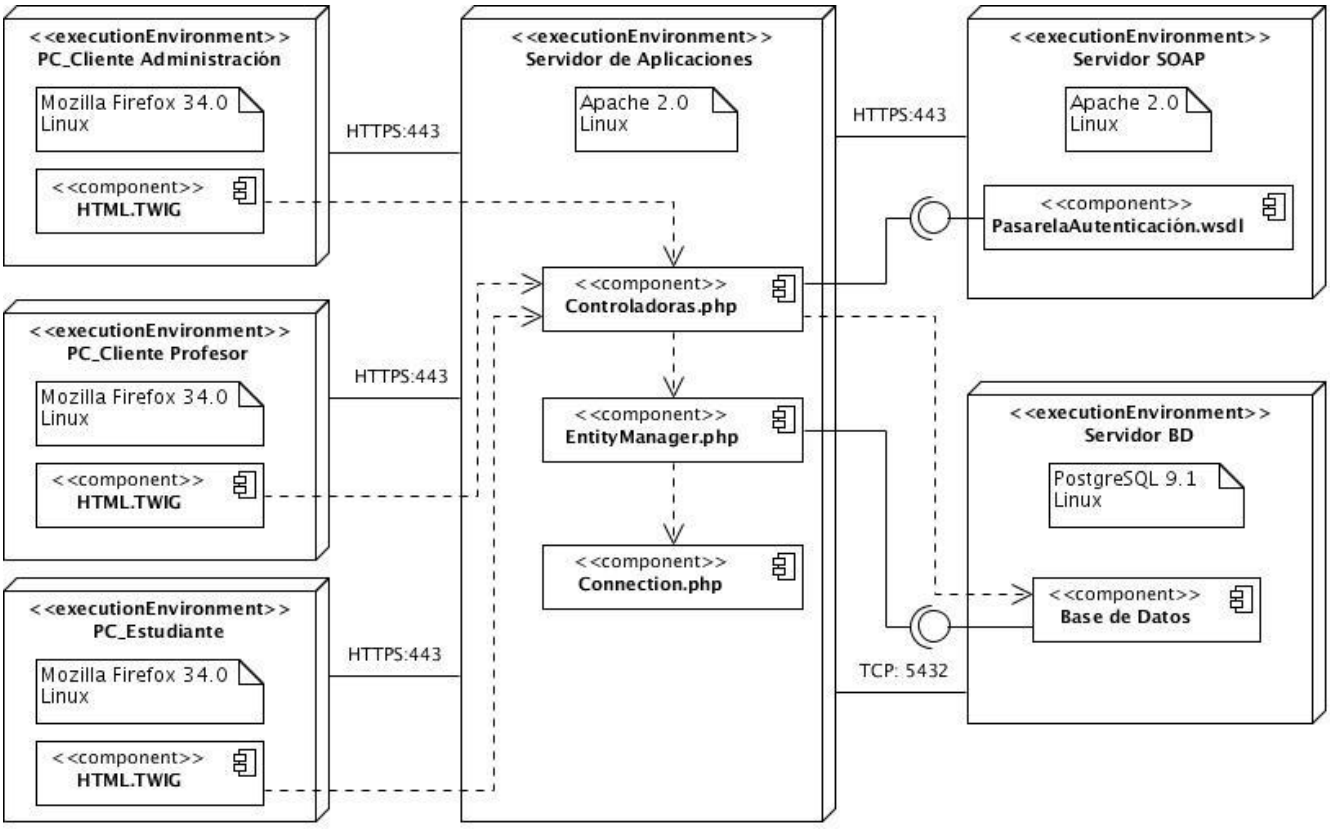

Figura 13. Diagrama de despliegue del SIAGEPTI

Los nodos que integran el diagrama de despliegue son:

- PC_Cliente Administración, PC_Cliente Profesor, PC_Cliente Estudiante: contendrán el sistema operativo Linux y el navegador web Mozilla Firefox 34.0 mediante el cual los administradores profesores y estudiantes tendrán acceso al sistema y harán uso del mismo.
- Servidor de aplicaciones: proporciona los servicios de la aplicación a las computadoras cliente y gestiona las funciones de la lógica de negocio y de acceso a los datos necesarios, el servidor de aplicaciones será el Apache 2.0 y el sistema operativo Linux.
- Servidor de Base de Datos: contiene la base de datos con la información del sistema y permite acceder a los datos almacenados en la base de datos y la organización de los mismos, el servidor de base datos será PostgreSQL 9.1 y el sistema operativo Linux.
- Servidor SOAP (Simple Object Access Protocol, Protocolo de Acceso a Objetos Simples): provee servicios para la autenticación de los administradores, profesores y estudiantes que interactuarán con el sistema y la obtención de los datos personales de los usuarios de SIAGEPTI. El sistema operativo que utiliza es Linux.

Los protocolos que permiten la relación entre los nodos de la aplicación son:

- HTTPS: Hypertext Transfer Protocol Secure (Protocolo seguro de transferencia de hipertexto) es el protocolo usado para transferir información entre los clientes y los servidores de aplicación y de autenticación de forma segura.
- TCP: Transmission Control Protocol (Protocolo de Control de Transmisión), es el protocolo utilizado para gestionar el flujo de datos entre el servidor de base de datos y el servidor de aplicaciones.

2.5.2 Estándares de codificación

Un estándar de codificación "es un conjunto de reglas, normas o patrones destinados a establecer uniformidad en el proceso de generación de un código. Son pautas de programación enfocadas a la estructura y apariencia física del código para facilitar su lectura, comprensión y mantenimiento" [76].

En la implementación del sistema se utilizaron los siguientes estándares de codificación:

- Notación de Pascal (Pascal Case): es un estilo de escritura que se utiliza para escribir el nombre de los identificadores, variables o componente de un proyecto colocando en mayúscula la primera letra de cada palabra que forme el nombre del elemento [77]. Esta notación fue empleada para declarar el nombre de las clases.
- Notación de camello (Camel Case): es un estilo de notación que se emplea para escribir la primera letra de la identificación con minúsculas y la inicial de cada una de las palabras concatenadas se escribe con mayúscula. Existen dos tipos de Camel Case, el estilo Upper Camel Case, que define que la primera letra de cada una de las palabras que identifican a un elemento de un proyecto es mayúscula y el Lower Camel Case cuando la primera letra es minúscula. [77] El estilo de notación utilizado en la declaración de variables y métodos es el segundo.
- El nombre de las clases controladoras terminan con la palabra Controller.
- El nombre de las funcionalidades públicas terminan con la palabra Action.
- Las variables que poseen nombres de palabras compuestas se escriben juntas.
- Se hizo uso de los comentarios para la explicación del código siempre que fue necesario.

En la Figura 14 se presenta un fragmento de código de la clase *TareaController*. En ella se ejemplifica el uso de los estándares de codificación utilizados en la implementación de la solución.

```
/**
 * Tarea controller.
 *
 */
class TareaController extends Controller {

    /**
     * @Route("tareas", name="tareas")
     */
    public function indexAction(Request $request) {
        $entities=  $this->tareasProfesor();
        return $this->render('AppBundle:ProfesoresTarea:GestionarTarea.html.twig', array(
                    'entities' => $entities,
        ));
    }

    private function tareasProfesor() {
        $em = $this->getDoctrine()->getManager();
        $usuarioProfesorAutenticado = $this->get('security.context')->getToken()->getUser()->getUserName();
        $profesorAutenticado = $em->getRepository('AppBundle:Usuario')->findOneByUsuario($usuarioProfesorAutenticado);
        $idProfesorAutenticado = $profesorAutenticado->getId();
```

Notación de Pascal

Notación de camello estilo Lower Camel Case

Figura 14. Ejemplificación del uso de los estándares de codificación

Los estándares de codificación definidos en la investigación permiten tener un estilo único de codificación facilitando el estudio y entendimiento del código de la aplicación y garantizando que este sea más fácil de mantener.

2.6 Interfaz del sistema SIAGEPTI

La interfaz de usuario es el medio de comunicación entre el usuario y el sistema y debe cumplir con las tres reglas siguientes: dar control al usuario, reducir la carga de memoria del usuario y debe ser consistente. El cumplimiento de estas reglas permite desarrollar interfaces flexibles, en las que el usuario no realice acciones innecesarias y no deseadas, así como la interrupción de una secuencia de acciones en el instante requerido por el usuario. Además deben desglosar la información de forma progresiva a través de una estructura organizada, orientar al usuario en las tareas a desarrollar y que esas tareas se realicen en el contexto adecuado [61]. La Figura 15 muestra la interfaz de usuario para la gestión de indicadores en la sesión de los profesores con el rol de administración.

Figura 15. Interfaz de la sesión del profesor con el rol de administración para la gestión de indicadores

2.7 Conclusiones del capítulo

La informatización de la gestión de indicadores de medición del talento para el seguimiento del desarrollo de las habilidades de los estudiantes potencialmente talentosos de la UCI se obtuvo a través de la implementación de SIAGEPTI. Se tuvieron en cuenta para el desarrollo de la solución 51 requisitos funcionales y 13 no funcionalidades cumpliendo con las necesidades del cliente. La utilización de los patrones de diseño y el patrón MVC durante el diseño y la implementación del sistema, proporcionó una mayor calidad del producto.

CAPÍTULO III: Validación del sistema

3.1 Introducción

En el presente capítulo se describe el proceso de verificación y validación desarrollado para evaluar la propuesta de solución. Se verifica la calidad del resultado de la implementación, mediante los productos de trabajos generados durante las disciplinas de pruebas, exponiendo los resultados obtenidos. Además se aplica la técnica de Iadov para medir la satisfacción de los futuros usuarios de SIAGEPTI.

3.2 Validación del diseño del sistema

La validación del diseño consiste en verificar que el diseño de un sistema cumple determinados atributos de calidad. En esta validación se utilizan las métricas técnicas del software [61]. Una métrica es una medida efectuada sobre algún aspecto del sistema en desarrollo, que permite obtener conclusiones sobre el aspecto medido con el fin de adoptar las decisiones necesarias mediante una previa comparación con unos valores de referencia o medida. Es un medio para controlar el desarrollo de un sistema de software y evaluar su calidad objetivamente [78].

En la validación de la propuesta se utilizaron las métricas Tamaño operacional de clase (TOC), para conocer el estado de la responsabilidad, complejidad de implementación y reutilización que poseen las clases definidas y Relaciones entre clases (RC), para medir el acoplamiento, la complejidad de mantenimiento y la reutilización que presentan estas clases así como la cantidad de pruebas necesarias para probar una clase. A continuación se explican los pasos seguidos para la aplicación de estas métricas y los resultados obtenidos.

3.2.1 Métrica Tamaño operacional de clase (TOC)

Está dado por el número de métodos asignados a una clase y evalúa los siguientes atributos de calidad:

- Responsabilidad: mide la responsabilidad que tiene una clase determinada. Un aumento del TOC implica un aumento de la responsabilidad asignada a la clase.
- Complejidad de implementación: evalúa la complejidad de implementación que tiene una clase del diseño. Un aumento del TOC implica un aumento de la complejidad de implementación de la clase.
- Reutilización: indica el grado de reutilización que tiene una clase. Un aumento del TOC implica una disminución del grado de reutilización de la clase.

Pasos seguidos para la aplicación de la métrica

1. Se determinó la cantidad de procedimientos que tiene las clases del sistema.
2. Se calcula el promedio de procedimientos:

Promedio=cantidad total de procedimiemtos/cantidad de clases

Promedio= 242/20 = 12,1

3. Se le asigna una categoría (Baja, Media, Alta) a cada atributo de calidad (Responsabilidad, Complejidad de implementación y Reutilización) por cada una de las clases siguiendo los siguientes criterios:

Responsabilidad y Complejidad de implementación:

- Baja: si la cantidad de procedimientos es <= Promedio.
- Media: si la cantidad de procedimientos está entre Promedio y 2* Promedio.
- Alta: si la cantidad de procedimientos > 2* Promedio.

Reutilización:

- Baja: si la cantidad de procedimientos es > 2* Promedio.
- Media: si la cantidad de procedimientos está entre Promedio y 2* Promedio.
- Alta: si la cantidad de procedimientos <=Promedio.

Resultados obtenidos

Tabla 4. Resultados de la aplicación de la métrica TOC

Resultados de la aplicación de la métrica TOC									
Aspectos	**Indicadores**								
	Responsabilidad			**Complejidad de implementación**			**Reutilización**		
	Baja	**Media**	**Alta**	**Baja**	**Media**	**Alta**	**Baja**	**Media**	**Alta**
Cantidad de clases	12	6	2	12	6	2	2	6	12
Promedio	60	30	10	60	30	10	10	30	60

Figura 16. Resultados de la aplicación de la métrica TOC

La aplicación de la métrica TOC arrojó como resultados que las clases del sistema poseen un 60% de baja responsabilidad, un 60 % de baja complejidad de implementación y un 60% de alta reutilización por lo que el diseño de las clases en cuanto a cantidad de funcionalidades o procedimientos es bueno.

3.2.2 Métrica Relaciones entre clases (RC)

Está dado por el número de relaciones de uso de una clase con otra y evalúa los siguientes atributos de calidad:

- Acoplamiento: representa las conexiones físicas (colaboraciones) entre las clases del diseño. Un aumento del RC implica un aumento del Acoplamiento de la clase.
- Cantidad de pruebas: evalúa la cantidad de pruebas necesarias para probar una clase. Un aumento del RC implica un aumento de la Cantidad de pruebas de unidad necesarias para probar la clase.
- Complejidad de mantenimiento: mide la complejidad de mantenimiento que tiene una clase determinada. Un aumento del RC implica un aumento de la complejidad del mantenimiento de la clase.
- Reutilización: proporciona una medida de la reutilización de una clase. Un aumento del RC implica una disminución en el grado de reutilización de la clase.

Pasos seguidos para la aplicación de la métrica

1. Se determinó la cantidad de relaciones de uso que tiene las clases del sistema.
2. Se calcula el promedio de relaciones de uso:

   Promedio = cantidad total de relaciones de uso/cantidad de clases

   Promedio = 48/20 = 2,4

3. Se le asigna una categoría (Ninguno, Bajo, Medio, Alto) al atributo de calidad Acoplamiento y una categoría (Bajo, Medio, Alto) a los siguientes atributos de calidad (Complejidad de mantenimiento, Reutilización y Cantidad de pruebas) por cada una de las clases siguiendo los siguientes criterios:

Acoplamiento:

- Ninguno: si las relaciones de uso es 0.
- Bajo: si las relaciones de uso es 1
- Medio: si las relaciones de uso es 2
- Alto: si las relaciones de uso es >2

Complejidad de mantenimiento y Cantidad de pruebas:

- Baja: si las relaciones de uso es <= Promedio.
- Media: si las relaciones de uso está entre Promedio y 2* Promedio.

- Alta: si las relaciones de uso > 2* Promedio.

Reutilización:

- Baja: si las relaciones de uso es > 2* Promedio.
- Media: si las relaciones de uso está entre Promedio y 2* Promedio.
- Alta: si las relaciones de uso <=Promedio.

Resultados obtenidos

Tabla 5. Resultados de la aplicación de la métrica RC.

Resultados de la aplicación de la métrica RC													
Aspectos	**Indicadores**												
	Acoplamiento				**Complejidad de mantenimiento**			**Reutilización**			**Cantidad de pruebas**		
	Ninguno	**Baja**	**Media**	**Alta**	**Baja**	**Media**	**Alta**	**Baja**	**Media**	**Alta**	**Baja**	**Media**	**Alta**
Cantidad de clases	0	8	7	5	11	9	0	0	8	12	12	8	0
Prome-dio	0	40	35	25	55	45	0	0	40	60	60	40	0

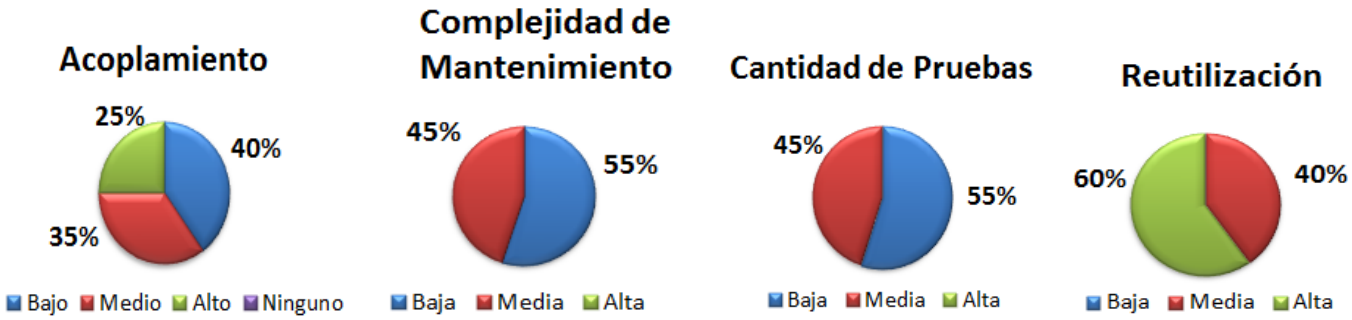

Figura 17. Resultados de la aplicación de la métrica RC

El resultado de aplicar la métrica RC evidencia un 40% de bajo acoplamiento, un 55% de la baja complejidad de mantenimiento, un 55% de la baja cantidad de pruebas y un 60% de alta reutilización que poseen las clases diseñadas para el sistema.

La aplicación de las métricas TOC y RC permitió tener una medida cuantitativa del grado en el que la aplicación cumple con los indicadores de calidad anteriormente descritos. Proporciona una visión del diseño realizado en el proceso de desarrollo de la propuesta de solución y demuestra que las clases del sistema poseen una baja carga de responsabilidades, un bajo acoplamiento y una alta reutilización permitiendo una fácil implementación, una baja complejidad de mantenimiento y baja la

cantidad pruebas de unidad necesarias para probar las clases, pudiéndose afirmar que los resultados obtenidos por la aplicación y evaluación de las métricas son positivos.

3.3 Pruebas de software

Los niveles de pruebas de software desarrolladas como parte del proceso de verificación y validación de la aplicación propuesta son: pruebas internas, pruebas de liberación y pruebas de aceptación.

3.3.1 Pruebas internas

En esta disciplina se verifica el resultado de la implementación probando cada construcción, incluyendo tanto las construcciones internas como intermedias, así como las versiones finales a ser liberadas, desarrollando artefactos de prueba como diseños de casos de prueba [26]. A continuación se presenta la prueba aplicada en esta disciplina.

Prueba de unidad o unitarias

Se concentran en probar cada componente individualmente para asegurar que funcione de manera apropiada como unidad. Emplean técnicas de prueba que recorren caminos específicos en la estructura de control de los componentes (pruebas estructurales) [79].

El método de prueba utilizado para la realización de esta prueba es la Prueba de caja blanca. La técnica de prueba contenida en este método que se empleó fue la Técnica del camino básico.

3.3.2 Pruebas de liberación

Las pruebas de liberación son diseñadas y ejecutadas por una entidad certificadora de la calidad externa, a todos los entregables de los proyectos antes de ser entregados al cliente para su aceptación [26]. La prueba realizada para la liberación del sistema fue la prueba de validación.

Pruebas de validación

La validación se refiere a verificar que el software funciona de acuerdo con los requerimientos del mismo (las necesidades del cliente) [80].

El método de prueba empleado en la validación fue la Prueba de caja negra y la técnica desarrollada fue la Partición equivalente.

3.3.3 Pruebas de aceptación

Es la prueba final antes del despliegue del sistema. Su objetivo es verificar que el software está listo y que puede ser usado por usuarios finales para ejecutar aquellas funciones y tareas para las cuales el software fue construido [26].

3.4 Métodos y técnicas de pruebas

Métodos de pruebas utilizados:

- Pruebas de caja blanca o estructurales: verifican la correcta implementación de las unidades internas, las estructuras y sus relaciones y hacen énfasis en la reducción de errores internos [79].
- Pruebas de caja negra o funcionales: verifican el correcto manejo de funciones externas provistas o soportadas por el software y que el comportamiento observado se apegue a las especificaciones del producto y a las expectativas del usuario [79].

Las técnicas de prueba utilizadas en la aplicación de los métodos de prueba fueron:

- Técnica del camino básico: es una técnica de prueba de caja blanca que permite obtener una medida de la complejidad lógica de un diseño procedimental y usar esa medida como guía para la definición de un conjunto básico de caminos de ejecución independiente en un componente o programa [61]. Un camino o ruta es una vía por la cual procede la ejecución a través de una función desde su inicio hasta el fin [80].
- Partición equivalente: es una técnica de prueba de caja negra que divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba [61]. La partición equivalente busca obtener casos de prueba ideales que descubran de forma inmediata clases de errores, reduciendo el número total de casos de prueba a desarrollar. Una clase de equivalencia representa un conjunto de estados válidos o no válidos para condiciones de entrada. Una condición de entrada es un valor numérico específico, un rango de valores, un conjunto de valores relacionados o una condición lógica [80].

3.5 Aplicación y resultados de las pruebas realizadas al sistema

3.5.1 Pruebas internas

Las pruebas internas se desarrollaron a través de pruebas unitarias utilizando la técnica del camino básico de las pruebas de caja blanca. En esta técnica se utilizó la notación de grafo de flujo para representar en un grafo de flujo o del programa el código de la funcionalidad o procedimiento a probar. Se utiliza además la métrica del software complejidad ciclomática que proporciona una medición cuantitativa de la complejidad lógica de un programa y cuando se usa en términos de prueba de camino básico su valor calculado define el número de caminos independientes (cualquier camino del programa que introduce, por lo menos un nuevo conjunto de sentencias de proceso o una nueva condición) del conjunto básico de un programa y proporciona el límite superior para el número de pruebas que se deben realizar para asegurar que se ejecuta cada sentencia al menos una vez [61]. A continuación se relacionan los pasos de la aplicación de la prueba del camino básico.

Pasos para la aplicación de la prueba del camino básico:

1. Dibujar el grafo de flujo de la funcionalidad o procedimiento a analizar.

```
/**
 * @Route("indicadores/{id}/update", name="indicadores_update")
 */
public function updateAction(Request $request, $id)
{
    $em = $this->getDoctrine()->getManager();

    $entity = $em->getRepository('AppBundle:Indicador')->find($id);

    if (!$entity) {
        throw $this->createNotFoundException('Unable to find Indicador entity.');
    }

    $deleteForm = $this->createDeleteForm($id);
    $editForm = $this->createEditForm($entity);
    $editForm->handleRequest($request);

    if ($editForm->isValid()) {
        $em->flush();

        return $this->redirect($this->generateUrl('gestionarindicadores', array('id' => $id)));
    }

    return $this->render('AppBundle:AdministracionIndicadores:ModificarIndicadores.html.twig', array(
        'entity'      => $entity,
        'edit_form'   => $editForm->createView(),
        'delete_form' => $deleteForm->createView(),
    ));
}
```

Figura 18. Código de la implementación del requisito funcional Modificar indicador

La Figura 19 presenta el grafo de flujo obtenido del código representado en la Figura 18.

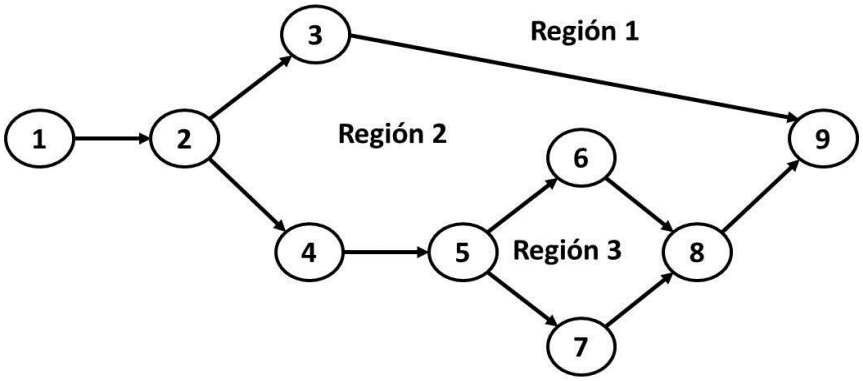

Figura 19. Grafo de flujo del código de implementación del requisito funcional Modificar indicador

2. Determinar la complejidad ciclomática del grafo de flujo resultante.

La complejidad ciclomática del grafo (V(G)) se puede calcular de las tres formas siguientes:

- V(G) = A - N + 2

  V(G) = 10 – 9 + 2 = 3

  donde: A es el número de aristas del grafo de flujo
  N es el número de nodos del grafo

- V(G) = P + 1

  V(G) = 2 + 1 = 3

  donde: P es el número de nodos predicados (nodos con más de una arista de salida) contenidos en el grafo.

- Regiones = V(G)

  3 = 3

  donde: Regiones son las área delimitadas por nodos y aristas en el grafo.

3. Determinar el conjunto básico de caminos linealmente independientes.

- Camino básico 1: 1-2-3-9
- Camino básico 2: 1-2-4-5-6-8-9
- Camino básico 3: 1-2-4-5-7-8-9

4. Definir los casos de prueba para comprobar la ejecución de cada camino del conjunto básico.

En el diseño de los casos de prueba se debe especificar los siguientes elementos:

- Descripción: contiene una descripción sobre las restricciones de los datos de entrada que debe tener el caso de prueba.
- Condición de ejecución: se especifican los parámetros que debe poseer el caso de prueba para que se cumpla una condición deseada como respuesta del funcionamiento del procedimiento.
- Entrada: se muestran los parámetros de entrada al procedimiento.
- Resultados esperados: se explica el resultado esperado de la ejecución del procedimiento.

La Tabla 6 muestra el diseño de caso de prueba para el camino 2 del conjunto básico de caminos linealmente independientes, correspondiente a la funcionalidad Modificar indicador. En este camino se prueba la modificación de los datos de un indicador en la base de datos de forma satisfactoria. Los diseños de casos de pruebas de los caminos restantes, 1 y 3 se presentan en los anexos 6 y 7 respectivamente.

Tabla 6. Diseño de caso de prueba para el camino 2

Diseño de caso de prueba para el camino 2	
Descripción	Los datos de entrada son válidos y deben cumplir con el formato indicado.
Condición	Campos válidos:

de ejecución	El campo Número permite la entrada del número del indicador y es un entero, único y no nulo. El campo Nombre permite la entrada del nombre del indicador y es una cadena de caracteres no nula. El campo Descripción se utiliza para introducir la descripción del indicador y es una cadena de caracteres no nula. El campo Ejercitación permite la entrada del nombre del indicador y es una cadena de caracteres no nula.
Entrada	Número: 1 Nombre: Capacidad para identificar y definir problemas en algunas de las áreas de las ciencias informáticas. Descripción: Se mide la capacidad que tiene un estudiante potencialmente talentoso para identificar y definir problemas en algunas de las áreas de las ciencias informáticas. Ejercitación: Análisis de casos de estudios.
Resultados esperados	Se modifican los datos del indicador satisfactoriamente.

A partir de la aplicación del caso de prueba expuesto anteriormente se comprobó que el flujo de trabajo de las funcionalidades es correcto, ya que se comprobó que cada sentencia es ejecutada al menos una vez, cumpliéndose así las condiciones de la prueba y el resultado esperado es satisfactorio.

3.5.2 Pruebas de liberación

En la pruebas de liberación se realizaron pruebas de validación, a través del método de caja negra y la técnica de partición de equivalencia. El diseño de casos de prueba para la partición equivalente se basa en una evaluación de las clases de equivalencia para una condición de entrada. Los casos de prueba diseñados contienen clases de equivalencias válidas y no válidas, para cada campo de entrada del sistema. A continuación se presenta el diseño del caso de prueba realizado para la validación del requisito funcional Adicionar tarea.

Descripción general del caso de prueba para el requisito funcional Adicionar tarea

El requisito funcional Adicionar tarea comienza cuando el profesor selecciona Adicionar en la interfaz Gestionar tarea. El sistema muestra la interfaz Adicionar tarea, en la que se deben insertar los datos requeridos en el formulario de la interfaz. El seleccionar el sistema valida los datos introducidos y si

son correctos son introducidos a la base de datos del sistema y concluye el requisito, si existen datos incorrecto el sistema informa el error existente y espera por una acción del usuario.

Condiciones de ejecución

1. Debe existir al menos un usuario registrado en el sistema con el rol de profesor para poder adicionar una tarea.
2. Debe existir al menos un usuario registrado en el sistema con el rol de estudiante para asignarle la tarea.
3. Debe existir al menos un indicador insertado en el sistema para ser adicionado a la tarea.
4. Se debe seleccionar la opción Adicionar de la interfaz Gestionar tarea.

Tabla 7. Escenarios de prueba para el requisito funcional Adicionar tarea

Escenarios de pruebas	Respuesta del sistema	Flujo central del escenario
EC 1.1: Adicionar tarea introduciendo datos válidos.	El sistema adiciona la tarea y la muestra en el listado de tareas.	Se presiona la opción Adicionar. Se muestra el formulario Adicionar tarea. Se introducen los datos de la tarea correctamente. Se presiona el botón Aceptar. El sistema valida los datos introducidos. El sistema adiciona los datos a la base de datos y se muestran en el listado de tareas.
EC 1.2: Adicionar tarea introduciendo datos inválidos.	El sistema no permite introducir datos incorrectos. Se muestra un mensaje de error.	Se presiona la opción Adicionar. Se muestra el formulario Adicionar tarea. Se introducen los datos de la tarea incorrectamente. Se presiona el botón Aceptar. El sistema valida los datos introducidos, muestra un mensaje de error y vuelve a visualizar el formulario.
EC 1.3: Adicionar tarea dejando campos vacíos.	El sistema marca en rojo el campo y muestra un mensaje de error.	Se presiona la opción Adicionar. Se muestra el formulario Adicionar tarea. Se introducen los datos de la tarea dejando campos vacíos. Se presiona el botón Aceptar. El sistema valida, muestra el error y vuelve a visualizar el formulario.

EC 1.4 Cancelar.	El sistema cierra la interfaz sin realizar ninguna operación. Regresa a la página Gestionar tarea.	Se presiona la opción Adicionar. Se muestra el formulario Adicionar tarea. Se presiona el botón Cancelar.

Definición de casos válidos y no válidos

Tabla 8. Casos válidos y no válidos del requisito funcional Adicionar tarea

Condición de entrada	Casos válidos	Casos no válidos
Tema	Cadena de texto alfanumérica.	Campo vacío.
Contenido	Cadena de texto alfanumérica.	Campo vacío.
Fecha de entrega	Seleccionar una fecha mayor a la fecha actual.	Seleccionar una fecha menor o igual a la fecha actual.
Archivo adjunto	Dejar el campo vacío o subir un archivo de tamaño menor o igual a 7 Mb.	Subir un archivo de tamaño mayor a 7 Mb.
Indicadores	Seleccionar como mínimo un indicador.	Campo vacío.
Estudiantes	Seleccionar como mínimo un estudiante	Campo vacío.

Las pruebas fueron realizadas por un especialista del grupo de calidad del Centro de Informatización de Entidades (CEIGE) de la Facultad 3. Se realizaron tres iteraciones, en la primera se encontraron tres no conformidades de funcionalidad, en la segunda iteración se encontraron dos no conformidades de funcionalidad y en la tercera iteración no se encontraron no conformidades. El sistema desarrollado fue liberado y para ello se emitió el acta de liberación que puede consultarse en el anexo 8. En la Figura 20 se presenta un gráfico con los resultados de las pruebas de liberación.

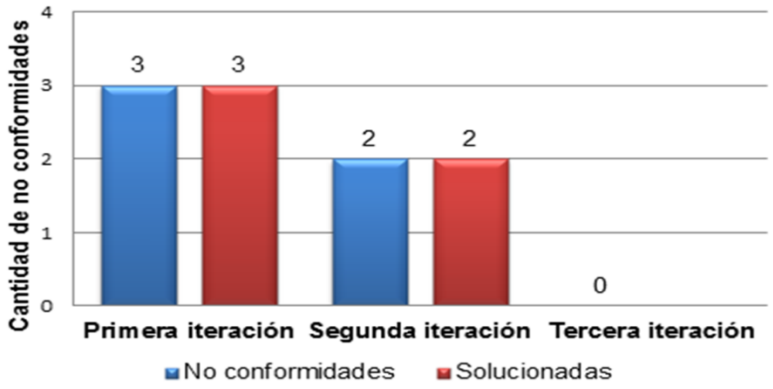

Figura 20. Resultados de las pruebas de liberación

3.5.3 Aplicación y resultados de las pruebas de aceptación

Las pruebas de aceptación fueron realizadas por el jefe del proyecto Talenmático y un profesor de dicho proyecto. Para ello se realizaron las siguientes actividades:

1. Revisión y aprobación de los requisitos de información por el cliente.
2. Realización de pruebas exploratorias, por parte del cliente, al sistema desarrollado para validar el cumplimiento de los requisitos aprobados.

Validadas todas las funcionalidades del sistema el cliente emitió el acta de aceptación que puede ser consultada en el anexo 9.

3.6 Validación del objetivo de la investigación

La validación del objetivo de la investigación se realizó con la técnica de Iadov ya que permite conocer el grado de satisfacción de potenciales usuarios. Esta técnica fue utilizada ya que "cuando se realiza una propuesta, es recomendable retroalimentarse con la opinión de los usuarios potenciales. Esta información es útil para conocer las debilidades de la propuesta y profundizar en sus fortalezas. En ese sentido, la técnica de Iadov es un instrumento que ayuda a conocer el grado de satisfacción de los potenciales usuarios. Recientemente ha sido adaptada para aplicarse en el sistema de validación en diversas investigaciones" [81].

La técnica de Iadov permitió conocer el grado de satisfacción de potenciales usuarios del SIAGEPTI para agilizar la gestión de indicadores de medición del talento para el seguimiento del desarrollo de las habilidades de los estudiantes potencialmente talentosos de la UCI. Para ello se aplicó una encuesta (ver anexo 10) a ocho profesores del proyecto Talenmático. La encuesta está estructurada en dos preguntas abiertas y cuatro cerradas las que guardan relación entre sí. La relación entre las preguntas cerradas se establece a través del cuadro lógico de Iadov, presentado en la Tabla 9.

Tabla 9. Cuadro lógico de Iadov

<table>
<tr><td rowspan="3">5. Luego de haber visto la gestión de indicadores de medición del talento refleje en qué medida le gusta la solución desarrollada.</td><td colspan="3">2. El Entorno Virtual de Aprendizaje disponible en la universidad para la actividad docente le permite definir los indicadores necesarios para el seguimiento del desarrollo de las habilidades de los estudiantes potencialmente talentosos.</td></tr>
<tr><td>No</td><td>No sé</td><td>Sí</td></tr>
<tr><td colspan="3">3. Considera usted que sería factible contar con un sistema informático que permita agilizar la gestión de indicadores de medición del talento para el seguimiento del desarrollo de las habilidades de los estudiantes potencialmente talentosos de la universidad.</td></tr>
</table>

	Sí	No sé	No	Sí	No sé	No	Sí	No sé	No
Me gusta mucho	1	2	6	2	2	6	6	6	6
Me gusta más de lo que me disgusta	2	2	3	2	3	3	6	3	6
Me da lo mismo	3	3	3	3	3	3	3	3	3
Me disgusta más de lo que me gusta	6	3	6	3	4	4	3	3	4
No me gusta nada	6	6	6	6	4	4	6	6	5
No sé decir	2	3	6	3	3	3	6	6	4

El número resultante de la interrelación de las tres preguntas indica la posición en la escala de satisfacción siguiente: clara satisfacción (A), más satisfecho que insatisfecho (B), no definida (C), más insatisfecho que satisfecho(D), clara insatisfacción (E) y contradictoria (C).

A partir de la cantidad de respuestas por categoría es posible calcular el Índice de Satisfacción Grupal (ISG) siguiendo la siguiente fórmula:

$$ISG = \frac{\mathrm{A}(+1) + \mathrm{B}(+0.5) + \mathrm{C}(0) + \mathrm{D}(-0.5) + \mathrm{E}(-1)}{N}$$

donde: N es la cantidad total de respuestas

El valor del ISG permite identificar las siguientes categorías grupales:

- Máxima insatisfacción: desde -1 hasta -0,49
- Más insatisfecho que satisfecho: desde -0,5 hasta -0,1
- No definido y contradictorio: 0
- Más satisfecho que insatisfecho: desde 0,1 hasta 0,49
- Máximo de satisfacción: desde 0,5 hasta 1

El índice general arroja valores entre + 1 y - 1. Los valores que se encuentran comprendidos entre - 1 y - 0,5 indican insatisfacción; los comprendidos entre - 0,49 y + 0,49 evidencian contradicción y los que están entre 0,5 y 1 indican que existe satisfacción.

Resultados obtenidos

Los resultados obtenidos de la aplicación del cuestionario se presentan en la Tabla 10.

Tabla 10. Resultados de las escalas de satisfacción

Total de usuarios de la muestra (N)	**8**	**Escala**
Clara satisfacción	8	A
Más satisfecho que insatisfecho	0	B
No definida	0	C
Más insatisfecho que satisfecho	0	D
Clara insatisfacción	0	E
Contradictoria	0	C

Cálculo del Índice de Satisfacción Grupal

ISG=A(+1)/N

ISG=8(+1)/8 = 1

El valor obtenido del ISG fue uno lo que indica máxima satisfacción de los usuarios con respecto al SIAGEPTI. Se puede afirmar que se cumplió el problema de la investigación. Las respuestas a las preguntas abiertas brindadas por los encuestados reafirman los beneficios que traerá la utilización del sistema propuesto.

3.7 Conclusiones del capítulo

La buena calidad del diseño se validó a través de la aplicación de las métricas TOC y RC arrojando resultados satisfactorios para cada uno de los atributos evaluados. La aplicación de la técnica Camino básico comprobó que el flujo de trabajo de las funcionalidades es correcto, ya que cada sentencia es ejecutada al menos una vez y la técnica de Partición equivalente verificó en tres iteraciones realizadas que el sistema cumple con los requisitos definidos. Se evidenció la alta satisfacción del cliente a través de las pruebas de aceptación y la aplicación de la técnica de Iadov.

CONCLUSIONES

La investigación realizada cumple con los objetivos planteados mediante el desarrollo de SIAGEPTI y se arriba a las siguientes conclusiones:

- El análisis de los referentes teóricos y de los sistemas informáticos estudiados evidenció la necesidad de desarrollar un sistema informático para agilizar la gestión de indicadores en el seguimiento de habilidades de estudiantes potencialmente talentosos.

- Se obtuvo un sistema informático que permite la gestión de indicadores de medición del talento para el seguimiento del desarrollo de las habilidades de los estudiantes potencialmente talentosos de la UCI cumpliendo con los requisitos definidos por el cliente.

- La validación de la investigación se realizó a partir de la aplicación de técnicas, métricas y pruebas que garantizaron el correcto funcionamiento del sistema y demostraron la satisfacción del cliente hacia el sistema desarrollado.

RECOMENDACIONES

Se recomienda:

- Utilizar el sistema desarrollado en el proceso de formación de estudiantes potencialmente talentosos de la Universidad de las Ciencias Informáticas.

- Realizar una adaptación de los indicadores definidos en el sistema, de forma tal que permita el seguimiento del desarrollo de las habilidades en los estudiantes con bajo rendimiento académico.

REFERENCIAS BIBLIOGRÁFICAS

1. Cardona Andújar, J., *Hacia un modelo para la gestión del talento humano en la enseñanza universitaria. Aspectos básicos*, in *Pampedia, No.7*. [En línea. Citado el: 09 - 10- 2014] 2011: ISSN:1870-428X. p. http://www.uv.mx/pampedia/numeros/numero-7/Hacia-un-modelo-para-la-gestion-del-talento-humano-en-la-ense%C3%B1anza-universitaria.pdf.
2. Ortiz Torres, E., Mariño Sánchez, María de los Á. y González Pérez del Villar, Ana María, *La identificación y desarrollo de alumnos talentos en la educación superior*, in *Revista Pedagogía Universitaria Vol. XVIII No. 1 2013*. Centro de Estudios sobre Ciencias de la Educación Superior (CECES). Universidad de Holguín Oscar Lucero Moya [En línea. Citado el: 09-10- 2014] La Habana, Cuba. 2013. p. http://cvi.mes.edu.cu/peduniv/index.php/peduniv/article/download/560/560.
3. Arocas Sánches, E., Martínez Coves, P. y Samper Cayuelas, I., *La respuesta educativa a los alumnos superdotados y/o con talentos específicos*. Ministerio de Educación y Ciencias, Valencia, España. 1994.
4. Alonslo, J., Benito, Y. , *Superdotados:adaptación escolar y social en secundaria*. Narcea, corp. Madrid, España: 1996. ISBN: 84-277-1101-8.
5. Porter, L., *Grifted Young Children A Guide for Teachers and Parent*. Open University Press. 1999.
6. Purcel, J.H., Renzulli , J. S, *The Total Talent Portfolio: A systematic plan to identify and nature gifts and talents*. Mansfield Center, CT: Creative Learning Press. 1998.
7. Gallardo Flores, P., *La Importancia de la ayuda externa en el desarrollo del sujeto talentoso*, in *Tesis en opción al grado de Master en Investigación Educativa*. Instituto Central de Ciencias Pedagógicas, La Habana, Cuba: 2000.
8. Castellanos Simons, D., *Modelo de Intervención educativa para el desarrollo del talento en el contexto escolar*. Universidad Pedagógica de La Habana, La Habana, Cuba. 2005.
9. Pérez Luján, D., González Morales, Dislayne., y Díaz Alfonso, Yoel D., *El talento: antecedentes, modelos, indicadores, condicionamientos, estrategias y proceso de identificación. Una propuesta desde la universidad cubana y el enfoque histórico-cultural*, in *Revista Iberoamericana de Educación. 2005*. [En línea. Citado el: 12-10-2014], 2005: ISSN: 1681-5653. p. http://dialnet.unirioja.es/servlet/articulo?codigo=3159275.
10. Díaz Sardiñas, A., Vázquez Cedeño, Rosa A., *Estrategia de atención masiva y sistemática al talento en la Universidad de las Ciencias Informáticas*. IV Evento Provincial de Educación Superior. Congreso Internacional Universidad 2008, La Habana, Cuba. 2008.
11. Cabero Almenara, J., *El aprendizaje autorregulado como marco teórico para la aplicación educativa de las comunidades virtuales y los entornos personales de aprendizaje*, in *Revista TESI*. Universidad de Salamanca. [En línea. Citado el: 15-01-2015] España. 2013. p.

http://rca.usal.es/index.php/revistatesi/article/download/10217/10626.

12. Vera Salazar, C., *Estrategia didáctica para el desarrollo del aprendizaje de losescolares con talento académico de la Educación Primaria*, in *Tesis en opción al grado científico de Doctora en Ciencias*. Instituto Superior Pedagógico Enrique José Varona, La Habana, Cuba: 2008.
13. Castellanos Simons, D., Martínez Llantada, Marta., Vera Salazar, Caridad., Lorenzo García, Raquel., otros . *Talento: concepciones y estrategias para su desarrollo en el contexto escolar.* Pueblo y Educación. La Habana, Cuba: 2009. ISBN:978-13-1789-6.
14. Menéndez Pérez, J.S., Villanueva Betancourt, Manuel., y Companioni Sardiñas, Yusmary. , *Definición de talento y de talento informático en el marco del Proyecto Talenmático*. Centro de Estudios Internacionales para el Derecho, Buenos Aires, Argentina. 2012.
15. Española., D.d.l.R.A. 2015.
16. Proyecto Talenmático, *Informe del Proyecto Talenmático al Consejo Científico UCI*. Universidad de las Ciencias Informáticas, La Habana, Cuba. 2015.
17. Aguirre Ruiz, A., *Sistema Web para la Identificación de Potenciales Talentos Informáticos en la Programación Competitiva*, in *Trabajo de Diploma para optar por el título de Ingeniero Informático*. Universidad de las Ciencias Informáticas, La Habana, Cuba: 2014.
18. Vallejo Pratts, G., y Rosado Vázquez, Gisela Susana., *Sistema informático para la estimulación del aprendizaje del Cálculo Diferencial e Integral en estudiantes potencialmente talentosos*, in *Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas*. Universidad de las Ciencias Informáticas, La Habana, Cuba: 2014.
19. GrupoCFDeveloper, *DocCF, Software de Gestión Escolar*. [En línea. Citado el: 23-05-2015] 2015. p. http://www.grupocfdeveloper.com/productos.html.
20. Portal Colegios Colombia, *Portal Colegios Colombia*. [En línea. Citado el: 23-05-2015], 2015. p. http://www.portalcolegioscolombia.com/.
21. EVA, *Curso de Moodle para Profesores Editores*. [En línea. Citado el: 19-05-2015], 2015. p. http://eva.uci.cu/course/view.php?id=56.
22. Junhua, Z., Liu, Qingtang., y Zongkai, Yang., *Development of a Moodle course for schoolchildrenIn*, in *Computers & Education. Volume 59, Issue 2, September 2012*. [En línea. Citado el: 23-05-2015] 2012. p. http://www.sciencedirect.com/science/article/pii/S0360131512000097.
23. Moodle, *Acerca de moodle*. [En línea. Citado el: 23-05-2015] 2015. p. https://docs.moodle.org/all/es/Acerca_de_Moodle.
24. INTECO, *Ingeniería del software: metodologías y ciclos de vida*. Instituto Nacional de Tecnologías de la Comunicación. Laboratorio Nacional de Calidad del Software, España, 2009.
25. Virrueta Méndez, A., *Metodologías de desarrollo de software*. Instituto Tecnológico Superior de

Apatzingán. [En línea. Citado el: 14-11-2014], Michoacán, México. 2010. p. http://www.itsapatzingan.edu.mx.

26. Rodríguez Sánchez, T., *Metodología de desarrollo para la Actividad productiva de la UCI*. Universidad de las Ciencias Informáticas, La Habana, Cuba. 2014
27. Pons, C., Giandini, Roxana., y Pérez, Gabriela *Desarrollo de software dirigido por modelos. Conceptos teóricos y su aplicación práctica*. 1era ed. Universidad Nacional de La Plata. Argentina: 2010. ISBN 978-950-34-0630-4.
28. Rumbaugh, J., Jacobson, Ivar., y Booch, Grady., *El Lenguaje Unificado de Modelado. Manual de referencias*. Pearson Educación. Madrid, España: 2002.
29. Kendall, K., y Kendall, Julie., *Análisis y Diseño de Sistemas*. 6ta ed. Pearson. México: 2005. ISBN: 970-26-0577-6.
30. Domingo, I., Rius, G., y Cuenca L., *Una revisión sobre el estado del arte en herramientas de modelado basado en UML.* 6th International Conference on Industrial Engineering and Industrial Management. VI Congreso de Ingeniería de Organización, Vigo. July 18-20, 2012.
31. Kioskea, *Patrones de diseño html*. [En línea. Citado el: 02-12-2014] p. http://es.kioskea.net/contents/224-patrones-de-diseno.
32. Tecnología, *Lenguajes de Programación*. [En línea. Citado el: 02-12-2014]. p. http://www.areatecnologia.com/informatica/lenguajes-de-programacion.html.
33. Eguíluz Pérez, J., *Introducción a XHTML*. 2010.
34. Gauchat, J.D., *El gran libro de HTML5, CSS3 y Javascript*. Marcombo Ediciones Técnicas. Barcelona, España: 2012. ISBN: 978-84-267-1782-5.
35. Eguíluz Pérez, J., *Introducción a CSS*. 2010.
36. W3C, *HTML*. [En línea. Citado el: 07-02-2015], 2015. p. http://www.w3c.es/
37. Eguíluz Pérez, J., *Introducción a JavaScrript*. 2010.
38. Sánchez Maza, M.Á., *JavaScript*. IC Editorial. 2012. ISBN: 978-849-5733184.
39. Welling, L., y Laura Thomson, *Desarrollo Web con PHP y MySQL*. Anaya Multimedia. [En línea. Citado el: 07-02-2015], España. 2010. p. http://www.researchgate.net/publication/44518944_Desarrollo_web_con_PHP_y_MySQL__Luke_Welling_Laura_Thomson.
40. Netbeans, *Netbeans*. [En línea. Citado el: 07-02-2015] 2015. p. http://netbeans.org/index_es.html.
41. Guerrero, C.A., Londoño, Jorge M., Suárez, Johanna M., y Gutiérrez, Luz E., *Estudio comparativo de marcos de trabajo para el desarrollo de software orientado a aspectos*. [En línea. Citado el: 02-12 -2014], La Serena, Colombia. 2014: ISSN 0718-0764. p. http://www.scielo.cl/scielo.php?pid=S0718-07642014000200008&script=sci_arttext.
42. Anglada Martínez, R.A., y Garófalo Hernández, Alain Abel., *Marco de trabajo para el desarrollo de*

herramientas orientadas a la gestión e integración de servicios telemáticos de infraestructura en GNU/Linux, in *Revista Cubana de Ciencias Informáticas vol.7 no.2 , abr.-jun. 2013*. [En línea. Citado el: 18-01-2015] La Habana, Cuba. 2013: ISSN 2227-1899. p. http://scielo.sld.cu/scielo.php?pid=S2227-89920130002000006&script=sci_arttext.

43. Pacheco, N., *Manual de Symfony 2. Release 2.0.1*. 2011.
44. Molina Bastidas, C.V., *Sistema de gestión de pedidos y proformas dinámicas por internet para la empresa. JIMEMOR CIA.LTDA utilizando Symfony*, in *Trabajo de diploma para optar por el título de Ingeniería en Sistemas Computacionales*. Universidad Técnica del Norte, Ibarra, Ecuador: 2012. p. http://repositorio.utn.edu.ec/bitstream/123456789/1814/1/Tesis%20formato%20Pdf.pdf.
45. Eguíluz Pérez, J., *Desarrollo web ágil con Symfony 2*. 2014.
46. Ortiz Batista, Y., *Sistema integrado de gestión de Ciencia, Tecnología e Innovación en la Universidad de las Ciencias Informáticas*, in *Trabajo final presentado en opción al título de Máster en Informática Aplicada*. Universidad de las Ciencias Informáticas, La Habana, Cuba: 2011. p. http://repositorio_institucional.uci.cu/jspui/bitstream/ident/8016/1/TM_04853_11.pdf.
47. Bazaco Caloto, J., *Sistema de registro de asistencia de alumnos y profesores*, in *Trabajo fin de grado*. Escuela Tecnica Superior de Ingeniería de Telecomunicacion. Universidad Rey Juan Carlos, España: 2014. p. http://gsyc.urjc.es/~grex/pfcs/2014-jorge-bazaco/JorgeBazaco-SistemaRegistroAsistenciaAlumnosYProfesores.pdf.
48. García Lara, E., *Diseño e implementación del Back-End del sitio web del proyecto Auralizarte. HTML5, JQUERY, PHP, SQLITE*, in *Titulación: Ingeniero Técnico de Telecomunicación Especialidad Sonido e Imagen*. Escuela Técnica Superior de Ingenieros Industriales y de Telecomunicación, Pamplona: 2014. p. http://academica-e.unavarra.es/bitstream/handle/2454/9487/629095.pdf?sequence=1.
49. Ecured., *Servidor de aplicaciones*. [En línea. Citado el: 28-01-2015] p. http://www.ecured.cu/index.php/servidor_aplicaciones.
50. Germán Agüero, R.M., y Suau Lovelle, José Manuel., *Herramienta para la administración de cluster en servidores de aplicaciones JBoss*, in *Trabajo de Diploma para optar por el título de Ingeniero Ciencias Informáticas*. Universidad de las Ciencias Informáticas, La Habana, Cuba 2013. p. http://repositorio_institucional.uci.cu/jspui/handle/ident/8516.
51. González-Vallés Saco, G., *Una Introducción a APACHE*. 2014.
52. Reyes, T., *Sistema de gestión de información odontológica utilizando orm para el departamento de bienestar universitario de la utn*. [En línea. Citado el: 27-01-2015] 2011. p. http://repositorio.utn.edu.ec/bitstream/123456789/571/1/Tesis.pdf.
53. Alvial Cid, R., Saavedra Quevedo, Bernardita., y Valenzuela Parada Victor., *ORM - Object Relational Mapping (Mapeo Objeto Relacional)*. [En línea. Citado el: 27-01-2015] 2011. p.

https://informaticaviernes2011.files.wordpress.com/2011/07/orm_victor_raquel_berni.pdf.

54. Flores Muñoz, V., *Desarrollo de página web para cliente de DIACODE*, in *Proyecto fin de carrera Ingeniería Técnica Informática de Gestión*. Universidad Carlos III de Madrid. Escuela Politécnica Superior, España. 2012. p. http://e-archivo.uc3m.es/bitstream/handle/10016/14767/PresentacionPFC_Victor_Flores.pdf?sequence=1 ..
55. Ecured., *SGBD*. [En línea. Citado el: 28-01-2015] 2015. p. http://www.ecured.cu/index.php/SGBD.
56. Cristiá Alvarez, A., Morgado Sánchez, Yeni., Ortíz Valmaseda, Marcos Luis., Vazquez Ortíz, Yudisney. , *Guía para la personalización de PostgreSQL 8.4*. Centro de Tecnologías de Gestión de Datos (DATEC) Universidad de las Ciencias Informáticas, La Habana, Cuba. 2010.
57. Padrón Ortíz, F.M., *PgAdmin III. Administrador de base de datos open source PostgreSQL*, in *Investigación formativa previa a la obtención del título de Ingeniero de Sistemas*. Universidad Católica de Cuenca, Cuenca, Ecuador: 2013. p. http://dspace.ucacue.edu.ec/bitstream/reducacue/5629/1/PGADMIN%20III%20Administrador%20de%20Base%20de%20Datos%20Open%20Source%20PostgreSQL.pdf.
58. Hernández Aguilar, V., Fernández Pérez, Yamilis., y Díaz Vera, Julio Cesar., *Protofase a la ingeniería de requisitos para facilitar la comprensión del negocio a informatizar en el desarrollo de software de gestión*. Eighth LACCEI Latin American and Caribbean Conference for Engineering and Technology. 2010, Arequipa, Perú. 2010. p. http://cybertesis.urp.edu.pe/ponencias/LACCEI_2010/Papers/Papers_pdf/IT042_Hernandez.
59. Guibert Estrada, L., y Altuna Castillo, Enrique José., *Ingeniería de requisitos del software educativo "Mis Mejores Cuentos"*. Centro de Tecnologías para la Formación. Universidad de las Ciencias Informáticas. [En línea. Citado el: 14-03-2015], La Habana, Cuba. 2011. p. http://gte2.uib.es/edutec/sites/default/files/congresos/edutec11/Ponencias/Mesa%205/Guibert%20-%20Altuna-%20Edutec%202011.F.pdf.
60. MTP., *Software Quality Assurance. Ingeniería de Requisitos*. [En línea. Citado el: 16-03-2015] Madrid, España. 2015. p. http://www.mtp.es/empresa.
61. Pressman, R.S., *Software engineering: a practitioner's approach*. 7th ed. McGraw-Hil. New York, EUA.: 2010. ISBN:978-0-07-337597.
62. Sommerville, I., *Ingeniería de software*. Novena ed. Pearson. España: 2011. ISBN: 9786073206044.
63. Martínez Guerrero, J.M., y Silva Delgado, Camilo Andrés., *Guía Metodológica. Levantamiento y análisis de Requerimientos de Software con base en procesos de negocio*, in *Guía metodológica elaborada para cumplir con uno de los requisitos para optar al título de Ingeniero de Sistemas*. Pontificia Universidad Javeriana, Bogotá D.C, Colombia: 2010. p.

http://pegasus.javeriana.edu.co/~CIS1010IS06/descargas/Anexos/Marco%20Teorico/Anexo%205.%20Gu%C3%ADa%20Metodol%C3%B3gica%20para%20el%20Levantamiento%20y%20An%C3%A1lisis%20de%20Requerimientos%20de%20Software%20en%20base%20a%20Procesos%20de%20Negocio.pdf.

64. MADEJA., *Especificación de Requisitos del Sistema*. Marco de Desarrollo de la Junta de Andalucía. [En línea. Citado el: 26-03-2015] Andalucía, España. 2015. p. http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/407
65. MADEJA., *Validar los requisitos del sistema*. Marco de Desarrollo de la Junta de Andalucía. [En línea. Citado el: 24-04-2015], Andalucía, España. 2015. p. http://www.juntadeandalucia.es/servicios/madeja/contenido/libro-pautas/185
66. Tahuiton Mora, J., *Arquitectura de software para aplicaciones Web*, in *Tesis presentada para obtener el grado de Maestro en Ciencias en Computación*. Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, México D.F. México: 2011. p. http://delta.cs.cinvestav.mx/~pmalvarez/tesis-tahuiton.pdf.
67. Venete, A., *Introducción a los Patrones de Arquitectura*. Universidad Jaume I. [En línea. Citado el: 18-04-2015] 2011. p. http://mahara.uji.es/artefact/file/download.php?file=54534&view=4648.
68. NTAULA., *Análisis y diseño de sistemas. Diagramación: Modelo de datos*. [En línea. Citado el: 18-03-2015] 2015. p. http://ntaula0.tripod.com/ads/anamdatos.html.
69. Martillo Hidalgo, L.S., y Mora Rodríguez, Diego Fernando. , *Análisis, desarrollo e implementación de un sistema para la gestión académica y administrativa de la Unidad Educativa Salesiana Santa María Mazzarello de Guayaquil*, in *Tesis de grado previa a la obtención del título de Ingeniero de Sistemas*. Universidad Politécnica Salesiana, Guayaquil, Ecuador: 2013. p. http://dspace.ups.edu.ec/bitstream/123456789/4527/1/UPS-GT000397.pdf.
70. Pérez Pérez, S.L., *Fundamentos de Ingeniería de Software. Patrones de diseño*. Universidad Autónoma de México. [En línea. Citado el: 22-03-2015] México D. F. México. 2013. p. http://computacion.cs.cinvestav.mx/~sperez/cursos/fis/PatronesDiseno.pdf.
71. Guerrero, C.A., Suárez, Johanna M., y Gutiérrez, Luz E., *Patrones de Diseño GOF (The Gang of Four) en el contexto de Procesos de Desarrollo de Aplicaciones Orientadas a la Web*, in *Revista Información Tecnológica Vol. 24 No.3, 2013*. [En línea. Citado el: 25-03-2015] La Serena, Colombia. 2013: ISSN 0718-0764. p. http://www.scielo.cl/scielo.php?pid=S0718-07642013000300012&script=sci_arttext.
72. Universidad Politécnica de Madrid, *Patrones del "Gang of Four"*. Unidad Docente de Ingeniería del Software. Facultad de informática - Universidad Politécnica de Madrid. [En línea. Citado el: 26-03-2015], España, 2013. p. http://www.pdfbook-s.com/gang-of-four/1/.
73. Sanz Campos, D., y Rueda Flores, Jorge Alberto. , *Diseño e implementación del módulo*

"Inscripción" del Sistema para la Gestión de Antecedentes Penales (SIGESAP), in *Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas*. Universidad de las Ciencias Informáticas, La Habana, Cuba: 2012. p. http://bibliodoc.uci.cu/RDigitales/2012/diciembre/3/TD_05408_12.pdf.

74. UNAD., *Lección 29. Diagramas de componentes*. Universidad Nacional Abierta y a Distancia. [En línea. Citado el: 28-04-2015], Colombia. 2015. p. http://datateca.unad.edu.co/contenidos/200609/exeuml/leccin_29_diagramas_de_componentes.html
75. Serna, S., *Especificación formal de requisitos temporales no funcionales*, in *Tesis para optar el título de Magister en Ingeniería de Sistemas*. Universidad Nacional de Colombia. [En línea. Citado el: 26-03 -2015], Colombia: 2011. p. http://www.bdigital.unal.edu.co/3928/1/98553427_2011_1.pdf.
76. Palma Osoria, D., *Módulo para la extracción y representación de los metadatos en el Sistema de Gestión Documental de audio y video digitales TeVeo Plus V1.0*, in *Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas*. Universidad de las Ciencias Informáticas, La Habana, Cuba: 2013. p. http://repositorio_institucional.uci.cu/jspui/bitstream/ident/8302/1/TD_06469_13.pdf.
77. López Takeyas, B., *Nomenclatura sugerida para identificar los componentes de un proyecto*. Instituto Tecnológico de Nuevo Laredo. [En línea. Citado el: 12-05-2015] México. 2015. p. http://www.itnuevolaredo.edu.mx/maestros/sis_com/takeyas/Apuntes/POO/Apuntes/02.-%20NomenclaturaComponentesProyecto.pdf.
78. Plaza Rubio, J., *Modelos de Fiabilidad del Software*, in *Ingeniería Técnica en Informática de Gestión*. Universidad de Valladolid, España: 2013.
79. Rodríguez Tello, E.A., *Estrategias y técnicas de prueba del software.* CINVESTAV, Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional. [En línea. Citado el: 15-04-2015], Tamaulipas, México. 2012. p. www.tamps.cinvestav.mx/~ertello/swe/sesion15.pdf.
80. Moreno Martín, M.Á., *Filosofía Lean aplicada a la Ingeniería del Software. Universidad de Sevilla*. Universidad de Sevilla. [En línea. Citado el: 18-04-2015] España. 2013. p. http://bibing.us.es/proyectos/abreproy/70201/descargar_fichero/02+-+Ingenieria+del+Software.
81. Silega Martínez, N., *Método para la transformación automatizada de modelos de procesos de negocio a modelos de componentes para sistemas de gestión empresarial*, in *Tesis presentada en opción al grado científico de Doctor en Ciencias Técnicas*. Universidad de las Ciencias Informáticas, La Habana, Cuba: 2014. p. http://repositorio_institucional.uci.cu/jspui/bitstream/ident/8584/2/Tesis_Nemury_V14.pdf.