# Universidad de las Ciencias Informáticas
**Facultad 3**


# Título: "Sistema de soporte a la decisión para la ubicación del estudiante en un rol del proceso de desarrollo de software"

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

**Autor(es):** Leanet Arza Pérez

Dianela Borrego León

**Tutor(es):** Ing. Lizandra Arza Pérez

Dr. Edistio Yoel Verdecia Martínez

**Fecha:** 15/12/2011

## DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores de este trabajo y autorizamos al Macro proyecto de investigación: Modelo de integración docencia-producción-investigación de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los _28_ días del mes de _Junio__ del año _2012_.

| | |
|---|---|
| _____ | _____ |
| Leanet Arza Pérez | Dianela Borrego León |
| Autor | Autor |
| _____ | _____ |
| Lizandra Arza Pérez | Edistio Yoel Verdecia Martínez |
| Tutor | Tutor |

## DATOS DE CONTACTO

**Lizandra Arza Pérez:** Graduada de Ingeniería Informática en la Cujae, año 2002. Profesora Asistente. Subdirectora de Investigaciones de Calisoft.

**Edistio Yoel Verdecia Martínez:** Graduado de Ciencias de la Computación en la UH, 1996. Profesor Auxiliar, Doctor en Ciencias Pedagógicas.

## AGRADECIMIENTOS

A la Revolución, por permitirnos la educación y formación profesional, por hacernos personas dignas de estos tiempos.

A la UCI, por ser nuestro hogar, por darnos la posibilidad de crecernos como personas, por educarnos, por darnos tantos amigos.

A Lizandra, por confiar en nosotras, por asesorarnos y por ser la mejor tutora que pudiéramos tener.

A Edistio, por su consejo y su guía como tutor.

A Francisco, por brindarnos su ayuda sin conocernos apenas.

A Isko, por su ayuda y paciencia.

A todas las amistades que estuvieron a nuestro lado en las buenas y en las malas.

A todos los que de una forma u otra nos ayudaron.

A todo el que alguna vez nos dijo: "¿Y la tesis?"

Gracias

## AGRADECIMIENTOS

A mi mamá, por ser mi amiga, por dejarme tomar sola el camino aunque muchas veces no fuese el correcto; a mi papá, por sus regaños constructivos y su apoyo incondicional; a ambos les agradezco haber confiado en mí y no haber borrado el sueño de ver llegar este momento.

A mi tata, por preocuparse y darme muestras de su cariño a diario.

A mis abus, por estar orgullosos de sus nietos, en especial a abuelito Ramón que está siempre en mi corazón.

A mis primos y tíos.

A tío Luis, por querer a sus sobrinos como hijos.

A Niurma, a Ani y a prima Evelyn, por su amistad sin límites y sin tachaduras, aunque el tiempo haya logrado que cada una tomase caminos distintos.

A Lotti, Saily y Yohan por estar junto a mí en los momentos más difíciles, por entenderme y quererme como soy.

A mi novio, por superar mis momentos de estrés y por quererme.

A Andy, a Berna, a Carli, al Dani y a Eddy por haberme brindado su ayuda siempre que la necesité.

A Sergi, por sus momentos de humor, con los cuales o calmaba las tensiones o las incrementaba.

A Liza, a Isko y a Clau por dejar que me sintiera parte de la familia.

A mi compañera de tesis, Lea, mi más antigua amiga en la universidad, quien ha estado junto a mí y con quien he compartido cada uno de estos 5 años, para ella mi agradecimiento por dejarme conocer a la magnífica persona que es...

**Nela**

A mi mamita por mostrarme el camino a seguir, por enseñarme a ver lo mejor de cada persona y ser un ejemplo de mujer y madre.

A mi papito por estar siempre conmigo, por demostrarme que si se lucha se logra lo que se quiere, por ser un gran padre y amigo.

A mi abuelita por cuidarme y quererme tanto.

A papito y mima por todo el cariño que me dieron.

A mis hermanas y hermanitos: a Lizandra que ha sido una segunda madre con su apoyo, confianza, cuidados, paciencia. A Laura por ser una gran amiga, por su cariño y todos los momentos juntas. A Agustincito por quererme tanto y desde chiquitico ser mi confidente y amigo. A Yiyo por darme su cariño y amor de hermano.

A mis sobrinos Claudita y Bryan por dar momentos tan felices a mi vida.

A Sergito por darme tanto amor y cuidarme, por su paciencia y apoyo incondicional.

A Mary y Reco por ser como padres para mí.

A Luisito, Javier y Emilito por ser mis hermanos.

A Cuqui, Luis, Ana y Grisel por su cariño y amor.

A Lourdes, Tere, Rafael y Rafelitín por ser como otra familia para mí, agradecerles por estar siempre ahí.

A Iskael por sus concejos y ayudas, por aguantarme.

A Ada y Domingo por ser como unos abuelos.

A Lily, Ailyn, Dianeisy, Denia, Yadanis y Marian por su amistad incondicional, estando siempre a mi lado.

A Lotti, Saily y Yohan por todos los buenos momentos que me han brindado durante estos 5 años, estando conmigo en las buenas y malas.

A mi compañera de tesis, Nela, por ser gran amiga y aconsejarme en los momentos más difíciles, por su ayuda, paciencia y dedicación en la realización de la tesis.

A Raúl y mis demás amigos de ciego por brindarme momentos tan especiales en mi vida.

A Edel y Yanet por ser buenos amigos y por su cariño.

A Dayana por darme su cariño y amistad.

**Lea**

## DEDICATORIA

***A mami y papi, a mi tata y a toda mi familia, a mis amigos... Dianela***

***A mami y papi, a abuela, papito y mima, a mis hermanos y hermanas y a mis queridos sobrinos… Leanet***

# RESUMEN

El modelo de integración formación-producción-investigación divide la docencia del estudiante en dos ciclos, el ciclo básico (CB) y el ciclo profesional (CP). En el ciclo básico se plantea la formación intensiva en las disciplinas fundamentales para el ejercicio de la profesión, para luego, en el ciclo profesional desempeñarse en diferentes roles del proceso de desarrollo de software, vinculado a los proyectos que se llevan a cabo en la Universidad de las Ciencias Informáticas (UCI). Para la ubicación del estudiante en un rol del proceso de desarrollo de software se ha diseñado un modelo que integra técnicas de inteligencia artificial para el análisis de la información, con el propósito de obtener criterios para su ubicación. Unido a este modelo, se ha definido una metodología para desarrollar el proceso de ubicación del estudiante en un rol del proceso de desarrollo de software.

Este modelo y metodología están siendo introducidos en la práctica, pero se tienen inconvenientes para su uso de manera generalizada en la universidad, los cuales están sobre la base de los grandes volúmenes de información a procesar y el manejo de las técnicas de inteligencia artificial que se necesitan aplicar. Es por ello que se ha planteado una investigación para la implementación de un sistema de apoyo a la decisión que permita la aplicación de la metodología propuesta.

Se presentan en este informe la documentación referente al desarrollo de un sistema de soporte a la decisión para la ubicación de los estudiantes en un rol del proceso de desarrollo de software. El sistema ha sido desarrollado siguiendo la metodología SXP, sistema gestor de base de datos Postgres SQL, entorno integrado de desarrollo el Eclipse.

## PALABRAS CLAVE

Sistemas de apoyo a la decisión, metodología SXP, proceso de ubicación, selección de personal

# ÍNDICE

## ÍNDICE DE FIGURAS



## ÍNDICE DE TABLAS

# INTRODUCCIÓN

La Industria de Software y Servicios Informáticos (ISWSI) basa sus producciones en el empleo de los conocimientos de los recursos humanos que la componen sobre las tecnologías existentes. La preparación y conocimiento de las personas que laboran en ella es lo que distingue el éxito en esta rama entre las diferentes empresas líderes en la actualidad, siendo entonces la gestión eficiente de los Recursos Humanos un factor clave de éxito para las organizaciones de software.

El desarrollo de software es un proceso que involucra a varias personas y en el cual es significativo la conformación del equipo, el establecimiento de roles dentro de los equipos de trabajo, la conceptualización de estos roles, la definición de las competencias para el desempeño de cada uno de ellos, y su evaluación, son elementos fundamentales para el desarrollo de esta industria. [1, 2]

En la Guía de los Fundamentos para la Dirección de Proyectos [3] se establece como una de las áreas de conocimiento de la Dirección de Proyectos, la Gestión de los Recursos Humanos, destacando la importancia que juegan en el desarrollo de un proyecto de software.

La Universidad de las Ciencias Informáticas (UCI) es parte de la Industria Cubana de Software, participa en ella como proveedora de profesionales para la industria y como ente activo con numerosos compromisos nacionales e internacionales. La participación en los proyectos es un elemento clave en la formación de los futuros egresados de Ingeniería en Ciencias Informáticas.

La UCI es considerada una universidad productiva, su misión declarada en su proyecto estratégico, es ser: *"Universidad innovadora de excelencia científica, académica y productiva que forma de manera continua profesionales integrales comprometidos con la Patria, soporte de la informatización del país y la competitividad internacional de la industria cubana del software" [4]*. Dada la complejidad de esta misión se aplica un modelo para integrar sus procesos fundamentales: la docencia, la producción y la investigación. Este modelo de integración divide la formación en dos ciclos, cada uno con características bien definidas:

- Ciclo Básico (CB): ocupa los primeros cinco semestres de la carrera, caracterizado por una formación académica intensiva.
- Ciclo Profesional (CP): compuesto por los restantes semestres de la carrera, donde el estudiante se forma desde el desempeño de los roles profesionales en el proyecto.

La manera en la que se realiza el proceso de ubicación de estudiantes en un rol del proceso de desarrollo de software, no satisface totalmente las necesidades de la formación y la producción, existe información y evidencias del desarrollo del estudiante, pero se hace muy engorrosa su utilización, por el volumen de la información a procesar y por la forma en que se presentan los resultados. En la mayoría de los casos los líderes de proyecto se guían solamente por los resultados académicos.

Es por ello que en la universidad se ha trabajado en la construcción de un modelo empleando técnicas de inteligencia artificial para analizar la información disponible de los estudiantes y los roles, y brindar criterios que ayuden a definir su ubicación en el rol para el cual es más adecuado. En este modelo se definen funciones matemáticas que son de engorrosa aplicación, se representa la información empleando la teoría de conjuntos borrosos y se trabaja con grandes volúmenes de información. La cantidad de estudiantes a ubicar es grande, lo cual también complejiza el ordenamiento de los estudiantes por roles. Se define además, una metodología que permite la concreción del modelo en el proceso de ubicación de los estudiantes en un rol del proceso de desarrollo en la UCI.

Partiendo de que la información que se tiene del estudiante se encuentra digital en un grupo de sistemas desarrollados en la universidad y teniendo en cuenta los elementos antes expuestos, se hace necesario el desarrollo de una aplicación informática que implemente los algoritmos necesarios para la aplicación de la metodología y mejore sus tiempos de ejecución.

Por lo que se describe el siguiente **problema a resolver**: *Cómo implementar el modelo matemático propuesto para la evaluación de los estudiantes, con vistas a brindar los criterios para su ubicación en un rol del proceso de desarrollo de software, como parte del ciclo profesional, de manera que se facilite el trabajo con grandes cantidades de estudiantes.*

Como **objeto de estudio** se define los *Sistemas de apoyo a la decisión* y como **campo de acción** el *proceso de ubicación de estudiantes en roles del proceso de desarrollo de software en la UCI.*

Se define como **objetivo general** de la presente investigación: *Realizar el análisis, diseño e implementación del sistema que implemente el modelo matemático para la obtención de los criterios para la ubicación de los estudiantes en un rol del proceso de desarrollo de software.*

A partir de un análisis del objetivo general se derivan los siguientes **objetivos específicos**:

1. Establecer el marco teórico referencial sobre los sistemas de apoyo a la decisión y el proceso de ubicación de estudiantes en la UCI
2. Modelar el sistema informático para la ayuda a la toma de decisiones en la ubicación de los estudiantes en un rol del proyecto.
3. Desarrollar el sistema informático para la ayuda a la toma de decisiones en la ubicación de los estudiantes en un rol del proyecto.
4. Validar la aplicación propuesta.

Para dar cumplimiento a los objetivos específicos anteriores se diseñaron las siguientes **tareas de investigación**:

1. Recopilación y estudio de la bibliografía referente a los sistemas de apoyo a la decisión, la selección de personal y el proceso de selección.

2. Análisis de la bibliografía para conocer las experiencias existentes en herramientas desarrolladas para la toma de decisiones.
3. Modelación del sistema a desarrollar.
4. Realización del análisis y diseño del sistema.
5. Descripción de los principales algoritmos.
6. Implementación del sistema modelado.
7. Validación del sistema mediante métricas TOC y RC.
8. Diseño de los casos de prueba para probar el sistema.
9. Análisis de los resultados de las validaciones.

Se plantea la siguiente **idea a defender** para la investigación: *Si se desarrolla un sistema de apoyo a la decisión que implemente el modelo matemático para la ubicación del estudiante en un rol del proceso de desarrollo de software, se logrará facilitar el trabajo con la gran cantidad de información involucrada en el proceso de ubicación.*

Los **métodos y técnicas** empleados fueron los siguientes:

- ✓ ***Métodos Teóricos***

Permiten descubrir en el objeto de investigación las relaciones esenciales y las cualidades fundamentales, no detectables de manera senso-perceptual. Por ello se apoya básicamente en los procesos de abstracción, análisis, síntesis, inducción y deducción.[5]

**Análisis-Síntesis:** Mediante este método se descompone un objeto, fenómeno o proceso en los principales elementos que lo integran para analizar, valorar y conocer sus particularidades.[6] En este caso se utilizó para el análisis de la información y para llegar a conclusiones sobre el estudio realizado del proceso de ubicación.

**Histórico-Lógico:** A través de este método se establece la necesaria correspondencia entre los elementos de los métodos lógico e histórico, proyectando el análisis de la evolución histórica de los fenómenos, con la proyección lógica de su comportamiento futuro.[6] En esta investigación se le dio utilidad para establecer la evolución y tendencia de los sistemas de apoyo a la decisión.

**Modelación:** Este consiste en la representación ya sea material o teórica de los objetos, fenómenos, o particularidades de estos, lo que permite descomponerlos, abstraer determinadas cualidades, operar y experimentar con él.[6] Para el entendimiento de la metodología y el modelo propuesto se hace uso de este método de la investigación.

- ✓ ***Métodos Empíricos***

Estos métodos posibilitan revelar las relaciones esenciales y las características fundamentales del objeto de estudio, accesibles a la detección de la percepción, a través de procedimientos prácticos con el objeto y diversos medios de estudio.[7]

**Observación:** Es un método para reunir información visual sobre lo que ocurre, lo que el objeto de estudio hace o cómo se comporta.[8] En este caso se utilizó para entender cómo se realiza hoy en la universidad el proceso de ubicación de los estudiantes en un rol del proceso de desarrollo de software.

**Análisis de documentos:** Se estudiaron los documentos relacionados con la metodología y el modelo a implementar.

- ✓ ***Técnicas de recopilación de información***

La recopilación de datos se utiliza para verificar los métodos empleados en lo investigado, para llegar a la conclusión del suceso, teniendo las pruebas y una serie de pasos que se llevan a cabo para comprobar la hipótesis planteada.[9]

**Entrevistas:** Es una técnica para obtener datos, que consiste en un diálogo entre dos personas.[9] En este caso se empleó con el fin de obtener información referente al modelo y la metodología a implementar, además de obtener datos referentes a la forma en que se realiza la ubicación de los estudiantes en un rol del proceso de desarrollo de software.

El documento de tesis está dividido en Introducción, el Capítulo 1 para establecer el marco teórico referencial de la investigación, Capítulo 2 donde se presentan los resultados de la modelación y análisis del sistema, Capítulo 3 para la descripción de la implementación del sistema y los resultados de la validación de la aplicación desarrollada, Conclusiones, Recomendaciones, Bibliografía y el cuerpo de Anexos.

# CAPÍTULO 1: MARCO TEÓRICO REFERENCIAL

Para el desarrollo de la presente investigación se hizo necesaria la revisión, estudio y análisis de diversos temas, que permitieron establecer el marco teórico referencial, para la obtención de los resultados que dan cumplimiento a los objetivos propuestos. Los principales temas abordados están relacionados con el objeto de estudio y campo de acción definidos en el diseño de la investigación.

En el presente capítulo se muestra el análisis realizado por las autoras, a partir del estudio sobre la toma de decisión y los sistemas de soporte a la decisión. De igual forma se realiza un análisis del proceso de ubicación de los estudiantes en roles del proceso de desarrollo de software en la UCI, así como la metodología y el modelo para su realización. El entendimiento de las características del proceso de selección de personal, según los referentes de la industria, son importantes para comprender tanto la metodología como el modelo que se han definido en la universidad; es por ello que se ha realizado una revisión de los principales conceptos en este sentido.

Por último, para el desarrollo de la aplicación es necesario definir la metodología de desarrollo y herramientas a utilizar, mostrándose en el último epígrafe la revisión y análisis realizado para ello.

## 1.1. Sistemas de Soporte a la Decisión.

Según [10] el proceso de tomar una decisión es una de las actividades que se realiza en el mundo de los negocios con mayor frecuencia. La misma se presenta en todos los niveles de la organización, ya sean asistentes, auxiliares, o directores generales de las empresas. En cualquier caso se tiene uno o varios objetivos a cumplir teniendo en cuenta un conjunto de restricciones. Los Sistemas de Soporte a la Decisión tienen como propósito apoyar y facilitar este proceso, a través del procesamiento de la información disponible para brindar criterios oportunos y confiables de relevancia para los decisores.

### 1.1.1. Proceso de toma de decisión.

Una investigación realizada por [11] plantea que la decisión comienza al aparecer un problema que afecta a un individuo y teniendo en cuenta que, para resolverlo, dispone varias alternativas de solución. La decisión es un acto del presente cuyo resultado obtendrá un efecto en el futuro. El decisor toma la decisión de acuerdo al enfoque que se posee de dicho problema y basándose en la experiencia del pasado. El acto de decidir es un trabajo intelectual, ya que es construido mentalmente por el que toma la decisión, basándose en sus experiencias, conocimientos y expectativas.

Otras definiciones de decisión son las planteadas por [12] que define "*una decisión como el proceso de elegir la solución para un problema suponiendo que existen varias alternativas*" y la planteada por [13] "*la toma de decisiones es un proceso de selección entre cursos alternativos de acción, basado en un conjunto de criterios, para alcanzar uno o más objetivos"*. Se puede destacar después de esta revisión,

que con diferentes palabras pero de manera general las distintas definiciones tienen elementos comunes, entre los que se señalan los siguientes:

- Se elige o selecciona.
- En función de criterios.
- Entre un grupo de alternativas.
- Para lograr un objetivo.

El proceso de toma de decisiones según [14] agrupa actividades que van desde la identificación de un problema hasta la solución del mismo haciendo uso de la alternativa seleccionada, por lo que se concluye que es un proceso enmarcado en problemas donde se deben encontrar las alternativas para su solución.

El mismo autor [14] plantea que la toma de decisión se desarrolla bajo determinadas circunstancias, teniendo en cuenta las características de la información que se maneja y las posibles alternativas que se tienen, estableciendo las siguientes categorías:

- Toma de decisiones bajo certidumbre: En la que se conocen los datos de forma determinista, toda la información que se debe manejar en el proceso se tiene con total certeza, de Fuentes confiables.
- Toma de decisiones bajo riesgo. En la que los datos se describen mediante distribuciones de probabilidad.
- Toma de decisiones bajo incertidumbre: En los que la información que se conoce no es precisa, existe vaguedad e incertidumbre en cuanto a la validez de la misma, no es posible representar la información de manera determinista.

Estas clasificaciones es importante conocerlas, para entender qué características tiene la información que se maneja en el proceso de ubicación de acuerdo a la clasificación que la autora del modelo le da a este problema de toma de decisión.

En [14] el proceso de resolución de problemas consta de siete fases, siendo la toma de decisiones el término que se asocia a las cinco primeras etapas de dicho proceso; comenzando con la identificación y definición del problema y concluyendo con la elección de una alternativa.


**Figura 1: Actividades del Proceso de Resolución de Problemas (Elaboración Propia)**

Las tres primeras fases del proceso constituyen la “Estructuración del problema” y las dos últimas fases son el “Análisis del problema”. La fase de análisis del proceso de toma de decisiones puede asumir dos formas básicas: cualitativa y cuantitativa. El análisis cualitativo se basa primordialmente en

el razonamiento y la experiencia del decisor; incluye la impresión intuitiva que el decisor tiene del problema. Cuando se utiliza el enfoque cuantitativo, el analista se concentra en los hechos o datos asociados al problema y desarrolla expresiones matemáticas que describen los objetivos, las restricciones y las relaciones existentes en el problema. Después utilizando uno o más métodos cuantitativos, el analista ofrece una recomendación con base en los aspectos cuantitativos del problema.


**Figura 2: Actividades del Proceso de Toma de Decisión**

Se establece en [14] que a los problemas que no implican más de un criterio de decisión se le denomina problemas de decisión de criterio único y en el caso contrario se les denomina problemas de criterios múltiples o problemas de decisión multicriterio. En el caso que se desea abordar se trata de un problema multicriterios, existen diversos criterios a tener en cuenta para tomar la decisión.

### 1.1.2. Sistemas de soporte a la toma de decisiones.

El término "sistema de soporte a las decisiones" (SSD) ha sido utilizado de maneras diferentes, y ha sido definido de varias formas en dependencia del punto de vista de cada autor. Por ejemplo, Finlay lo define como *"un sistema basado en computadora que soporta el proceso de toma de decisiones"* [15]. Turban lo define más específicamente como *"un sistema de información basado en computadora, especialmente desarrollado para dar apoyo a la solución de un problema no estructurado de dirección para mejorar la toma de decisiones. Utiliza datos, proporciona una interfaz fácil de utilizar y permite considerar la propia visión del decisor"* [16].

Para [17], *"un SSD es un modelo basado en establecer procedimientos para el procesamiento de datos y juicios, con el fin de asistir a un director en la elaboración de una decisión"*. Keen plantea que *"un SSD combina los recursos intelectuales de los individuos con las capacidades de una computadora para mejorar la calidad de las decisiones"*[18]. Moorelo los define como *"sistemas extensibles capaces de soportar análisis de conocimiento"*[19]. De acuerdo a [20], el término "sistema de soporte a las decisiones" permanece como un término inclusivo y útil para muchos tipos de sistemas de información que apoyan la toma de decisiones.

Al estudiar las definiciones planteadas anteriormente, se puede concluir que existe diversidad de definiciones de los SSD, sin embargo, todas ellas tienen elementos en común que las autoras han considerado característicos de los SSD, y que son de importante conocimiento para el desarrollo de la investigación. Estos elementos son:

- Son sistemas informáticos.
- Utilizan datos e información que se procesa y analiza.
- Permiten combinar las capacidades de la computadora con las de intelecto humano.
- Ayudan a la solución de problemas.
- Asisten a los decisores brindando criterios para la decisión.

El objetivo del mismo es proporcionar la mayor cantidad de información relevante en el menor tiempo posible, con el fin de facilitar la toma de la decisión más adecuada. Para profundizar un poco en la teoría de los sistemas de soporte a la decisión se describen a continuación un grupo de clasificaciones y tipos de ayuda que pueden brindar estos sistemas. Estos elementos serán útiles para establecer las características que el sistema a desarrollar debe tener de acuerdo al problema tratado.

Es importante que se tenga en cuenta las formas en la que estos sistemas pueden apoyar a los decisores, según se plantea en [21]. Entre ellas:

- Alerta al usuario para una toma de decisiones oportuna.
- Reconoce los problemas que deban resolverse en el marco del proceso de toma de decisiones.
- Resuelve los problemas reconocidos por sí mismo o por el usuario.
- Se le facilita al usuario la capacidad para procesar el conocimiento.
- Se le ofrece al usuario asesoramiento, expectativas, evaluaciones, hechos, análisis y diseños.
- Simula la percepción, la imaginación o la visión creativa de los usuarios.
- Coordina o facilita las interacciones entre los participantes y los decisores.

El tipo de ayuda a brindar por el sistema desarrollado debe definirse a partir del conocimiento del problema y la información para su solución, así como las expectativas de los decisores respecto a lo que desean que se realice en la aplicación.

#### *1.1.2.1. Clasificación*

Al igual que las definiciones, existe diversidad de criterios para clasificar los SSD.

Según la relación con el usuario [22]:

- Sistemas cooperativos: permite al decisor modificar, completar o refinar las sugerencias ofrecidas por el sistema y las envía para su validación, haciendo que el proceso comience de nuevo cuando se genere una solución consolidada.
- Activos: presenta sugerencias y soluciones explícitas.
- Pasivos: que no puede presentar sugerencias o soluciones explícitas.

Según la naturaleza de las decisiones [23]:

- Estructurados: Son aquellos en los que se pueden definir reglas y procedimientos de decisión preestablecidos, las variables que intervienen son conocidas y repetitivas.
- No estructurados: No son repetitivas y no poseen reglas predefinidas, no existen puntos de referencia idénticos en el pasado, las reglas deben ser creadas para cada ocasión. El papel del sistema de información en estos casos es de apoyo al decisor.

Según la evolución que han experimentado [24]:

- Sistemas de soporte orientados al procesamiento de datos: EDP, OAS, MIS.
- Sistemas de soporte orientados al análisis de información: DSS, EIS, y los sistemas de soporte para la decisión en grupo (GDSS, Group decisión support system).
- Sistemas de soporte orientados al conocimiento: DW, aplicaciones de exploración (Data Mining).

#### *1.1.2.2. Características, usos, beneficios y limitaciones.*

Los Sistemas de Soporte a la Decisión son aplicaciones informáticas que reúnen un grupo de características para ser clasificados como tal, según [24] las características generales de este tipo de sistemas, son:

- Decisiones en contexto semiestructurado o no estructurado.
- Brinda soporte a los encargados de las decisiones sin tratar de reemplazarlos.
- Respalda todas las etapas del proceso de toma de decisiones.
- Apunta hacia la eficacia más que hacia la eficiencia.
- Se encuentra bajo control del usuario encargado de la decisión.
- Usa modelos y datos básicos.
- Es interactivo y amigable al usuario.
- Está basado en un proceso iterativo.
- Proporciona soporte a todos los niveles administrativos.
- Respalda decisiones individuales y grupales.

Los beneficios que según [24] tiene el uso de un SSD son:

- Amplía la habilidad del decisor para procesar y asimilar información.
- Amplía la habilidad del gerente para enfrentar problemas complejos.
- Reduce el tiempo asociado a la toma de decisiones.
- Mejora la confiabilidad del proceso de decisión, así como el resultado.
- Incentiva la exploración y descubrimiento de parte del decisor.
- Crea una ventaja competitiva sobre la competencia.

Mientras que entre las limitaciones que según [24] tiene el uso de un SSD se encuentran:

- No puede incluir talentos distintivos humanos como la creatividad, imaginación, intuición.

- Su desempeño está limitado al sistema computacional en el que funcione.
- Los lenguajes e interfaces no son aún lo suficientemente sofisticados para procesar lenguaje natural.
- Por su especialización, un DSS no puede llegar a usarse de manera generalizada en múltiples contextos de toma de decisiones.

Una vez estudiada y analizada la teoría alrededor de los SSD, es importante la revisión de la metodología y cómo se realiza la ubicación de los estudiantes en un rol, a partir de entender el proceso de selección de personal. Esto con el objetivo de identificar las clasificaciones, el tipo de ayuda y la implementación de las características de los sistemas de soporte a la decisión en el problema que se desea solucionar.

### 1.2. La ubicación del estudiante en un rol del proceso de desarrollo de software en la UCI

El proceso de ubicación de los estudiantes en los roles del proceso de desarrollo de software en la UCI, sustenta la definición de sus actividades en las propuestas para un proceso de selección de personal y en la tendencia del empleo de técnicas de inteligencia artificial y modelación matemática para brindar soluciones a este problema. [25]

En un ambiente de formación como el que caracteriza la universidad, no es posible que se pretenda seleccionar candidatos más preparados o con un nivel elevado de cubrimiento de las competencias definidas para ellos, es por lo que no se pueden aplicar los métodos que se aplican en la industria de una manera exacta. El objetivo en la industria es siempre quedarse con el candidato más apto, mientras que en la UCI es necesario ubicar a todos los estudiantes en un rol de un proyecto.

Se pretende definir el proceso de selección para el caso de la ubicación de los estudiantes que laboran en proyectos de software, como parte de su formación profesional y fuerza laboral importante de estos proyectos. Significativo destacar que la asignación de los estudiantes a roles es parte del proceso de formación, por lo que este proceso debe partir de tener como evidencias los resultados del tránsito del estudiante por su CB y debe brindar una caracterización del estudiante para su desarrollo en el CP, de manera que se pueda plantear un plan de formación más adecuado. Todos los estudiantes deben ser ubicados en un rol de un proyecto, por lo que no se pueden aplicar técnicas que descarten candidatos.[26]

Se estudian la metodología y modelo desarrollados en la UCI para la ubicación del estudiante en un rol del proceso de desarrollo de software, la cual ha tenido en cuenta las características y tendencias mencionadas.

#### 1.2.1. Modelo para la Ubicación de un estudiante en un rol.

El modelo propuesto, permite recomendar el orden de asignación de los roles en función de la predicción del éxito en el desempeño que pueda tener el individuo.

Este modelo representa el razonamiento que los decisores siguen para la ubicación de los estudiantes en un rol del proceso de desarrollo de software, a partir de la representación de una caracterización del estudiante y de los roles en los que deben ser ubicados. Se representa la lógica del razonamiento en varios componentes.

El modelo tiene como función principal el análisis de la información del estudiante, su comportamiento en cada uno de los roles, brindando los criterios para tomar la decisión de la ubicación del estudiante, con la caracterización en cada uno de ellos de manera que pueda elaborarse su plan de formación.

El modelo se define sobre la base del empleo de técnicas de inteligencia artificial para la representación y procesamiento de la información. las funciones definidas trabajan con elementos que se describen empleando la lógica borrosa.

El modelo se representa gráficamente de la siguiente forma:


**Figura 3: Modelo de ubicación del estudiante en los roles. [22]**

El modelo tiene *como entradas* fundamentales:

- Información del estudiante (incluye las notas de las asignaturas cursadas y los resultados de los diferentes diagnósticos realizados durante el desarrollo del CB).
- Información del rol (incluye los elementos del perfil del rol, dentro de los que se encuentran las competencias genéricas, los conocimientos y las competencias genéricas).

*Como salidas:*

- Índice de acercamiento a cada rol (IAR): definido como una distancia al nivel menor en el que el individuo se puede desempeñar, refleja además, cómo el individuo se puede desempeñar en el rol.
- Estado del rol: se considera como estado del rol al estado de cada competencia o elemento definido e incluido en su caracterización.

- Roles ordenados por cercanía: según el IAR calculado, se busca ordenar en función del rol al que el individuo se encuentre más cercano, representando esta cercanía el nivel de éxito que pueda tener.

La relación entre los diferentes componentes del modelo está dada sobre la base de las propias relaciones de información que se establecen entre las entradas y salidas que en cada uno de ellos se utiliza y procesa, para lograr realizar su función principal. La estructura del modelo está en correspondencia con los componentes, partiendo de las entradas, dadas por los componentes de caracterización de estudiantes y roles, el procesamiento de la información, así como la predicción, que daría como resultado la recomendación en función del orden de los roles para cada estudiante según el IAR, que es la salida del modelo. Se ha definido una metodología para la ubicación del estudiante en un rol del proceso de desarrollo de software que soporte el modelo.

### 1.2.2. Metodología para la ubicación de un estudiante en un rol.

El recurso humano objeto del proceso estudiado son los estudiantes que laboran en los proyectos de software como parte de su formación profesional y fuerza laboral importante de estos proyectos. El proceso de ubicación es parte del proceso de formación en el cual se evalúan las evidencias del estudiante para definir su ubicación. Además se brinda una caracterización en función de la cual se puede elaborar el plan de formación una vez ubicado en el rol.

El proceso de asignación tiene como base la información registrada sobre el estudiante durante el ciclo básico y sobre el diseño de los roles:

- Resultados académicos.
- Evaluaciones de competencias genéricas.
- Resultados de los diagnósticos aplicados.
- Perfiles de competencias de los roles.

La metodología propuesta tiene como premisa que el estudiante haya cumplido las asignaturas y evaluaciones del CB, que se tenga el registro de dichas evaluaciones. Es necesario además contar con las definiciones de los perfiles por competencia para cada uno de los roles donde se ubicarán los estudiantes. Esta metodología define un conjunto de fases en las cuales se ejecutan determinados pasos, que, centrados en el cálculo de un índice de acercamiento del estudiante al rol, brinda la información necesaria para la final ubicación del estudiante en un rol de un proyecto al comienzo de su ciclo profesional.[25]

Se proponen cuatro fases, preparación, ejecución, ubicación y cierre, representadas de manera cíclica en el esquema para dar intencionalidad a la retroalimentación necesaria de la ejecución del proceso e implementar mejoras que permitan la evolución hacía la obtención de mejores resultados.

El esquema simplificado de la metodología es el siguiente:


**Figura 4: Fases de la metodología para la ubicación de estudiantes en roles del Proceso de Desarrollo de Software**

La primera fase es la de **preparación**. En esta se deben revisar y ajustar todas las herramientas que se aplican durante la ejecución de la metodología, diagnósticos, método matemático y sus componentes.

La siguiente fase es la fase de **ejecución** en la cual se aplican diagnósticos a los estudiantes y se recopila la información para la aplicación del modelo. Es en esta etapa donde se evalúan las funciones definidas en el modelo para definir el acercamiento al rol del estudiante.

La fase de **ubicación** es la tercera y tiene como objetivo fundamental, a partir de la información obtenida en la fase anterior, caracterizar al estudiante, realizar entrevistas en las que se puedan determinar algunos elementos de particular interés para los proyectos y que no pueden ser identificados a partir de los diagnósticos, y finalmente la ubicación de un estudiante en un rol dentro del proyecto.

La última fase, el **cierre**, permite la evaluación de los pasos realizados para una retroalimentación de la metodología y el método aplicado para este proceso de ubicación.

A diferencia de los métodos y metodologías tradicionales para la selección de personal, que se realizan con el objetivo de contratar a la persona más capacitada y mejor formada para el puesto de trabajo, valorando las competencias que tiene, la metodología presentada está diseñada para personal en formación y se basa fundamentalmente en la determinación de conocimientos básicos y capacidades de la persona para su formación y desempeño en uno de los roles.[25]

Teniendo en cuenta los elementos expresados, dado que este proceso o parte de él, se desea informatizar en la presente investigación, es necesario para su entendimiento hacer un estudio sobre el proceso de selección de personal.

### 1.2.3 La selección de personal

La selección de personal es uno de los procesos que se plantea realizar dentro de la Gestión de Recursos Humanos [27-37], es donde se definen todos los procesos y actividades relacionados con la gestión del personal en una organización.

El elemento más importante para la definición de selección de personal es ser un proceso de comparación y decisión. De las definiciones estudiadas una de las más completas y que además permite establecer a partir de ella una modelación matemática al problema es la de [29], es por ello que se cita de manera textual a continuación:

> *"La mejor manera de formular el concepto de selección es representarlo como una comparación entre dos variables: por un lado, los requisitos del cargo que debe llenarse (requisitos que el cargo exige de su ocupante) y, por otro lado, el perfil de las características de los candidatos que se presentan para disputarlo. La primera variable es suministrada por la descripción y el análisis del cargo, mientras que la segunda se obtiene mediante la aplicación de las técnicas de selección. Sea **X** la primera variable y **Y** la segunda variable."*

En la metodología propuesta para la ubicación de los estudiantes en la UCI, teniendo en cuenta la distintiva característica de ser un personal que se encuentra en medio de un proceso de formación, se define el término como:

> *"La **selección de personal** es el proceso mediante el cual se recopila y analiza información con el objetivo de ubicar el candidato en el puesto más adecuado de acuerdo al estado de sus competencias, tomando como elementos para la decisión los criterios resultantes de la comparación del candidato con las competencias deseadas para el puesto o cargo a través de un proceso de valoración, en el que intervienen la aplicación de diferentes técnicas e instrumentos, así como la información suministrada por los propios candidatos. El proceso culmina con un diagnóstico actual y futuro del candidato seleccionado en cuanto a tiempo de aprendizaje y ejecución".* [25]

Esta definición se enmarca en el contexto de una gestión por competencias, siendo este el criterio mediante el cual se establece la comparación del candidato y lo que se quiere para el puesto.

Sobre la información a tener en cuenta para la obtención de los criterios y el desarrollo del proceso de selección, los autores [27, 29, 32-34, 36-38] coinciden en partir de la información respecto a la necesidad de las plazas o puestos a cubrir, en cantidad, definiendo además, para cada uno de ellos cuáles son las responsabilidades, tareas asociadas, conocimientos y otros elementos que definen los criterios a buscar en cada candidato para tomar la decisión final.[25]

Un elemento clave en la obtención de información sobre los candidatos es el empleo de técnicas de selección, algunas de las técnicas más empleadas son las entrevistas, pruebas de conocimiento, pruebas sicológicas y de habilidades, entre otras. [25]

Varios autores [27, 29, 32-34, 36-38] definen actividades para el proceso de selección de personal, algunos las especifican en mayor detalle y otros de manera general, siendo las más comunes en las definiciones las siguientes:

- Describir las características del puesto de trabajo que se somete al proceso de selección de personal [29, 33, 38, 39].
- Reclutamiento de los candidatos [29, 33, 38, 39].
- Elaborar solicitud y formularios iniciales del candidato [29, 33, 34, 36, 38, 39].

- Recopilar la información de candidato [29].
- Aplicar las técnicas de selección del personal [29, 33, 34, 36, 38, 39].
- Evaluar y clasificar a los candidatos [39]
- Selección del candidato, toma de decisión[29, 32-34, 36, 38]
- Informar a los candidatos el resultado del proceso [29]

Con la revisión de estas actividades es posible lograr un mejor entendimiento de las actividades planteadas en la metodología que se propone y un mejor análisis de cuáles de ellas pudieran ser informatizadas como parte del sistema a desarrollar. Siempre es importante para cualquier análisis tener en cuenta la característica distintiva del personal que se desea ubicar, no es un profesional sino un estudiante, un recurso humano en formación.

El modelo que se propone para la ubicación de los estudiantes en la universidad se basa en el empleo de técnicas de inteligencia artificial, fundamentalmente la lógica borrosa, empleando el modelado lingüístico para la representación de la información, el ordenamiento mediante métodos de distancia y operadores de agregación para la evaluación del estudiante. Es necesario entonces hacer una breve revisión de esta teoría para entender el modelo y poder realizar el análisis y diseño de la aplicación a implementar.

### 1.2.4 Elementos de Inteligencia Artificial.

La teoría de los conjuntos borrosos ha sido aplicada a procesos de gestión de recursos humanos por varios autores. Esta técnica facilita abordar los problemas de decisión en los que existe imprecisión y ausencia de criterios claramente definidos.[39]

Es la teoría de los conjuntos borrosos la técnica más empleada, debido a que en la mayoría de las investigaciones se coincide en el tratamiento de la incertidumbre asociada al problema de selección, por lo que se integra al resto de las técnicas empleadas.[25]

La lógica borrosa facilita el tratamiento de la incertidumbre en las situaciones que se utiliza, en el modelo a proponer, se emplean los elementos de esta teoría para la representación de la información, fundamentado en que las competencias son aspectos que su medición de manera exacta no es una representación de la realidad, sino que su evaluación natural está dada por niveles de desarrollo.

Teniendo en cuenta los principales elementos que se mencionan son empleados en el modelo y se hace una revisión de las definiciones que ayudan al entendimiento de los mismos. Los principales conceptos asociados a la lógica borrosa en el modelo son:

**Definición 1:** Un conjunto (clase) borroso A en X se caracteriza por una función de pertenencia $\mathrm{f_A(x)}$ que asocia a cada elemento de X un número real del intervalo $[0,1]$, donde el valor de $\mathrm{f_A(x)}$ en x representa el “grado de pertenencia de x al conjunto A”.[40]

**Definición 2**: Se entiende por variable lingüística una variable cuyos valores son palabras o sentencias en un lenguaje natural o artificial. En términos más específicos una variable lingüística es caracterizada por una quíntupla (N, T(N), U, G, M), donde N es el nombre de la variable, T(N) es el conjunto de términos, que son el conjunto de valores lingüísticos, U es el universo de discurso, G es la regla sintáctica para generar los términos en T(N), y M es la regla semántica para asociar a cada término lingüístico X su significado.[41-43]

Como enfoque para la semántica de las variables lingüísticas que se definen se utilizan funciones triangulares. Las etiquetas lingüísticas de las variables definidas tienen una representación con números borrosos triangulares.

**Definición 3:** Un número borroso es un subconjunto borroso que posee tres características: el referencial pertenece al campo de los números reales R, la función característica de pertenencia es normal y la función característica de pertenencia es convexa. [41-43]

**Definición 4:** Un número borroso triangular está determinado por su función de pertenencia, está formada por dos segmentos de líneas, uno del punto $(\mathrm{a}, 0)$ a $(\mathrm{m}, 1)$ y el segundo siguiendo desde $(\mathrm{m}, 1)$ a $(\mathrm{b}, 0)$. Su dominio es el intervalo [0,1]. Un número borroso triangular se representa por una tripleta ordenada $(\mathrm{a}, \mathrm{m}, \mathrm{b})$.[44]

Las operaciones con números borrosos triangulares se definen como:

**Definición 5:** Sean $\widetilde{\mathrm{A}}$ y $\widetilde{\mathrm{B}}$ dos números triangulares difusos representados por las tripletas $(\mathrm{a}_1, \mathrm{a}_2, \mathrm{a}_3)$ y $(\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3)$, para los que se definen las siguientes operaciones:

- Suma: $\widetilde{\mathrm{A}} \oplus \widetilde{\mathrm{B}} = \left(\mathrm{a}_1 + \mathrm{b}_1,\ \mathrm{a}_2 + \mathrm{b}_2, \mathrm{a}_3 + \mathrm{b}_3\right)$
- Resta: $\widetilde{\mathrm{A}} \ominus \widetilde{\mathrm{B}} = \widetilde{\mathrm{A}} + \left(-\widetilde{\mathrm{B}}\right) = \left(\mathrm{a}_1 - \mathrm{b}_3,\ \mathrm{a}_2 - \mathrm{b}_2, \mathrm{a}_3 - \mathrm{b}_1\right)$
- Multiplicación: $\widetilde{\mathrm{A}} \otimes \widetilde{\mathrm{B}} = \left(\mathrm{a}_1 \times \mathrm{b}_1,\ \mathrm{a}_2 \times \mathrm{b}_2, \mathrm{a}_3 \times \mathrm{b}_3\right)$
- División: $\widetilde{\mathrm{A}} \oslash \widetilde{\mathrm{B}} = \left(\mathrm{a}_1 \,/\, \mathrm{b}_1,\ \mathrm{a}_2 \,/\, \mathrm{b}_2, \mathrm{a}_3 \,/\, \mathrm{b}_3\right)$
- Multiplicación por un escalar: $\mathrm{k} \odot \widetilde{\mathrm{A}} = \left(\mathrm{k}\mathrm{a}_1, \mathrm{k}\,\mathrm{a}_2, \mathrm{k}\mathrm{a}_3\right)$

**Definición 6:** Sean $\widetilde{\mathrm{A}}$ y $\widetilde{\mathrm{B}}$ dos números triangulares difusos representados por las tripletas $(\mathrm{a}_1, \mathrm{a}_2, \mathrm{a}_3)$ y $(\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3)$, se define la distancia entre ellos como:

- $d\left(\tilde{a}, \tilde{b}\right) = |a_1 - b_1| + |a_2 - b_2| + |a_3 - b_3|$

Entender estas definiciones es importante para la modelación del sistema de soporte a la decisión en la ubicación del estudiante.

### 1.3 Herramientas y metodologías de desarrollo.

Es de suma importancia para el adecuado desarrollo de un software realizar, en su fase de concepción, la selección de las herramientas a utilizar, mitigando de esta forma cualquier problema

referente al tiempo o la calidad del software. En Cuba existe una política referente al uso de herramientas libres debido a la imposibilidad de adquirir software con licencias privadas, por lo que se han analizado una serie de herramientas que respondan a esta condición, además de lenguajes de programación y metodologías de desarrollo de software que respondan a las necesidades de la aplicación a implementar.

El estudio realizado está enfocado a la definición de lo necesario para cumplir con cada una de las partes del desarrollo de un sistema informático, es por ello que se presentan las metodologías de desarrollo y herramientas CASE, estas fundamentalmente para la descripción del análisis y modelación del sistema. Se describen además las herramientas de desarrollo, tanto para la implementación como el sistema gestor de base de datos.

### 1.3.1. Metodologías para el desarrollo

*"Desarrollar un buen software depende de un gran número de actividades y etapas"*, así opina [45] y continua su reflexión, *"donde el impacto de elegir la mejor metodología para un equipo, en un determinado proyecto, es trascendental para el éxito del producto".* Este autor aclara en su investigación que el uso de las metodologías para el desarrollo de un software es esencial en un proyecto, la cual debe encajar en el equipo, guiar y organizar actividades que conlleven a las metas trazadas en el grupo.

La investigación realizada por [46] se encuentra centrada en aquellas metodologías más tradicionales centralizadas especialmente en el control del proceso, estableciendo rigurosamente las actividades involucradas, los artefactos que se deben producir, y las herramientas y notaciones que se usarán. Este tipo de metodologías tienen a su favor que proveen al equipo de un alto grado de ordenamiento, de disciplina, pero al no adaptarse apropiadamente a los cambios no son métodos adecuados cuando se trabaja en un entorno donde los requisitos no pueden predecirse o pueden variar, estas metodologías se centran más en los procesos que en las personas, el cliente puede llegar a ser relegado. Estas propuestas han demostrado ser efectivas y necesarias en un gran número de proyectos, pero también han presentado problemas en otros muchos.

Por lo que se presentan las metodologías **ágiles** centradas, según [47], en el factor humano o en el producto del software. Las mismas están mostrando su efectividad fundamentalmente en proyectos que presentan requisitos cambiantes y cuando se exige reducir drásticamente los tiempos de desarrollo pero sin alterar la calidad. Las necesidades del cliente pueden variar desde el momento de contratación de un software al momento de su entrega; y es de mayor importancia satisfacer estas últimas que las primeras. Esto requiere procesos de software que en lugar de rechazar los cambios los incorpore. Las metodologías ágiles suministran una serie de normas y principios junto a técnicas que

intentan hacer la entrega del proyecto menos complicada y más satisfactoria tanto para los clientes como para los equipos de entrega.

Teniendo en cuenta las características que se han mencionado de cada uno de los tipos de metodologías, se ha definido que se utilizarán metodologías ágiles para la realización del producto, debido a que permite principalmente disminuir costos, brinda flexibilidad a proyectos de software donde la incertidumbre está presente, como es el caso que se describe en esta investigación, y donde los requisitos pueden variar con el tiempo. Además de presentar un equipo de desarrollo pequeño y para el desarrollo de la aplicación es muy conveniente hacer al cliente parte del mismo.

#### *1.3.1.1. Scrum, XP, SXP*

Dentro de las metodologías ágiles se estudiaron SCRUM (Gestión Ágil de Proyectos) y XP (Programación Extrema) debido a las características que presentan que se ajustan a las necesidades para el desarrollo del software a implementar, además del estudio de una personalización de estas dos metodologías que se ha utilizado en proyectos de la Universidad de las Ciencias Informáticas con el nombre de SXP.

El estudio realizado por [48] muestra a **XP** como una metodología de desarrollo ligera basada en una serie de prácticas que respaldan un aumento en la productividad al hacer software. Controla los problemas de riesgo en los proyectos, permitiendo la participación de pequeños grupos de programadores. Esta metodología demanda un variado equipo de desarrollo. Teniendo como meta entregar el software requerido a tiempo. Una diferencia que la caracteriza, según [45],sobre las metodologías tradicionales es que pone más énfasis en la adaptabilidad que en la previsibilidad.

El mismo autor, [45], propone algunas características de este método

- **Desarrollo iterativo e incremental:** pequeñas mejoras, unas tras otras.
- **Pruebas unitarias continuas**: frecuentemente repetidas y automatizadas, incluyendo regresión de pruebas. Se aconseja escribir el código de la prueba antes de la codificación.
- **Programación por parejas**: se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto. Se supone que la mayor calidad del código escrito de esta manera -el código es revisado y discutido mientras se escribe. Es más importante que la posible pérdida de productividad inmediata.
- **Frecuente interacción del equipo de programación con el cliente o usuario:** se recomienda que un representante del cliente trabaje junto al equipo de desarrollo.
- **Corrección** de todos los errores antes de añadir nueva funcionalidad. Hacer entregas frecuentes.

- **Refactorización del código:** es decir, reescribir ciertas partes del código para aumentar su legibilidad y mantenibilidad pero sin modificar su comportamiento. Las pruebas han de garantizar que en la refactorización no se ha introducido ningún fallo.
- **Propiedad del código compartida**: en vez de dividir la responsabilidad en el desarrollo de cada módulo en grupos de trabajo distintos, este método promueve el que todo el personal pueda corregir y extender cualquier parte del proyecto. Las frecuentes pruebas de regresión garantizan que los posibles errores serán detectados.
- **Simplicidad en el código**: es la mejor manera de que las cosas funcionen. Cuando todo funcione se podrá añadir funcionalidad si es necesario. La programación extrema apuesta que es más sencillo hacer algo simple y tener un poco de trabajo extra para cambiarlo si se requiere, que realizar algo complicado y quizás nunca utilizarlo.

Proponiendo, [49], algunas desventajas:

- Es recomendable emplearlo solo en proyectos a corto plazo.
- Altas comisiones en caso de fallar.
- Conseguir su implantación en un equipo podría resultar dificultoso.

La investigación de [45] expone además que **Scrum** es un proceso ágil y liviano utilizado para administrar y controlar el desarrollo de un software. Se emplea como marco para otras prácticas de ingeniería de software, como el Proceso Unificado de Desarrollo de Software (RUP, Rational Unified Process, según las siglas en inglés) o XP. Focalizada en priorizar el trabajo, maximizando la utilidad de lo que se construye. Los requisitos y las prioridades se examinan y ajustan durante el proyecto en intervalos cortos y regulares. Se busca entregar software que realmente resuelva las necesidades, aumentando la satisfacción del cliente.

El mismo autor [45] propone que en Scrum, el equipo se enfoca en construir software de calidad. Teniendo en cuenta que la gestión de un proyecto Scrum se concentra en definir qué características debe tener el producto a construir, transformando cualquier obstáculo que pudiera entorpecer la tarea del equipo de desarrollo. Este proceso busca que los equipos sean lo más prácticos y fructíferos posible.

El investigador [45] continua su estudio al proponer que Scrum posee un conjunto de reglas muy simples, basado en los principios de inspección continua, adaptación, auto-gestión e innovación. Teniendo como objetivo que el cliente se entusiasme y se comprometa con el proyecto, debido a que ve crecer el producto iteración tras iteración, localizando las herramientas para alinear el desarrollo con las metas de negocio de su empresa. Por otro lado, el equipo encuentra un ámbito propicio para desarrollar sus capacidades profesionales y resultando un incremento en la motivación de los integrantes del mismo.

[46] expone un grupo de características sobre esta metodología:

Scrum por su proceso iterativo incremental produce un grupo de funcionalidades en cada fin de iteración.

- Proceso ágil para el manejo y control del trabajo de desarrollo.
- Contenedor de prácticas de ingeniería existentes.
- Enfoque basado en equipos, incrementa el desarrollo cuando los requerimientos cambian rápidamente.
- Proceso que controla el caos entre los conflictos de interés y las necesidades.
- Camino para mejorar las comunicaciones y maximizar la cooperación.
- Camino para detectar la causa y solucionar cualquier problema en el desarrollo.
- Escalable desde proyectos simples a proyectos completos organizacionales, Scrum ha controlado y organizado el desarrollo de productos y proyectos con miles de desarrolladores e implementadores.
- Ruta para sentirse bien en el trabajo.

Seguido por las ventajas propuestas por [50]:

- Se trabaja en iteraciones cortas, de alto enfoque y total transparencia.
- Se acepta que el cambio es una constante universal y se adapta el desarrollo para integrar los cambios que son importantes.
- Se incentiva la creatividad de los desarrolladores haciendo que el equipo sea auto administrado.
- Se mantiene la efectividad del equipo habilitando y protegiendo un entorno libre de interrupciones e interferencias.
- Permite producir software de una forma consistente, sostenida y competitiva.
- Las reuniones se dedican a inconvenientes recientes, evitando el estancamiento.

**SXP**, es un híbrido cubano de metodologías ágiles (XP y Scrum), que ofrece una estrategia tecnológica, a partir de la introducción de procedimientos ágiles que permitan actualizar los procesos de software para el mejoramiento de la actividad productiva fomentando el desarrollo de la creatividad, aumentando el nivel de preocupación y responsabilidad de los miembros del equipo, y ayudando al líder del proyecto a tener un mejor control del mismo. Consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar el éxito del proyecto. Basada completamente en los valores y principios de las metodologías ágiles expuestos en el Manifiesto Ágil. Como método de estimación se utiliza la opinión de expertos y constan con métricas o indicadores para lograr una eficiente calidad.[51]

La creación de SXP con la unión de las metodologías XP y Scrum se centra principalmente en que con la utilización de SCRUM para la gestión, se logra una correcta planificación y organización; mientras

que XP respalda con sus prácticas todo el proceso de desarrollo y de esta forma se obtiene un proceso de software completo. Siendo XP y no otra la metodología candidata para guiar el proceso ingenieril, caracterizándose por presentar una documentación discreta y de mayor dinamismo. La idea de las duplas para el desarrollo resultó interesante para los investigadores, pues en pequeñas iteraciones dos desarrolladores lograrían hacer lo mismo que un equipo grande de desarrollo (analista, arquitecto, diseñador, desarrollador, probador). Scrum es entonces la metodología ideal para toda la gestión de proyectos, sirviendo de soporte para acelerar el dinamismo identificado en XP. La identificación de los pequeños sprint (iteraciones) y las reuniones diarias con el Scrum Máster (líder del proyecto) se acercaban a la disciplina que se quería alcanzar por los desarrolladores.[51]

Este híbrido consta de 4 fases principales: Planificación-Definición donde se establece la visión, se fijan las expectativas y se realiza el aseguramiento del financiamiento del proyecto; Desarrollo, es donde se realiza la implementación del sistema hasta que esté listo para ser entregado; Entrega, puesta en marcha; y por último Mantenimiento, donde se realiza el soporte para el cliente. De cada una de ellas se despliegan 7 flujos de trabajo: concepción inicial, captura de requisitos, diseño con metáforas, implementación, prueba, entrega de la documentación, soporte e investigación, utilizándose este último por el equipo de desarrollo cuando sea necesario, es decir, es un flujo que se puede mover y utilizarlo en cualquier parte del ciclo de vida del proyecto. [51]

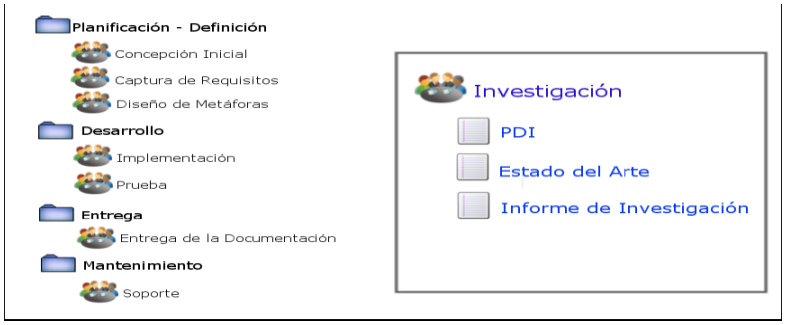



**Figura 5: Fases y flujos de trabajo de SXP. [46]**

Las autoras del presente trabajo elaboraron un esquema con estas fases, actividades y los artefactos que se generan en ellas, para lograr una mayor comprensión de la metodología a utilizar.

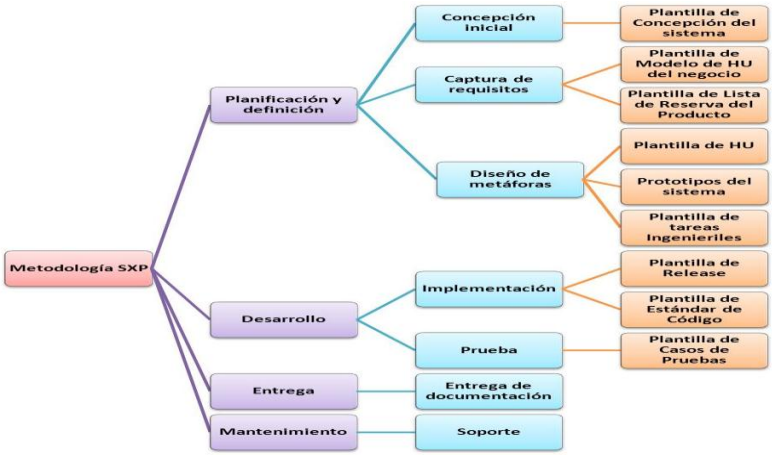



**Figura 6: Esquema de la metodología SXP**

En la presente investigación se decidió usar esta personalización de las metodologías XP y Scrum por las características que logran hacer, de un proceso de desarrollo, un proceso eficiente y eficaz. Ajustándose las mismas a las características propias del equipo de desarrollo entre las que se destacan principalmente: el equipo se encuentra formado por 2 estudiantes y un líder, el mismo necesita reunirse diariamente para discutir avances y posibles cambios, es de vital importancia para el equipo ganar en tiempo para no retrasar la entrega.

### 1.3.2 Herramientas Case

#### *1.3.2.1 Visual Paradigm sobre Rational Rose*

Al realizar un estudio al material [52] se concluye que **Rational Rose** es una herramienta de producción y comercialización, es un instrumento operativo conjunto, que utiliza el lenguaje para modelamiento unificado (UML, Unifed Modeling Languaje, por sus siglas en inglés) como medio para facilitar la captura de dominio de la semántica, la arquitectura y el diseño. Este software crea, modificar y manipular los componentes de un modelo. Uno de sus inconvenientes es que no es gratuito, se debe hacer un previo pago para adquirir el producto. Rose habilita asistentes para crear clases y provee plantillas de código que pueden aumentar significativamente la cantidad de código fuente generado. Adicionalmente, se pueden aplicar los patrones de diseño Banda de los Cuatro (GOF) para Java.

Se tomaron en cuenta además, para la elección de la herramienta de modelo a usar, las características que posee **Visual Paradigm** y que presenta [53], entre las que se encuentran que es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite diseñar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas, documentación, bases de datos, transforma diagramas de Entidad de Relación en tablas de una base de datos. También proporciona abundantes tutoriales de UML, demostraciones interactivas de modelado y proyectos del mismo. Se puede adquirir mediante licencia gratuita y comercial. Es fácil de instalar y actualizar y compatible entre ediciones. . Importación y exportación de ficheros XML. [53] [52] Existe amplia experiencia en la universidad en el uso de esta herramienta CASE en los proyectos productivos.

Al analizar las características de cada una de estas herramientas y teniendo en cuenta las normativas del país y la calidad que presenta Visual Paradigm, las autoras deciden utilizar la misma para la modelación.

### 1.3.3 Lenguaje de programación

#### *1.3.3.1 Java sobre PHP*

**Preprocesador de Hipertexto (PHP**, Hypertext Pre-processor según sus siglas en inglés**)** es un lenguaje de programación interpretado en el servidor, así lo denomina [54]. Publicado bajo licencia PHP, la cual se considera por la Fundación de Software Libre como una licencia de Software Libre. Fácil de aprender, multiplataforma, Soporta en cierta medida la programación orientada a objetos (POO) presentando algunas deficiencias en este sentido. Para su uso se necesita instalar un servidor web. Todo el trabajo es realizado por el servidor y no delega al cliente, por tanto puede ser más ineficiente a medida que las solicitudes aumenten de número. La legibilidad del código puede verse afectada al mezclar sentencias HTML y PHP. Dificulta la organización por capas de la aplicación. En cuanto a la seguridad, PHP es un poderoso lenguaje e intérprete, ya sea incluido como parte de un servidor web en forma de módulo o ejecutado como un binario CGI (Puerta Común de Interfaz) separado, es capaz de acceder a archivos, ejecutar comandos y abrir conexiones de red en el servidor, pero estas propiedades hacen que cualquier cosa que sea ejecutada en un servidor web sea insegura por naturaleza.

AL analizar la información expuesta por [54] es necesario señalar que **Java** es un lenguaje de programación orientado a objetos el mismo se presenta bajo licencia GNU GPL, por tanto se puede considerar el lenguaje Java como de Software Libre. Actualmente java se ha convertido en uno de los lenguajes más usados y más demandados por los desarrolladores. Es independiente de la plataforma de desarrollo. Existen dentro de su librería clases gráficas como awt y swing, las cuales permiten crear objetos gráficos comunes, altamente configurables y con una arquitectura independiente de la plataforma, por lo que es mucho más rápido que PHP. Este lenguaje permite aprovechar la flexibilidad de la POO en el diseño de sus aplicaciones. Con este lenguaje se puede acceder a bases de datos fácilmente con la conectividad de la Base de datos de Java (JDBC, Java Data Base Connectivity, según sus siglas en inglés), independientemente de la plataforma utilizada. El manejo de las bases de datos es uniforme, transparente y simple. En cuanto a la seguridad Java es mucho más confiable que PHP.

Teniendo en cuenta que ambos son lenguajes sobre los cuales se puede desarrollar basado en plataformas libres, para elegir el lenguaje a utilizar se tienen en cuenta las fortalezas y debilidades de cada uno. Se considera que para el desarrollo de la aplicación es más adecuado el empleo del lenguaje Java, es más potente y con mayores facilidades para la interacción con la base de datos, además de facilidades para la implementación de las capas en una arquitectura multicapas y para una programación orientada a objetos.

### 1.3.4 Marcos de trabajo

#### *1.3.4.1 ExtJS sobre JSF*

Teniendo en cuenta lo publicado en [54], **Java Server Faces** (**JSF**) un marco de trabajo para aplicaciones Java basado en el patrón Modelo Vista Controlador (MVC) simplificando el desarrollo de las interfaces de usuario. Suministra tags HTML para interfaces de una mayor complejidad. Esta herramienta permite la conversión y validación de los campos de los formularios, aunque estos validadores no son tan completos por no poder realizarse del lado del cliente. No soporta GET (peticiones en las que se utilizan pocos datos), lo cual da cierta seguridad, pero quita flexibilidad y en algunos casos "direccionalidad". Otro factor en su contra es que no existe mucha documentación sobre esta tecnología. [55] [54] [54] [54] [54] [54] [54] [54] [54]

Siguiendo la investigación realizada por [56] las autoras se percataron de que **ExtJS** es una biblioteca de JavaScript para aplicaciones Web interactivas, la cual hace uso de tecnologías como AJAX, DHTML y DOM. Actualmente se utiliza como extensión para las bibliotecas jQuery y Prototype. Desde su primera versión puede ejecutarse como una aplicación independiente. Consta de un conjunto de componentes extensibles y un API (Interfaz de Programación de Aplicaciones) fácil de usar. Entre los componentes con los que cuenta esta herramienta se encuentran: cuadros y áreas de texto, campos para fechas, campos numéricos, editor HTML, menús al estilo de Windows, paneles divisibles en secciones, entre otras.

ExtJS es un motor que permite crear Aplicaciones Enriquecidas en Internet (por sus siglas en inglés, RIA), mediante JavaScript. Si enmarcamos a ExtJS dentro del desarrollo RIA, éste sería el de la generación de imágenes de la aplicación que controla el cliente y que se encargaría de enviar y obtener información del servicio.

Continúa [56] en su estudio alegando que una de las grandes ventajas de utilizar ExtJS es que permite la creación de aplicaciones complejas utilizando componentes predefinidos que agilizan y facilitan el diseño..

Usar un motor de generación de imágenes como ExtJS permite tener además una serie de beneficios:

- Existe un balance entre Cliente – Servidor. La carga de procesamiento se distribuye, permitiendo que el servidor, al tener menor carga, pueda manejar más clientes al mismo tiempo.
- Comunicación asíncrona. En este tipo de aplicación el motor de renderización puede comunicarse con el servidor sin necesidad de estar sujeta a un clic o una acción del usuario, dándole la libertad de cargar información sin que el cliente se percate de ello.
- Eficiencia de la red. El tráfico de red puede disminuir al permitir que la aplicación elija que información desea transmitir al servidor y viceversa, sin embargo la aplicación que haga uso de la pre-carga de datos puede que revierta este beneficio por el incremento del tráfico.[56]

Teniendo en cuenta los elementos estudiados se decide utilizar ExtJS, fundamentalmente por la característica de permitir crear aplicaciones complejas utilizando componentes predefinidos, haciéndose más fácil su uso, además de ser independiente del cliente web que se emplee(Firefox, IExplorer, Safari, Opera, entre otros). Basando también su elección en el dominio que presenta el grupo de desarrollo sobre este marco de trabajo.

#### *1.3.4.2 Spring vs Seam*

Según [57], **SEAM** es un marco de trabajo de código abierto, el cual posee como fin unir diferentes tecnologías y estándares de Java, a la vez que añade algunas funcionalidades no contempladas por ellos.

Una de las grandes ventajas de esta tecnología es que se puede disponer de todas las funcionalidades que se buscan pero en uno solo, sin necesidad de integrar diferentes componentes y sin renunciar a la posibilidad de agregar alguno que falte. Es decir, si se necesita una funcionalidad estándar, SEAM permite iniciar el proyecto desde el primer día, sin necesidad de buscar diferentes marcos de trabajo o librerías, asegurando que estos componentes se encuentren ya probados y correctamente integrados entre sí.

[58] Propone algunas desventajas, una de ellas es que es un poco lento, siendo más difícil de usar que los componentes de fricción.

En el trabajo investigativo realizado por [59] se evidencia que **Spring** es un marco de trabajo de código abierto para el desarrollo de aplicaciones para la plataforma Java. Es el más popular y ambicioso de todos los marcos de trabajo de peso ligero. EL mismo interviene en todas las capas arquitectónicas de una aplicación Java. Teniendo en cuenta que su diseño brinda flexibilidad arquitectónica.

[60] comenta que los principales valores de Spring, se pueden resumir en: No es agresivo, provee un modelo consistente de programación, promueve la reusabilidad del código, facilita el diseño POO en aplicaciones Java, permite la extracción de valores de configuración desde el código java a archivos XML o archivos de propiedades. Diseñado para ser usado por las aplicaciones para que las pruebas sean lo más fácil posible. Spring hace de soluciones existentes un uso más fácil, dentro de una arquitectura consistente.

Teniendo en cuenta que ambos marcos de trabajo son de código abierto, siendo esta una característica muy importante para los desarrolladores debido a que es la utilización de herramientas libres una de las normativas de la UCI, se decide utilizar, para la capa del modelo de negocio, Spring, por las facilidades que brinda para el desarrollo de la solución.

#### *1.3.4.3 Hibernate sobre TopLink*

[61] Muestra a **Toplink** como un marco de trabajo de persistencia que permite el mapeo entre el modelo relacional y de objetos y entre XML y los objetos. Basado en especificaciones Java. Permite el

mapeo de objetos a múltiples tablas, el uso de herencia, transacciones anidadas, realizar el diseño en ambas direcciones, objetos / tablas y tablas / objetos, la adaptación de código SQL generado y el uso de procedimientos almacenados. [62] Propone como desventaja que es dependiente de APIs propietarias.

Según [63], **Hibernate** es un marco de trabajo objeto/relacional y un generador de sentencias sql, liberando al desarrollador del manejo manual de los datos que resultan de la ejecución de dichas sentencias. Permite diseñar objetos persistentes que pueden incluir polimorfismo, relaciones, colecciones, y un gran número de tipos de datos. Se integra a todo tipo de aplicación. Mantiene la portabilidad entre todas las bases de datos con un ligero incremento en el tiempo de ejecución, ofrece también un lenguaje de consulta de datos llamado HQL (*Hibernate Query Language, por sus siglas en inglés*), al mismo tiempo que un API para construir las consultas programáticamente (conocida como *"Criteria"*).

Según el equipo de desarrollo el marco de trabajo a utilizar para la capa de acceso a la base de datos es Hibernate, basándose en cómo genera esta herramienta las sentencias SQL y la posibilidad de liberar al desarrollador del manejo manual de los datos que resultan de dicha generación.

### 1.3.5 Gestores de Base de Datos

Al estudiar bibliografía referente a **MySQL**, [63], a las autoras les llamó la atención que el mismo es un sistema de gestión de bases de datos relacional, multihilo y multiusuario. Se puede encontrar bajo Licencia Pública General (GPL), pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar una licencia específica que les permita este uso. MySQL es patrocinado por una empresa privada, la cual posee el derecho de autor de la mayor parte del código. Su principal objetivo de diseño fue la velocidad. No suele perder información ni corromper los datos. No soporta transacciones, retrocesos ni subconsultas. Al no manejar la integridad referencial, este gestor se convierte en una solución pobre para muchos campos de aplicación. No es viable para su uso con grandes bases de datos, a las que se acceda continuamente, ya que no implementa una buena escalabilidad. [64]

Por lo que se tomó la decisión de usar **PostgresSQL**, [65], el cual es un sistema de gestión de base de datos relacional orientada a objetos y libre, publicado bajo la licencia de Distribución de Software Berkeley (BSD, Berkeley Software Distribution, según sus siglas en inglés). PostgresSQL da la posibilidad de que mientras un proceso es escrito en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos. Esta estrategia es superior al uso de bloqueos por tabla o por filas común en otras bases. Implementa el uso de retrocesos, subconsultas y transacciones, haciendo su funcionamiento mucho más eficaz, y ofreciendo soluciones en campos en las que MySQL no podría. Posee la capacidad de comprobar la integridad referencial, así como también la de almacenar

procedimientos en la propia base de datos, equiparándolo con los gestores de bases de datos de alto nivel, como puede ser Oracle. Soporta transacciones, claves ajenas (con comprobaciones de integridad referencial). Tiene mejor soporte para disparadores y procedimientos en el servidor.

### 1.3.6 Entornos de Desarrollo Integrado (IDEs)

#### *1.3.6.1 Eclipse sobre NetBeans*

Referente a lo expresado por [66], se puede señalar que **NetBeans** es un entorno de desarrollo integrado, libre y gratuito, sin restricciones de uso, desarrollado esencialmente para el lenguaje de programación Java.. Dicha plataforma es utilizada como estructura de integración para crear aplicaciones de escritorio grandes. Empresas especializadas en desarrollo de software, proponen extensiones adicionales que se integran con mayor facilidad a la plataforma, pudiendo desarrollar sus herramientas y soluciones. Este editor de interfaces gráficas produce código fuente solamente para el lenguaje Java.

[67] opina que no es posible editar el código fuente generado automáticamente, ya que este es bloqueado; pero a pesar de esto el usuario sí puede añadir su propio código fuente. Pero como desventaja se tiene que las modificaciones que se realizan de forma manual no logran verse en la vista de diseño, y de esta forma se hace necesario y obligatorio para comprobar su resultado compilar y ejecutar dicho código.

El equipo de desarrollo ha considerado utilizar **Eclipse** como IDE de desarrollo ya que según [67] este IDE es un entorno de desarrollo integrado de código abierto, multiplataforma para desarrollar "Aplicaciones de Cliente Enriquecido" proporcionando una interfaz gráfica, escrita con una sintaxis basada en XML, inverso a las aplicaciones "Cliente-liviano" que son basadas en navegadores.. La interfaz de usuario de Eclipse tiene una capa intermedia Interfaz Gráfica de Usuario, conocida también como GUI (del inglés Graphical User Interface) llamada JFace, la cual facilita la construcción de aplicaciones basadas en SWT (Standard Widget Toolkit).[62]Eclipse utiliza módulos (en inglés *plug-in*) para facilitar toda su funcionalidad al frente de la plataforma de cliente enriquecido, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no. Con respecto a las aplicaciones clientes, abastece al programador con marcos de trabajo muy ricos para las aplicaciones gráficas, definición y manipulación de modelos de software, aplicaciones web, entre otros. [68] [67]

### 1.3.7. Servidor de Aplicaciones

#### *1.3.7.1 Tomcat sobre GlassFish*

Teniendo en cuenta lo publicado por [69] se puede decir que **Glassfish** es un servidor de aplicaciones que implementa la plataforma **JavaEE5,** por lo que soporta las últimas versiones de tecnologías como: JSP, JSF, Servlets, EJBs, Java API para Servicios Web (JAX-WS), Arquitectura Java para Enlaces

XML (JAXB), Metadatos de Servicios Web para la Plataforma Java 1.0, y muchas otras tecnologías. Uno de los mayores inconvenientes que trae consigo este servidor es la compleja instalación que el mismo requiere. Es un servidor de software libre desarrollado por Sun Microsystems, compañía adquirida por Oracle Corporation. Es gratuito y de código libre.

Según [70] y [71] **Tomcat** es un servidor Web con soporte para servlets y JSPs. Trae consigo el compilador Jasper, que compila JSPs convirtiéndolas en servlets., el mismo puede funcionar como servidor Web por sí mismo. Al inicio de su desarrollo existía la idea de que su utilización era de manera autónoma, es decir sólo recomendable para entornos de desarrollo con mínimos requisitos de velocidad y gestión de transacciones, pero en la actualidad ya no es de esa forma, y es usado como servidor Web independiente en entornos con alto nivel de tráfico y alta disponibilidad. Es gratis, fácil de instalar, se ejecuta en máquinas más pequeñas y presenta compatibilidad con las API más recientes de Java. Permite la descarga, la instalación y la prueba en el iSeries en menos de una hora. Ocupa poco espacio, presentando su código binario un megabyte de tamaño, logrando con ello que se ejecute de manera rápida. Otra característica de Tomcat es que es muy fiable. Presenta gran éxito como producto de código libre.

El equipo de desarrollo se decidió por Tomcat además del estar escrito en Java, por las facilidades de uso, este satisface las necesidades para el desarrollo de la aplicación que se propone.

### 1.3.8. Controlador de versiones

#### *1.3.8.1 SVN sobre CVS (Concurrent Version System)*

A partir del estudio realizado a los materiales [72] y [73] se puede argumentar que CVS es una herramienta que marcha sobre un diseño de cliente y servidor. Consta de un cliente o estación de trabajo donde los desarrolladores realizan trasformaciones al código y hacen las pruebas que se necesiten para el cumplimiento delos requerimientos. El servidor contiene una versión consolidada del proyecto. Repetidamente los desarrolladores realizan actualizaciones de las versiones de trabajo desde el servidor y por otra parte envían sus propios cambios hacia el servidor.

Presenta grandes problemas, de los cuales se pueden mencionar los siguientes:

- Commits no atómicos: Si un commit falla a mitad de proceso, el servidor se quedará con un estado inconsistente (una parte de los archivos cambiados y otros no).
- Falta de cambios: La historia de los archivos se guarda individualmente (i.e., cada fichero guarda su propio historial de cambios). Es muy difícil ver los cambios globales a un grupo de archivos. (Este problema puede mitigarse en cierto modo si se usa un script de notificación de cambios.)

- Imposibilidad de cambiar nombres: No se puede cambiar el nombre de un archivo ni de un directorio sin perder información del histórico (ya que un cambio de nombre tiene que hacerse con una eliminación y una posterior adición; en el caso de los directorios, es aún peor).
- No controla directorios: CVS sólo guarda el historial de los archivos, no de los directorios.
- Ramas "caras" (traducción literal del inglés): El soporte para la gestión de ramas en CVS es muy caro en términos de rendimiento. Otros sistemas pueden manipular muy bien las ramas, y proporcionan varios algoritmos de "mezclado"

Por lo que el equipo tomó la decisión de hacer uso de **SVN**, pudiendo expresar del mismo según [74] primeramente que es software libre. Una característica importante de **Subversion** es que, a diferencia de CVS, los archivos versionados no tienen cada uno un número de revisión independiente. En cambio, todo el repositorio tiene un único número de versión que identifica un estado común de todos los archivos del repositorio en cierto punto del tiempo.

Ventajas frente al CVS

- Se sigue la historia de los archivos y directorios a través de copias y renombrados.
- Las modificaciones (incluyendo cambios a varios archivos) son atómicas.
- La creación de ramas y etiquetas es una operación más eficiente (tiene costo de complejidad constante y no lineal como en CVS).
- Se envían sólo las diferencias en ambas direcciones (en CVS siempre se envían al servidor archivos completos).
- Puede ser servido mediante Apache, sobre WebDAV/DeltaV. Esto permite que clientes WebDAV utilicen Subversion en forma transparente.
- Maneja eficientemente archivos binarios (a diferencia de CVS que los trata internamente como si fueran de texto).
- Permite selectivamente el bloqueo de archivos. Se usa en archivos binarios que, al no poder fusionarse fácilmente, conviene que no sean editados por más de una persona a la vez.

**Conclusiones Parciales**

- Los sistemas de soporte a la decisión permiten aumentar la capacidad de los decisores para tomar una decisión más acertada y con menor tiempo y costo.
- El sistema de soporte a la decisión que se debe implementar para apoyar el proceso de ubicación del estudiante, se puede clasificar:
    - De acuerdo al tipo de ayuda: Se le ofrece al usuario asesoramiento, expectativas, evaluaciones, hechos, análisis y diseños.
    - De acuerdo a su relación con el usuario: Pasivo
    - De acuerdo a la naturaleza de las decisiones: no estructurado

  - De acuerdo a evolución que han experimentado: sistemas de soporte orientados al análisis de la información

- El proceso de ubicación de los estudiantes es un proceso de selección en un ambiente de incertidumbre. Tiene características distintivas relacionadas con el hecho ser parte de un proceso de formación.
- El modelo para la ubicación de estudiantes en un rol del proceso de desarrollo puede ser implementado en un sistema que permita la ejecución de las funciones construidas y obtener los criterios para la decisión.
- De las fases definidas en la metodología para la ubicación se implementan como parte de un sistema de soporte a la decisión, actividades de las fases de ejecución y ubicación.
- Para el desarrollo del sistema se empleará la metodología SXP y la herramienta CASE Visual Paradigm para la descripción y modelación del sistema.
- Como sistema gestor de base de datos se utilizará el Postgres SQL.
- La implementación del sistema se hará empleando IDE Eclipse y en la implementación de cada una de las capas los marcos de trabajo: Hibernate para el acceso a datos, Spring para el negocio y ExtJs para la capa de presentación.
- Para el control de versiones se define SVN y como servidor de aplicaciones el apache Tomcat.

# CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

Para la construcción y desarrollo de un sistema informático, es de suma importancia la utilización de métodos y técnicas, que permitan la solución de problemas presentes a lo largo del ciclo de vida de un software. La correcta captura de requisitos y el modelamiento del sistema permiten que se mitiguen las fallas que puedan aparecer durante su desarrollo, además de establecer un entendimiento común entre lo que se desea y lo que se elabora. La implementación de la solución propuesta se caracteriza por basarse en los principios y reglas de la metodología SXP, utilizando las tecnologías y herramientas definidas.

La metodología empleada para la descripción del sistema se divide en fases y actividades dentro de estas, generándose un grupo de artefactos que permiten especificar los elementos fundamentales del sistema. En el capítulo 1 se muestra el esquema de las fases, actividades y artefactos de la metodología, de ellos en este capítulo se describen los artefactos correspondientes a la fase de Planificación y Definición.

## 2.1. Descripción del problema

Por las características de ser una universidad productiva, en la UCI se aplica un modelo que integre la docencia, la producción y la investigación. Para ello, el estudiante en los primeros 5 semestres de la carrera se formará académicamente de forma intensiva y el resto será fundamentalmente desde su desempeño en los distintos roles profesionales del proyecto donde se encuentre ubicado.

En entrevistas y encuestas realizadas se logra constatar que la manera en la que se realiza este proceso de ubicación no satisface totalmente las necesidades de la formación y la producción, por lo que ha sido diseñado un modelo y una metodología para la ubicación del estudiante, existiendo información y evidencias del desarrollo de dicho estudiante, pero haciéndose muy engorrosa su utilización por el volumen de la información a procesar y por la forma en que se presentan los resultados, a su vez, la cantidad de estudiantes a ubicar es grande, por lo que también complejiza el ordenamiento de los estudiantes por roles.

## 2.2. Solución Propuesta

Partiendo de que la información que se tiene del estudiante se encuentra digital en un grupo de sistemas desarrollados en la universidad, que existe un modelo para el procesamiento de dicha información y una metodología para el proceso, se hace necesario el desarrollo de una aplicación informática que implemente los algoritmos necesarios para la aplicación de la metodología y el modelo basado en las características de un sistema de soporte a la decisión. Esto permitirá el procesamiento

de los grandes volúmenes de información necesarios y generalizar el uso de la metodología y el modelo.

### 2.3 Concepción inicial del sistema

En la etapa de concepción del sistema se genera la plantilla de concepción del sistema, en ella se especifican los aspectos generales organizativos y de concepción del sistema, su objetivo, principales involucrados y otros aspectos que permiten la posterior organización del desarrollo del proyecto. Esta plantilla se muestra en el anexo 1.

### 2.4 Captura de requisitos

La ingeniería de requisitos es el conjunto de actividades implicadas en descubrir, documentar y mantener un conjunto de requisitos[51]. La captura de los mismos es un proceso en el cual los datos son extraídos de las personas pudiendo variar, dependiendo de la persona consultada.

Para la adquisición de estos requisitos se emplearon técnicas que permitieron hacer este proceso de forma más adecuada y segura. Entre las técnicas existentes para la captura de requerimientos se empleó la entrevista, específicamente la discusión, donde se sostuvo una discusión con el cliente sobre su problemática, para tratar de determinar en conjunto los requisitos del sistema, además de la arqueología de documentos donde se tratan de determinar posibles requerimientos sobre la base de inspeccionar la documentación utilizada por el cliente.

Como resultado de la aplicación de estas técnicas se obtuvo la plantilla de modelos de historias de usuarios del negocio y la lista de reserva del producto, estos elementos son descritos en los siguientes epígrafes.

Estos elementos que constituyen los requisitos de la aplicación a implementar fueron validados por el cliente, partiendo de que se describen estos artefactos de conjunto los desarrolladores y el cliente. Luego se sometieron a 2 rondas de revisiones lo cual permitió llegar a un consenso y claridad de las historias de usuario y sus descripciones.

#### 2.4.1 Historias de usuarios del negocio

En la plantilla del modelo de historias de usuarios del negocio se describen los actores y trabajadores del negocio, además se presenta un diagrama de historias de usuarios del negocio que permite ver la relación entre los usuarios y las actividades que se realizan, de ahí que se puedan obtener de ello los requisitos. Esta plantilla se describe en el anexo 2.

#### 2.4.2 Lista de reserva del producto (LRP)

En la lista de reserva, artefacto generado en la captura de requisitos, se describen los mismos como funcionalidades que el sistema debe cumplir en su desarrollo. Estos son descritos en la siguiente tabla:

| Prioridad | Ítem | Descripción | Estimación en sprint | Estimado por | Asignado a |
|---|---|---|---|---|---|

| | | | | | |
|---|---|---|---|---|---|
| **Requisitos Funcionales**<br>**Muy Alta** | | | | | |
| | 1 | Autenticación de usuarios | 1 | programador | Dianela Borrego León |
| | 2 | Cargar información del estudiante | 3 | programador | Leanet Arza Pérez |
| | 3 | Cargar información de la matriz de aporte | 3 | programador | Dianela Borrego León |
| | 4 | Cargar evidencias del estudiante | 3 | programador | Leanet Arza Pérez |
| | 5 | Predecir y Ordenar | 3 | programador | Dianela Borrego León |
| **Alta** | | | | | |
| | 6 | Mostrar reporte de ordenamiento | 3 | programador | Leanet Arza Pérez |
| | 7 | Mostrar caracterización del estudiante en un rol | 3 | programador | Dianela Borrego León |
| **Media** | | | | | |
| | 8 | Gestionar atributos del estudiante | | | |
| | 9 | Gestionar atributos del rol | | | |
| | 10 | Gestionar rol | | | |
| | 11 | Gestionar candidatos | | | |
| **Requisitos No Funcionales** | | | | | |
| **Usabilidad** | | | | | |
| | 1 | El software tendrá siempre la posibilidad de ayuda disponible para cualquier tipo de usuario, lo que le permitirá un avance considerable en la explotación de la aplicación en todas sus funcionalidades. | | Programador | Dianela Borrego León |
| | 2 | El tiempo de entrenamiento requerido para que usuarios normales y avanzados sean productivos operando el sistema es de 2 días. | | Programador | Leanet Arza Pérez |
| | 3 | Debe poseer una interfaz agradable para el cliente. | | Diseñador | Dianela Borrego León |
| **Fiabilidad** | | | | | |
| | 4 | El sistema estará disponible 24 horas del día, los siete días de la semana. | | Programador | Leanet Arza Pérez |
| | 5 | La herramienta de implementación a utilizar tiene soporte para recuperación ante fallos y errores. | | Programador | Dianela Borrego León |
| **Eficiencia** | | | | | |
| | 7 | El número de clientes o transacciones que el sistema puede alojar es de 2000. | | Programador | Dianela Borrego León |
| **Seguridad** | | | | | |
| | 8 | Protección contra acciones no autorizadas o que puedan afectar la integridad de los datos. | | Programador | Leanet Arza Pérez |
| | 9 | El sistema debe mantener en todo momento la seguridad de la información asegurando la autenticidad de la misma. | | Programador | Dianela Borrego León |
| | 10 | El sistema debe garantizar la confidencialidad, integridad y disponibilidad de la información que se procese en el sistema. | | Programador | Leanet Arza Pérez |
| | 11 | El control de acceso se establecerá por roles que se le asignarán a los usuarios que interactúen con el sistema. | | Programador | Dianela Borrego León |
| **Soporte** | | | | | |
| | 12 | Soporte para grandes volúmenes de datos y velocidad de procesamiento. | | Programador | Dianela Borrego León |
| | 13 | Tiempo de respuesta rápido en accesos concurrentes. | | Programador | Leanet Arza Pérez |
| **Restricciones de diseño** | | | | | |
| | 14 | El lenguaje de programación es Java 1.6. | | Programador | Dianela Borrego León |
| | 15 | El marco de trabajo de desarrollo es Hibernate 3.6.3, ExtJs3.1.0y Spring 3.1. | | Programador | Leanet Arza Pérez |
| | 16 | La herramienta IDE de desarrollo utilizada será Eclipse Indigo. | | Programador | Dianela Borrego León |
| | 17 | La herramienta case utilizada es Visual Paradigm 8.0 para modelado. | | Programador | Leanet Arza Pérez |
| | 18 | La herramienta gestor de base de datos es el PostgreSQL 9.1.2.1 | | Programador | Dianela Borrego León |

| | | | | |
|---|---|---|---|---|
| **Interfaz** | | | | |
| | 19 | El sistema tiene que ofrecer una interfaz amigable, fácil de operar. | Diseñador | Dianela Borrego León |
| | 20 | Diseño sencillo, con pocas entradas, permitiendo que no sea necesario mucho entrenamiento para que los usuarios puedan utilizar el sistema. | Diseñador | Dianela Borrego León |

**Tabla 1: Lista de Reserva del Producto**

## 2.5 Diseño de metáforas

A partir de la definición de las funcionalidades descritas en la LRP es posible establecer las historias de usuarios, prototipos del sistema y las tareas ingenieriles que permiten su desarrollo siendo estas las actividades que se realizan y se describen a continuación.

### 2.5.1 Historias de Usuario

Básicamente una historia es una lista priorizada de requisitos o funcionalidades, descritas usando la terminología del cliente. Estas historias de usuario se especifican en el anexo 3. En el documento se presentan dos de ellas para dar continuidad en la explicación de las tareas ingenieriles definidas.

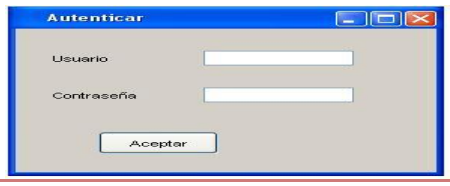
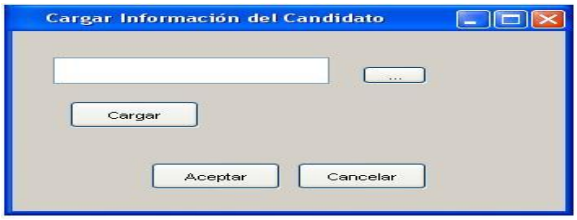
| Historia de Usuario | |
|---|---|
| **Número:** HU_1 | **Nombre Historia de Usuario:** Autenticación de usuarios |
| **Modificación de Historia de Usuario Número:** Ninguna | |
| **Usuario:** Dianela Borrego León | **Iteración Asignada:** Sprint 2 |
| **Prioridad en Negocio:** *Muy Alta* | **Puntos Estimados:**1 |
| **Riesgo en Desarrollo:** Alto | **Puntos Reales:** 1 |
| **Descripción:** *El usuario accede al sistema insertando un nombre y una contraseña, el sistema comprobará que estos datos son correctos, al serlo, el sistema le dará al usuario acceso a las funcionalidades a las cuales tiene permisos.* | |
| **Observaciones:** | |
| **Prototipo de interfaces:**   | |

**Tabla 2: HU Autenticación de usuarios**

| Historia de Usuario | |
|---|---|
| **Número:** HU_2 | **Nombre Historia de Usuario:** Cargar información del candidato |
| **Modificación de Historia de Usuario Número:** Ninguna | |
| **Usuario:** *Leanet Arza Pérez* | **Iteración Asignada:** Sprint 2 |
| **Prioridad en Negocio:** Muy Alta | **Puntos Estimados:**3 |
| **Riesgo en Desarrollo:** Alto | **Puntos Reales:** 3 |
| **Descripción:** *El usuario administrador del sistema especifica el fichero Excel en el que se encuentra la información del candidato, esta información se lee y se persiste en la base de datos para su posterior utilización.* | |
| **Observaciones:** | |

**Prototipo de interfaces:**



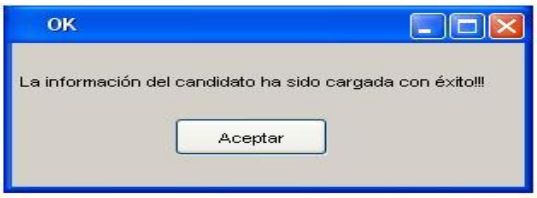




**Tabla 3: HU Cargar información del candidato**

| Historia de Usuario | |
|---|---|
| **Número:** HU_5 | **Nombre Historia de Usuario:** Ordenar roles y candidatos |
| **Modificación de Historia de Usuario Número:** Ninguna | |
| **Usuario:** Dianela Borrego León | **Iteración Asignada:** Sprint 2 |
| **Prioridad en Negocio:** Muy Alta | **Puntos Estimados:**3 |
| **Riesgo en Desarrollo:** Alto | **Puntos Reales:** 3 |

**Descripción**: *El usuario con los permisos requeridos tendrá, partiendo de criterios de búsquedas, la posibilidad de elegir los candidatos de su interés para una predicción y posterior ordenación de los roles según el índice de acercamiento al rol.*

*Para el grupo de candidatos seleccionados se debe calcular el índice de acercamiento al rol, para esto es necesario primero procesar la información que se tiene del candidato para obtener su evaluación en cada uno de los atributos del rol, de manera que se pueda comparar con los niveles deseados. Esta evaluación se hace mediante la función:*

$$\mathbf{g^{j}_{i,q}(\widetilde{E}_j) = \frac{1}{CAR(r_{i,q})} \sum_{p=1}^{n} \left(Vn(\widetilde{ac_{j,p}}) \times \widetilde{NA^{I}_{p,q}}\right)}$$

$\mathbf{g^{j}_{i,q}(\widetilde{E}_j)}$ *Recibe la información del estudiante j y evalúa la función para determinar la evaluación del estudiante j en el atributo q del rol i. El resultado de esta evaluación se persiste en la base de datos pues se necesita para la caracterización del estudiante.*

*Una vez obtenida la evaluación del candidato en el rol, el índice se determina a partir de los índices de acercamiento a cada uno de los casos ideales del rol, mediante la siguiente función:*

$$\mathbf{IAR_{j,i} = \frac{IAR^{k}_{j,i}}{CN}} \quad \textbf{donde i es el rol, j el candidato y k el caso dentro del rol}$$

*El Índice de Acercamiento al Rol del estudiante a cada caso se determina mediante la ecuación:*

$$\mathbf{IAR^{k}_{j,i} = \delta\left(\widetilde{E^{j}_{I}}, \widetilde{R^{k}_{I}}\right) = \frac{1}{n} \sum_{q=1}^{n} D\left(\widetilde{ar^{j}_{I,q}}, \widetilde{ar^{k}_{I,q}}\right)}$$

$$\boldsymbol{Donde\ E^{j}_{i}\ son\ los\ valores\ del\ estudiante\ \ j\ en\ el\ rol\ i,}$$
$$\boldsymbol{R^{k}_{i}\ son\ los\ valores\ del\ caso\ k\ del\ rol\ i,}$$
$$\boldsymbol{\widetilde{ar^{J}_{I,q}}\ es\ un\ n\acute{u}mero\ borroso\ triangular}$$

*El valor del* $\mathbf{IAR^{k}_{j,i}}$ *se persiste en la base de datos. Una vez calculado todos los IAR se ordenan de menor a mayor para el candidato y se almacena ese orden.*

**Observaciones**:

**Prototipo de interfaces**:


**Tabla 4: HU Ordenar roles y candidatos**

### 2.5.2 Tareas Ingenieriles

Partiendo de las historias de usuarios se establecen un grupo de tareas ingenieriles para cada una de ellas, sirviendo de guía para el posterior desarrollo de la solución propuesta.

Algunas de las tareas ingenieriles están relacionadas con todas las historias de usuario preparando el ambiente de implementación, estas son:

| Tarea de Ingeniería | |
|---|---|
| **Número Tarea:** T_1 | **Número Historia de Usuario:** todas |
| **Nombre Tarea:** Generación de la base de datos | |
| **Tipo de Tarea :** Desarrollo | **Puntos Estimados:** 1 |
| **Fecha Inicio:** 15-enero-2012 | **Fecha Fin:** 20-enero-2012 |
| **Programador Responsable:** Leanet Arza Pérez | |
| **Descripción:** *Se genera la base de datos a partir del diseño realizado, esta debe implementarse sobre el sistema gestor definido para el desarrollo de la aplicación.* | |

**Tabla 5: Tarea Ingenieril 1**

| Tarea de Ingeniería | |
|---|---|
| **Número Tarea:** T_2 | **Número Historia de Usuario:** todas |
| **Nombre Tarea:** Montaje del ambiente de desarrollo con la integración de los marcos de trabajo seleccionados para el desarrollo | |
| **Tipo de Tarea :** Desarrollo | **Puntos Estimados:** 1 |
| **Fecha Inicio:** 15-enero-2012 | **Fecha Fin:** 5-febrero-2012 |
| **Programador Responsable:** Dianela Borrego León | |
| **Descripción:** instalar y configurar los marcos de trabajo definidos para el desarrollo de manera que se pueda comenzar la implementación de la aplicación. | |

**Tabla 6: Tarea Ingenieril 2**

| Tarea de Ingeniería | |
|---|---|
| **Número Tarea:** T_3 | **Número Historia de Usuario:** todas |
| **Nombre Tarea:** Realizar el mapeo de la base de datos en el marco de trabajo Hibernate. | |
| **Tipo de Tarea :** Desarrollo | **Puntos Estimados:** 1 |
| **Fecha Inicio:** 5-febrero-2012 | **Fecha Fin:** 20-febrero-2012 |
| **Programador Responsable:** Dianela Borrego León | |
| **Descripción:** Realizar el mapeo de la base de datos en el framework de acceso a datos seleccionado. | |

**Tabla 7: Tarea Ingenieril 3**

Otras están asociadas a cada historia de usuario, en particular, para poder dar cumplimiento a la implementación de las funcionalidades definidas en ellas, por ejemplo las relacionadas con la HU_5 son:

| Tarea de Ingeniería | |
|---|---|
| **Número Tarea:** T_11 | **Número Historia de Usuario:** HU_5 |
| **Nombre Tarea:** Diseñar las interfaces necesarias para Ordenar roles y candidatos | |
| **Tipo de Tarea :** Desarrollo | **Puntos Estimados:** 1 |
| **Fecha Inicio:** 15-marzo-2012 | **Fecha** Fin: 20-marzo-2012 |
| **Programador Responsable:** Dianela Borrego León | |
| **Descripción:** Se diseñan las funcionalidades necesarias para la implementación de la HU_5, teniendo en cuenta los prototipos diseñados. | |

**Tabla 8: Tarea Ingenieril 11**

| Tarea de Ingeniería | |
|---|---|
| **Número Tarea:** T_12 | **Número Historia de Usuario:** HU_5 |
| **Nombre Tarea:** Implementar Ordenar roles y candidatos | |
| **Tipo de Tarea :** Desarrollo | **Puntos Estimados:** 1 |
| **Fecha Inicio:** 20-marzo-2012 | **Fecha Fin:** 10-abril-2012 |
| **Programador Responsable:** Dianela Borrego León | |
| **Descripción:** Se implementa la funcionalidad vinculada con la historia de usuario. | |

**Tabla 9: Tarea Ingenieril 12**

Las tareas ingenieriles definidas se encuentran en la plantilla del anexo 4.

### 2.6 Arquitectura seleccionada

La Arquitectura de Software es, a grandes rasgos, una vista del sistema que incluye los componentes principales del mismo, la conducta de esos componentes según se la percibe desde el resto del sistema y las formas en que los componentes interactúan y se coordinan para alcanzar la misión del sistema. La vista arquitectónica es una vista abstracta, aportando el más alto nivel de comprensión y la supresión o diferimiento del detalle inherente a la mayor parte de las abstracciones [75].

Para el desarrollo del sistema se ha definido emplear una arquitectura base, para orientar el diseño de las capas lógicas a través de una propuesta de diseño base; organizar la forma de codificar según las propuestas de convenciones o estándares de códigos y recursos; brindar una estructura física para soportar el código, creando así un esqueleto base; y proponer mecanismos de colaboración entre los componentes integrados en ella.

La arquitectura a la cual se hace referencia es la arquitectura en 3 niveles, en la cual existe un nivel intermediario. Esto significa que la arquitectura generalmente está compartida por:

1. Un cliente, es decir, el equipo que solicita los recursos, equipado con una interfaz de usuario (generalmente un navegador Web) para la presentación
2. El servidor de aplicaciones (también denominado software intermedio), cuya tarea es proporcionar los recursos solicitados, pero que requiere de otro servidor para hacerlo
3. El servidor de datos, que proporciona al servidor de aplicaciones los datos que requiere

**Patrón Modelo-Vista-Controlador (Model-View-Controller, según sus siglas en inglés (MVC))**: usado principalmente en aplicaciones que manejan gran cantidad de datos y transacciones complejas donde se requiere una mejor separación de conceptos para que el desarrollo esté estructurado de una

mejor manera, facilitando la programación en diferentes capas de manera paralela e independiente. MVC sugiere la separación del software en 3 estratos: Modelo, Vista y Controlador, que serán explicados brevemente:[76]

**Modelo:** Es la representación de la información que maneja la aplicación. El modelo en sí son los datos puros que puestos en contexto del sistema proveen de información al usuario a la aplicación misma.[76]

**Vista:** Es la representación del modelo en forma gráfica disponible para la interacción con el usuario. En el caso de una aplicación web, la "vista" es una página HTML con contenido dinámico sobre el cual el usuario puede realizar operaciones.[76]

**Controlador:** Es la capa encargada de manejar y responder las solicitudes del usuario, procesando la información necesaria y modificando el modelo en caso de ser necesario.[76]

El sistema que se implementa desarrolla su arquitectura empleando el patrón arquitectónico Modelo-Vista-Controlador, que se basa en una arquitectura de 3 niveles. En el diagrama de clases se describe como se materializa el uso de este patrón arquitectónico.

### 2.7 Patrones de diseño

Para la definición de las clases del sistema, el diseño del sistema a implementar, es importante la revisión de algunos patrones que permiten realizar un diseño adecuado y consistente. La asignación de responsabilidades es la habilidad más importante en el análisis y diseño orientado por objetos, para ello tiene suma importancia la utilización de los patrones GRASP.

En términos generales, un patrón es un conjunto de información que proporciona respuesta a un conjunto de problemas similares, es decir, un patrón es una solución a un problema en un contexto.[77]

#### 2.7.1 Patrones GRASP (Patrones de Software para la Asignación General de Responsabilidad)

*"Los patrones **GRASP** describen los principios fundamentales de diseño de objetos para la asignación de responsabilidades. Constituyen un apoyo para la enseñanza que ayuda a entender el diseño de objeto esencial y aplica el razonamiento para el diseño de una forma sistemática, racional y explicable."* Estos patrones se describen a continuación:*[60]*

***Bajo acoplamiento:*** El Bajo Acoplamiento es un patrón evaluativo que el diseñador aplica al juzgar sus decisiones de diseño. Estimula asignar una responsabilidad de modo que su colocación no incremente el acoplamiento tanto que produzca los resultados negativos propios de un alto acoplamiento.[60]

***Alta Cohesión:*** Asignar una responsabilidad de modo que la cohesión siga siendo alta. Representa una Clase con responsabilidades moderadas en un área funcional, colaborando con otras para

concretar tareas. Diseño más claro y comprensible. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme.[60]

***Experto:*** Es un principio básico que suele utilizarse en el diseño orientado a objetos. Con él no se pretende designar una idea oscura ni extraña; expresa simplemente la "intuición" de que los objetos hacen cosas relacionadas con la información que poseen. Ofrece una analogía con el mundo real. [60]

***Creador:*** El patrón Creador guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que se debe conectar con el objeto producido en cualquier evento, se debe buscar una clase de objeto que agregue, contenga y realice otras operaciones sobre este tipo de instancias.[60]

***Controlador:*** Es un evento generado por actores externos. Se asocian con operaciones del sistema, operaciones del sistema como respuestas a los eventos del sistema, tal como se relacionan los mensajes y los métodos[60].Este patrón se pone de manifiesto con la utilización de una clase que coordina o controla las actividades que son necesarias realizar con las demás clases, en este caso ApoloController es quien hace uso de este patrón. No realiza mucho trabajo por sí mismo.

Otros patrones utilizados:

**Patrón Fachada:** Es un patrón de diseño de tipo Estructural. Proporciona una interfaz unificada de alto nivel para un subsistema, que oculta las interfaces de bajo nivel de las clases que lo implementan. Con esto se consiguen dos objetivos fundamentales: hacer el subsistema más fácil de usar y desacoplar a los clientes de las clases del subsistema.[78]

**Patrón DAO:** El problema que viene a resolver este patrón es el de contar con diversas fuentes de datos (base de datos, archivos, servicios externos, etc). De tal forma que se encapsula la forma de acceder a la fuente de datos. Este patrón surge históricamente de la necesidad de gestionar una diversidad de fuentes de datos, aunque su uso se extiende al problema de encapsular no sólo la fuente de datos, sino además ocultar la forma de acceder a los datos. Se trata de que el software cliente se centre en los datos que necesita y se olvide de cómo se realiza el acceso a los datos o de cuál es la fuente de almacenamiento.[79]

En la descripción del diagrama de clases se especifica cómo se materializa el uso de estos patrones en el diseño realizado.

### 2.8 Diagrama de Clases

El propósito de este diagrama es el de representar los objetos fundamentales del sistema, es decir, los que percibe el usuario y con los que espera tratar para completar su tarea en vez de objetos del sistema o de un modelo de programación.[80]

El diagrama de clases del diseño representa los métodos y atributos de cada una de las clases del sistema, para mostrar de forma simple la colaboración y las tareas de cada una de ellas en relación al sistema que conforman.

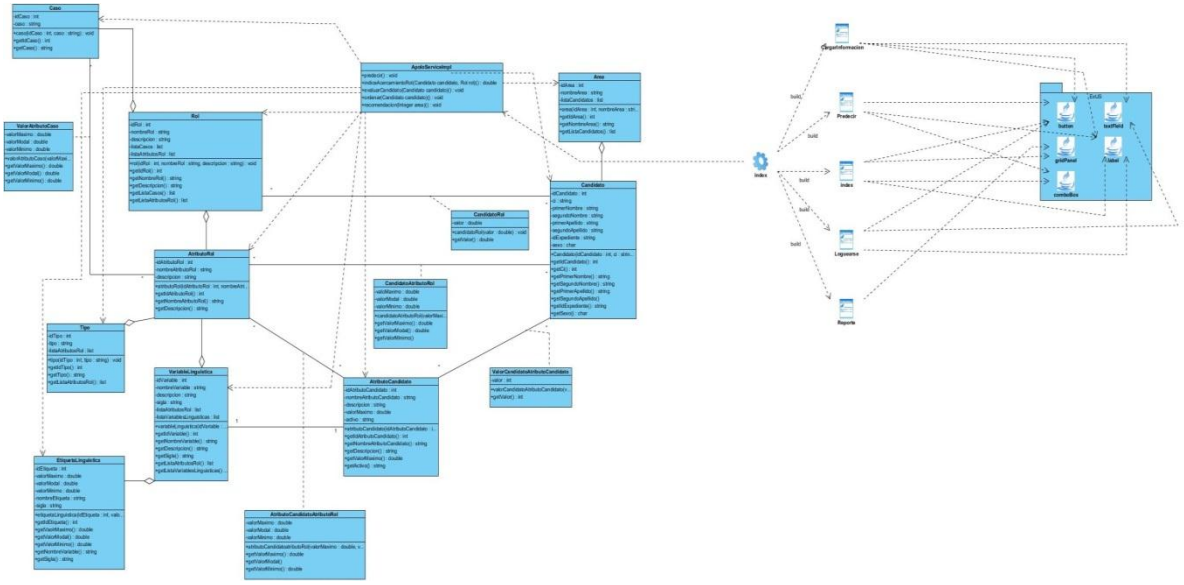


**Figura 7: Diagrama de clases**

En el diagrama de clases se pone de manifiesto el uso del patrón arquitectónico MVC, dada la definición de las clases interfaces que se corresponden con las vistas del patrón, ellas son CargarInformación, Predecir, MostrarCandidato, Loguearse y Reporte . El Controlador responde a la definición de la clase ApoloController que modelan el negocio implementado. El Modelo se corresponde con el diseño de las clases Candidato, Rol, AtributoRol, AtributoCandidato, Caso, VariableLinguistica, EtiquetaLinguistica, Area y Tipo, que responden a la representación de la información que se maneja en el sistema.

Los patrones GRASP utilizados se ponen de manifiesto en el diagrama de clases, asociado a las relaciones entre las clases y las responsabilidades asociadas a cada una de ellas para dar cumplimiento a los requisitos. En el caso del patrón creador es utilizado para dar la responsabilidad de la creación de los objetos a las clases que corresponde, el patrón experto se utilizó para asignar las responsabilidades relacionadas con las principales funcionalidades de la aplicación, por ejemplo para asignar la responsabilidad de la funcionalidad Recomendacion, que se dio a la clase ApoloServiceImpl que es la que cuenta con la información de los roles y los candidatos para poder responder esa petición. Relacionado con esto también el patrón alta cohesión, logrando que las clases tengan solo las responsabilidades que le corresponden de acuerdo a su objetivo e información que maneja.

Para analizar el uso del patrón bajo acoplamiento se vincula con otro patrón empleado, el patrón Fachada, el cual se utiliza para brindar una interfaz de comunicación entre varias clases, haciendo que

las relaciones entre las clases sean optimizadas en función de hacer que el sistema sea menos resistente a los cambios que se puedan producir. En este caso la clase fachada es ApoloFacade.

El patrón DAO se empleó para crear el conjunto de clases que permiten el manejo de los datos persistentes independizando esta gestión del sistema gestor de base de datos que se utilice. En la clase donde se aplica este patrón es la clase ApoloServiceImpl.

Cada una de las clases del diagrama se describe en el anexo 6.

### 2.9 Modelo de Datos

Es un conjunto de conceptos que nos permiten describir los datos, las relaciones entre ellos, la semántica y las restricciones de consistencia.[81]

Existen 3 tipos de modelos de datos:[81]

- **Modelos externos o lógicos basados en objetos:** permiten representar los datos que necesita cada usuario con las estructuras propias del lenguaje de programación que se vaya a usar.
- **Modelos globales o lógicos basados en registros:** ayuda a escribir los datos para el conjunto de usuarios.
- **Modelos físicos o de datos:** orientado a la máquina.

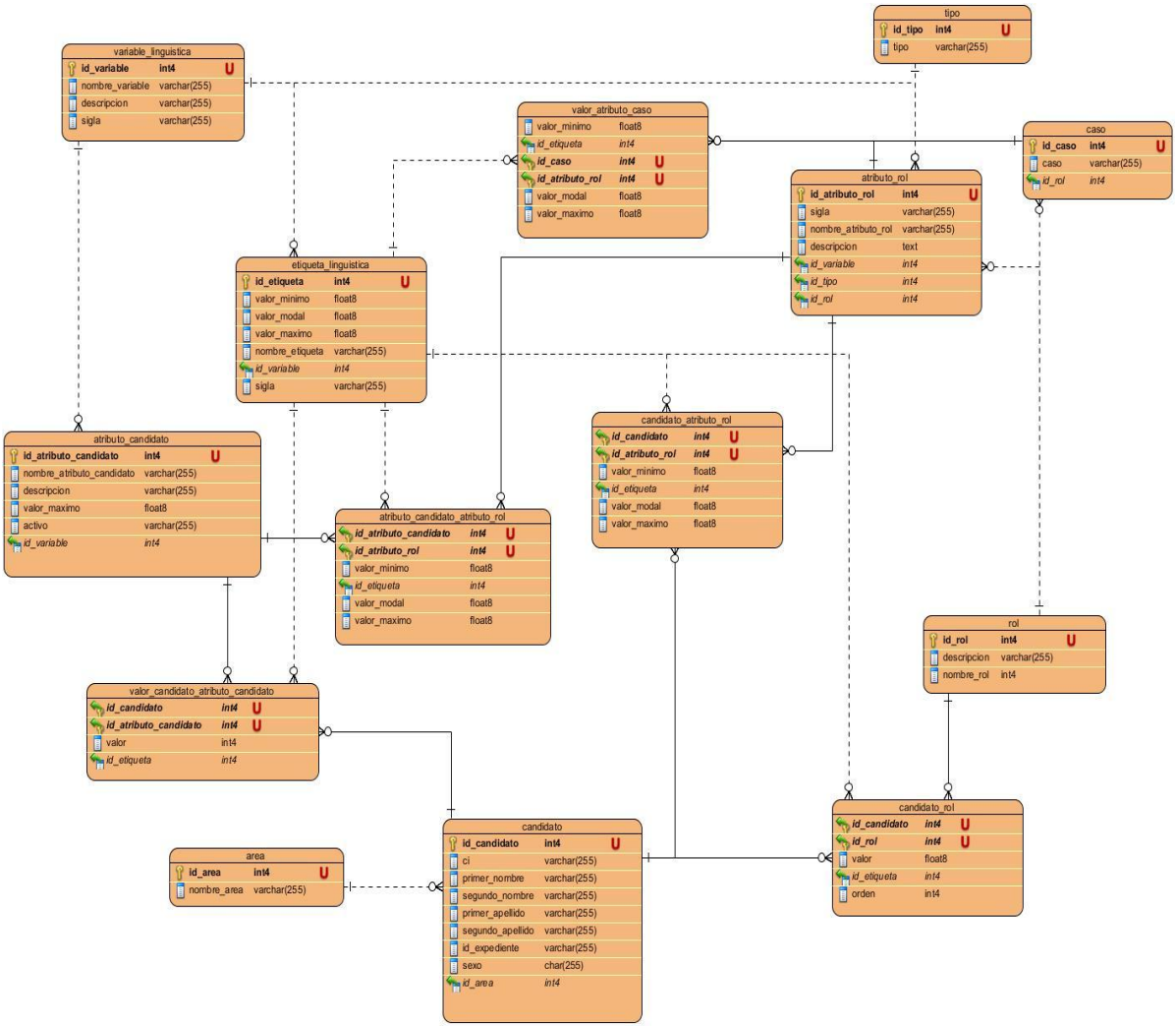


**Figura 8: Modelo de datos**

Se describe el modelo de datos en el anexo 5, cada una de las tablas y sus atributos, un ejemplo de la descripción de estos modelos es la tabla candidato.

| **Nombre Tabla:** atributo_candidato | | |
|---|---|---|
| **Descripción:** En esta tabla se almacenan los datos correspondientes a un atributo del candidato. | | |
| **Atributos** | **Tipo** | **Descripción** |
| id_atributo_candidato | Int | Es el identificador del atributo del candidato. Es la llave primaria de la tabla. |
| nombre_atributo_candidato | String | Es el nombre del atributo del candidato. |
| descripción | String | Describe a cada atributo del candidato. |
| valor_maximo | Int | Es el valor máximo que pueda tener el atributo del candidato. |
| activo | Booleano | Es para saber si el atributo del candidato está activo o no. |
| id_variable | Int | Es la llave foránea de la relación existente entre la tabla atributo_candidato y la tabla variable_ lingüística. |

## Conclusiones Parciales

- Al realizarse una correcta captura de requisitos, el equipo de desarrollo tuvo una mayor claridad de las necesidades del cliente permitiendo esto la comprensión del sistema y posterior representación de las funcionalidades requeridas.
- Se documentaron en la Lista de Reserva del Producto las principales funcionalidades a implementar.
- Se realizó la planificación de las tareas a cumplir por el equipo.
- Mediante las historias de usuarios, se sentaron las bases para las restantes fases del proceso.
- Con el diseño de un Modelo de Datos se representaron las entidades relevantes del sistema, y por medio del Diagrama de Clases se visualizaron las relaciones entre las clases que involucran dicho sistema, permitiendo una mejor comprensión del mismo.
- Estos resultados son utilizados como entrada en la siguiente etapa de trabajo.

# CAPÍTULO 3 IMPLEMENTACIÓN Y PRUEBA

A través de la realización de este capítulo se describe cómo fue implementada la aplicación en términos de componentes. Se detalla mediante el diagrama de despliegue como quedará distribuida la aplicación. Se presenta la validación de las pruebas de calidad desarrolladas. Al final de este capítulo se realizará un análisis de la solución obtenida.

## 3.1 Implementación

### 3.1.1. Diagrama de componentes

Un diagrama de componentes muestra las dependencias lógicas entre componentes software, sean éstos componentes fuentes, binarios o ejecutables, ilustran las piezas del software, controladores embebidos, etc**.** Los diagramas de Componentes prevalecen en el campo de la arquitectura de software pero pueden ser usados para modelar y documentar cualquier arquitectura del sistema, es decir para describir la vista de implementación estática de un sistema. Los diagramas de componentes se relacionan con los diagramas de clases, ya que un componente normalmente se corresponde con una o más clases, interfaces o colaboraciones pero un diagrama de Componentes tiene un nivel más alto de abstracción que un diagrama de clase, usualmente un componente se implementa por una o más clases (u objetos) en tiempo de ejecución. Estos son bloques de construcción, como eventualmente un componente puede comprender una gran porción de un sistema.[82]

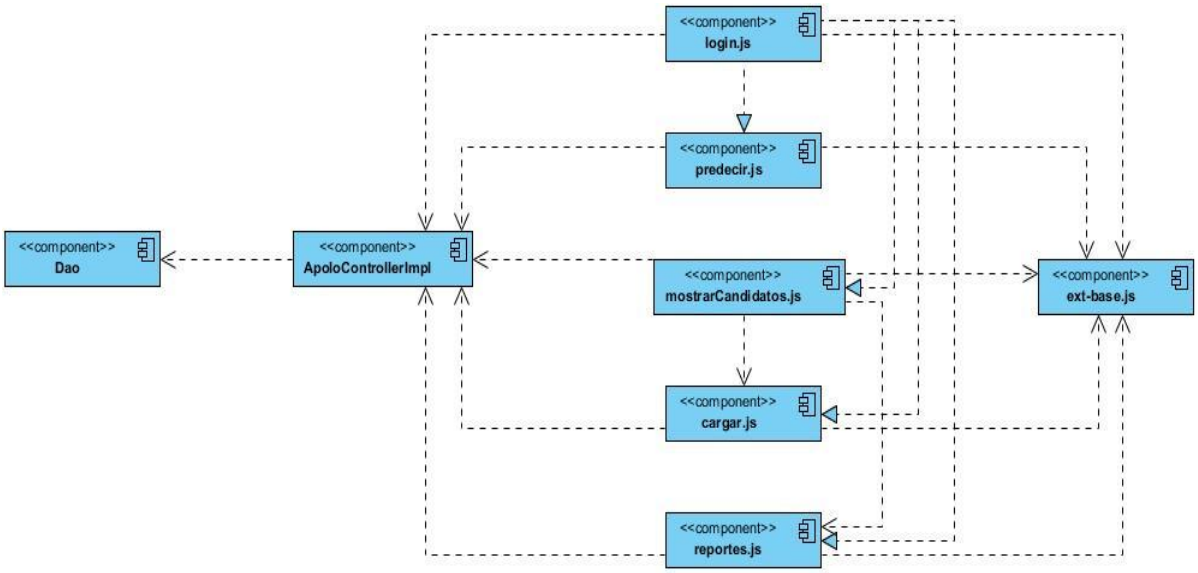



**Figura 9: Diagrama de componentes**

### 3.1.2. Diagrama de Despliegue

Para comprender como se ejecutará a nivel de hardware un sistema desarrollado y tener una visión clara de la estructura del sistema en ejecución y las relaciones entre los componentes que interactúan

en el mismo, se realiza el diagrama de despliegue. Dicho diagrama tiene como objetivo reflejar lo mencionado anteriormente, representando la disposición de las instancias de los componentes de ejecución, en instancias de nodos conectados por enlaces de comunicación.

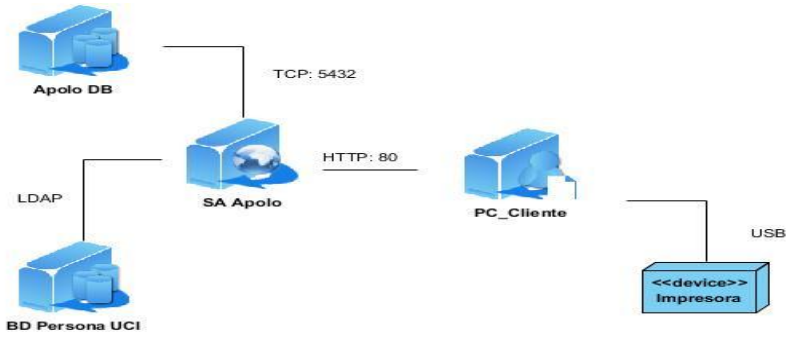



**Figura 10: Diagrama de despliegue**

Para el despliegue del sistema a implementar se tiene que la PC_Cliente se encuentra conectada por USB a una impresora donde se podrá obtener de forma física los reportes y por HTTP a través del puerto 80 al servidor de aplicación. Este último mediante el protocolo TCP y puerto 5432 se conectará a la base de datos del sistema y mediante LDAP a la base de datos de personas de la UCI.

### 3.2 Diseño de los casos de prueba

Las pruebas que se le realizan al sistema para la validación de sus funcionalidades son pruebas de caja negra, estas pruebas se llevan a cabo sobre la interfaz del software, obviando el comportamiento interno y la estructura del sistema.[83]

Los casos de prueba de caja negra pretenden demostrar que:[83]

- Las funciones del software son operativas.
- La entrada se acepta de forma correcta.
- Se produce una salida correcta.
- La integridad de la información externa se mantiene.

A continuación se derivan conjuntos de condiciones de entrada que utilizan todos los requisitos funcionales de una aplicación.

Las pruebas de caja negra pretenden encontrar errores como:[83]

- Funciones incorrectas o ausentes.
- Errores en la interfaz.
- Errores en la estructura de datos o en accesos a base de datos externas.
- Errores de rendimiento.
- Errores de inicialización y de terminación.

Algunos ejemplos de casos de prueba, para la aplicación, se muestran seguidamente:

| | |
|---|---|
| **Código Caso de Prueba:** A-1-1 | **Nombre Historia de Usuario:** Autenticación de usuarios |
| **Nombre de la persona que realiza la prueba:** Leanet Arza Pérez | |
| **Descripción de la Prueba:** Se ejecuta la prueba y se verifica que el usuario se autentique correctamente. | |
| **Condiciones de Ejecución:**<br>• Que se inserte correctamente el usuario y la contraseña.<br>• Que se encuentre activa la conexión con los servicios de LDAP. | |
| **Entrada / Pasos de ejecución:** Se inserta un usuario y una contraseña y se verifican que sean los correctos. | |
| **Resultado Esperado:** Que el usuario acceda al sistema y se activen las funcionalidades correspondientes. | |
| **Evaluación de la Prueba:** | |

**Tabla 10: Caso de Prueba de Autenticación de usuarios**

| Caso de Prueba de Aceptación | |
|---|---|
| **Código Caso de Prueba:** A-2-1 | **Nombre Historia de Usuario:** Cargar Información del candidato. |
| **Nombre de la persona que realiza la prueba:** Dianela Borrego León | |
| **Descripción de la Prueba:** Se ejecuta la prueba y se verifica que el administrador del sistema cargue correctamente la información de los candidatos. | |
| **Condiciones de Ejecución:**<br>• Se comprueba que el archivo a cargar esté en el formato establecido. | |
| **Entrada / Pasos de ejecución:** Al elegir la opción de examinar el sistema debe dar la posibilidad de buscar el lugar donde se encuentra el archivo que se decida cargar. | |
| **Resultado Esperado:** Que el administrador del sistema cargue correctamente la información referente a los candidatos. | |
| **Evaluación de la Prueba:** | |

**Tabla 11: Caso de Prueba de Cargar Información del candidato**

| Caso de Prueba de Aceptación | |
|---|---|
| **Código Caso de Prueba:** A-5-1 | **Nombre Historia de Usuario:** Ordenar Roles y Candidatos |
| **Nombre de la persona que realiza la prueba:** Leanet Arza Pérez | |
| **Descripción de la Prueba:** Se ejecuta la prueba y se verifica que se ordene correctamente. | |
| **Condiciones de Ejecución:**<br>• Que se halla cargado previamente la información referente a los candidatos, la matriz de aporte y las evidencias de los candidatos.<br>• Que se encuentre definido el ideal en el cual se basa el ordenamiento. | |
| **Entrada / Pasos de ejecución:** Para la ejecución de esta historia se tiene como entrada toda la información cargada y el ideal, luego se pasa a ordenar los candidatos atendiendo cuán aptos son los mismos para cada uno de los roles. | |
| **Resultado Esperado:** Que se ordene según el acercamiento de cada candidato a cada uno de los roles. | |
| **Evaluación de la Prueba:** | |

**Tabla 12: Caso de Prueba de Ordenar Roles y Candidatos**

El resto de los casos de pruebas definidos para la aplicación se encuentran en la planilla del anexo 7.

### 3.3 Resultados de la aplicación de las métricas para la validación

Las métricas de software son una medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo dado. Permiten averiguar cuán bien están definidas las clases y el sistema, lo cual tiene un impacto directo en el mantenimiento del mismo, tanto por la comprensión de lo desarrollado como por la dificultad de modificarlo con éxito. Estas métricas tienen como propósito entender y mejorar la calidad del producto, evaluar la efectividad del proceso y mejorar la calidad del trabajo llevado a cabo al nivel del proyecto (PRESSMAN, R. Ingeniería del Software: Un Enfoque Práctico *Mc Graw*, 2005.).

Para la evaluación de la calidad del diseño propuesto se hará uso de las métricas Tamaño Operacional de Clase (TOC) (LORENZ, M. and J. KIDD Object-oriented software metrics *Journal of Systems and Software*, 1994) (PRESSMAN, R. Ingeniería del Software: Un Enfoque Práctico *Mc Graw*, 2005.) y

Relaciones entre Clases (RC) ) (LORENZ, M. and J. KIDD Object-oriented software metrics *Journal of Systems and Software*, 1994) ( PRESSMAN, R. Ingeniería del Software: Un Enfoque Práctico *Mc Graw*, 2005).

TOC y RC permiten medir los siguientes atributos de calidad:

Las métricas empleadas están diseñadas para evaluar los siguientes atributos de calidad:[84]

- **Responsabilidad:** Consiste en la responsabilidad asignada a una clase en un marco de modelado de un dominio o concepto, de la problemática propuesta.
- **Complejidad de implementación:** Consiste en el grado de dificultad que tiene implementado un diseño de clases determinado.
- **Reutilización:** Consiste en el grado de reutilización presente en una clase o estructura de clase, dentro de un diseño de software.
- **Acoplamiento:** Consiste en el grado de dependencia o interconexión de una clase o estructura de clase, con otras, está muy ligada a la característica de *Reutilización*.
- **Complejidad del mantenimiento:** Consiste en el grado de esfuerzo necesario a realizar para desarrollar un arreglo, una mejora o una rectificación de algún error de un diseño de software. Puede influir indirecta, pero fuertemente en los costos y la planificación del proyecto.
- **Cantidad de pruebas:** Consiste en el número o el grado de esfuerzo para realizar las pruebas de calidad (Unidad) del producto (componente, módulo, clase, conjunto de clases, etc.) diseñado.

### 3.3.1 Métrica TOC

El TOC está dado por el número de métodos asignados a una clase y evalúa los siguientes atributos de calidad[85]:

- **Responsabilidad:** Un aumento del TOC implica un aumento de la responsabilidad asignada a la clase.
- **Complejidad de implementación:** Un aumento del TOC implica un aumento de la complejidad de implementación de la clase.
- **Reutilización:** Un aumento del TOC implica una disminución del grado de reutilización de la clase.

| | Categoría | Criterio |
|---|---|---|
| **Responsabilidad** | Baja | < =Promedio. |
| | Media | Entre Promedio y 2* Promedio |
| | Alta | > 2* Promedio |
| **Complejidad de implementación** | Baja | < =Promedio |
| | Media | Entre Promedio y 2* Promedio |
| | Alta | > 2* Promedio |
| **Reutilización** | Baja | > 2*Promedio |
| | Media | Entre Promedio y 2* Promedio |
| | Alta | <= Promedio. |

**Tabla 13: Rango de valores para la evaluación técnica de los atributos de calidad relacionados con la métrica TOC[86]**

A continuación se muestran las clases del sistema evaluadas en los atributos de calidad Responsabilidad, Complejidad de implementación y Reutilización.

| Clase | Cantidad de Procedimientos | Responsabilidad | Complejidad | Reutilización |
|---|---|---|---|---|
| **GenericHibernateDao** | 7 | Media | Media | Media |
| **Dao** | 4 | Baja | Baja | Alta |
| **ApoloFacadeImpl** | 2 | Baja | Baja | Alta |
| **ApoloFacade** | 2 | Baja | Baja | Alta |
| **ApoloServiceImpl** | 12 | Alta | Alta | Baja |
| **ApoloService** | 11 | Media | Media | Media |
| **ApoloController** | 7 | Media | Media | Media |

**Tabla 14: Evaluación de las clases del sistema mediante la métrica TOC**

A continuación se muestra una gráfica con agrupaciones por intervalos de la cantidad de clases según la cantidad de procedimientos:

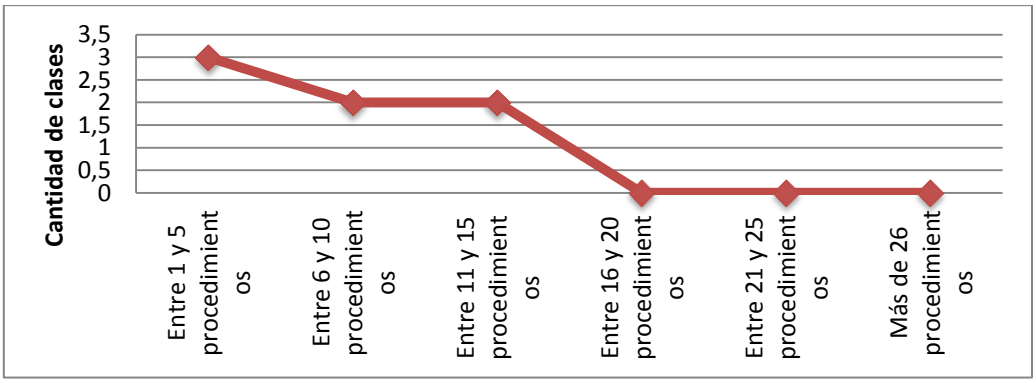



**Figura 11: Representación de la cantidad de clases agrupadas en intervalos según la cantidad de procedimientos.**

Las gráficas que corresponden a los resultados obtenidos se presentan en las siguientes figuras.

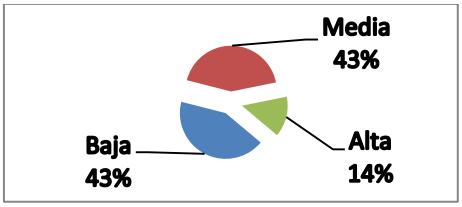



**Figura 12: Representación en por ciento (%) de los resultados obtenidos en el atributo Responsabilidad.**

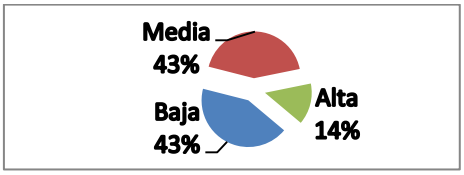



**Figura 13: Representación en por ciento (%) de los resultados obtenidos en el atributo Complejidad de implementación.**

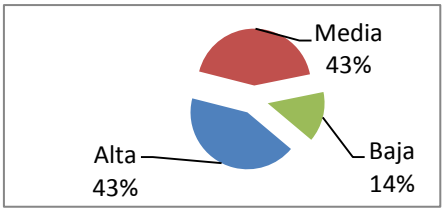


**Figura 14: Representación en por ciento (%) de los resultados obtenidos en el atributo Reutilización.**

### 3.3.2 Métrica RC

Está dado por el número de relaciones de uso de una clase con otra y evalúa los siguientes atributos de calidad.[85]

**Atributos de Calidad:[84]**

- **Acoplamiento:** Un aumento del RC implica un aumento del Acoplamiento de la clase.
- **Complejidad de mantenimiento:** Un aumento del RC implica un aumento de la complejidad del mantenimiento de la clase.
- **Cantidad de pruebas:** Un aumento del RC implica un aumento de la Cantidad de pruebas de unidad necesarias para probar una clase.
- **Reutilización:** Un aumento del RC implica una disminución en el grado de reutilización de la clase.

| | Categoría | Criterio |
|---|---|---|
| **Acoplamiento** | Ninguno | 0 |
| | Bajo | 1 |
| | Medio | 2 |
| | Alto | >2 |
| **Complejidad de Mantenimiento** | Baja | <= Promedio |
| | Media | Entre Promedio. y 2* Promedio. |
| | Alta | > 2* Promedio. |
| **Reutilización** | Baja | >2* Promedio. |
| | Media | Entre Promedio. y 2* Promedio. |
| | Alta | <= Promedio. |
| **Cantidad de Pruebas** | Baja | <= Promedio. |
| | Media | Entre Promedio. y 2* Promedio. |
| | Alta | > 2* Promedio. |

**Tabla 15: Rangos de valores para la evaluación técnica de los atributos de calidad relacionados con la métrica RC.[86]**

A continuación se muestran las clases del sistema evaluadas en los atributos de calidad Acoplamiento, Complejidad del mantenimiento, Cantidad de pruebas y Reutilización.

| Clase | Cantidad de Relaciones de Uso | Acoplamiento | Complejidad Mant. | Reutilización | Cantidad de Pruebas |
|---|---|---|---|---|---|
| **GenericHibernateDao** | 0 | Ninguno | Baja | Alta | Baja |
| **Dao** | 0 | Ninguno | Baja | Alta | Baja |
| **ApoloFacadeImpl** | 2 | Medio | Baja | Media | Media |
| **ApoloFacade** | 2 | Medio | Baja | Media | Media |
| **ApoloServiceImpl** | 2 | Medio | Baja | Media | Media |
| **ApoloService** | 2 | Medio | Baja | Media | Media |
| **ApoloController** | 2 | Medio | Baja | Media | Media |

**Tabla 16: Clases del Sistema evaluadas en los atributos de calidad según la métrica RC.**

A continuación se muestra una gráfica con agrupaciones por intervalos de la cantidad de clases según las dependencias entre ellas.

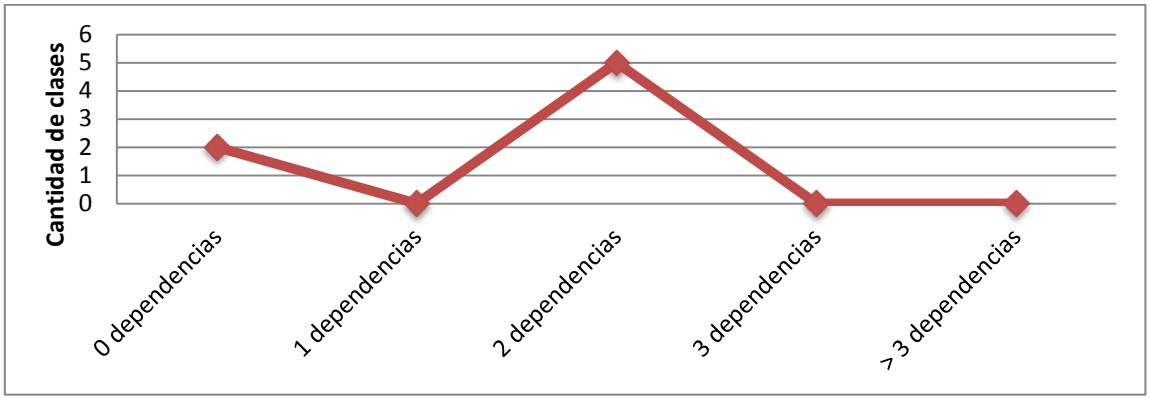



**Figura 15: Intervalos de las clases agrupadas según las dependencias entre ellas.**

El sistema cuenta de 7 clases las cuales poseen más de 3 dependencias entre ellas.

En la Figura 16 se muestran los valores en por ciento obtenidos al evaluar el diseño en el atributo Acoplamiento.

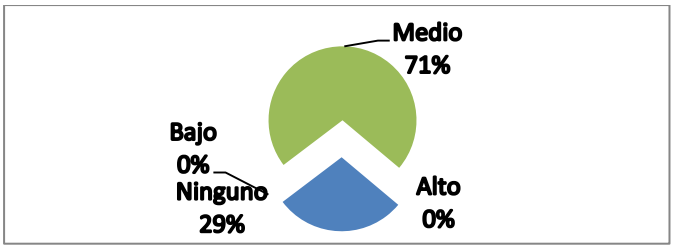



**Figura 16: Representación en porcientos (%) de los atributos obtenidos en el atributo Acoplamiento.**

Luego de la evaluación anterior se obtuvo que el 100% del acoplamiento posee un valor alto.

En la Figura 17 se muestran los resultados obtenidos luego de evaluar el atributo Complejidad de Mantenimiento.

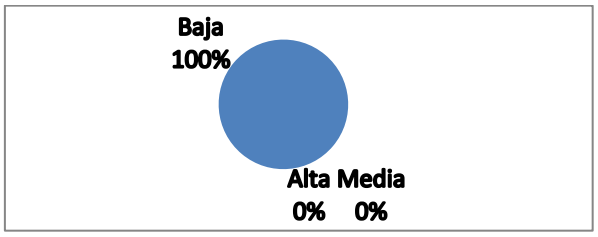



**Figura 17: Representación en porcientos (%) de los atributos obtenidos en el atributo Complejidad de Mantenimiento.**

En la Figura 18 se muestran los resultados obtenidos al evaluar el atributo Cantidad de Pruebas.

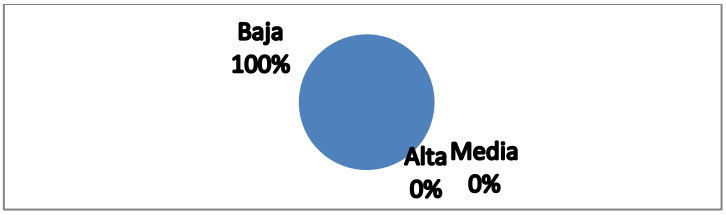



**Figura 18: Representación en porcientos (%) de los atributos obtenidos en el atributo Cantidad de Pruebas.**

En la Figura 19 se muestran los resultados obtenidos al evaluar el atributo Reutilización.

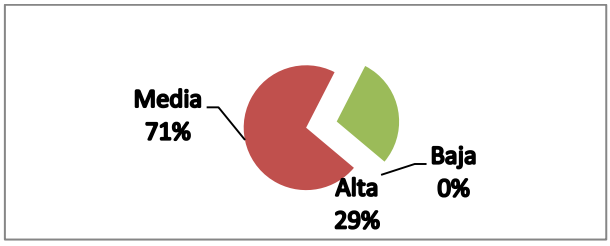



**Figura 19: Representación en porcientos (%) de los atributos obtenidos en el atributo Reutilización.**

## Conclusiones parciales

Como resultado de este capítulo:

- Se realizó el diagrama de componentes y de despliegue, realizados para mostrar una vista de cómo quedó la aplicación a nivel de componentes y su distribución.
- Se han descrito los diseños de casos de pruebas para realizar pruebas funcionales a la aplicación.
- Se aplicaron las métricas TOC y RC que arrojaron resultados positivos en la valoración de la validación de la aplicación propuesta.
- Con las pruebas y validaciones realizadas se puede concluir que la herramienta desarrollada cumple con las especificaciones y requisitos definidos por los clientes en la etapa de concepción del sistema.

# CONCLUSIONES

Durante el desarrollo de la presente investigación, descrita en el documento, se ha dado cumplimiento a los objetivos planteados, teniendo como principales conclusiones las siguientes:

Entre las principales conclusiones a las que se puede arribar con la investigación y trabajo desarrollado están:

- El modelo para la ubicación de estudiantes en un rol del proceso de desarrollo puede ser implementado en un sistema que permita la ejecución de las funciones construidas y obtener los criterios para la decisión.
- De las fases definidas en la metodología para la ubicación se implementan como parte de un sistema de soporte a la decisión, actividades de las fases de ejecución y ubicación.
- El sistema de soporte a la decisión implementado para apoyar el proceso de ubicación del estudiante, se puede clasificar:
  - De acuerdo al tipo de ayuda: Se le ofrece al usuario asesoramiento, expectativas, evaluaciones, hechos, análisis y diseños.
  - De acuerdo a su relación con el usuario: Pasivo.
  - De acuerdo a la naturaleza de las decisiones: no estructurado.
  - De acuerdo a  evolución que han experimentado: sistemas de soporte orientados al análisis de la información.
- Para el desarrollo del sistema se empleará la metodología SXP y la herramienta CASE Visual Paradigm para la descripción y modelación del sistema.
- La implementación del sistema se hará empleando IDE Eclipse y en la implementación de cada una de las capas los marcos de trabajo: Hibernate para el acceso a datos, Spring para el negocio y ExtJs para la capa de presentación. Como sistema gestor de base de datos se utilizará el Postgres SQL.
- Se ejecutaron las etapas y actividades correspondientes a la metodología definida, generándose los artefactos correspondientes.
- Los artefactos generados ayudan a una mejor comprensión y documentación del sistema implementado con vistas a futuros mantenimientos y escalabilidad de la aplicación.
- Se realizó la validación del sistema implementado mediante pruebas de funcionalidad y la aplicación de métricas. Esta validación arrojó como resultados que la aplicación desarrollada satisface los requerimientos especificados en las historias de usuario, así como el diseño de las clases e implementación de acuerdo a las métricas aplicadas.

# RECOMENDACIONES

Para una futura versión del sistema se recomienda:

- Darle fin a la implementación de las historias de usuarios relacionadas con la gestión de candidatos, roles y atributos de cada uno de ellos.
- Actualmente el sistema no permite realizar la gestión de ninguna de las variables e información que interviene en el modelo, solo realiza la predicción y recomendación. La información que se obtiene de otros sistemas de la universidad se realiza mediante la carga de ficheros Excel. Solo se implementa una función de semejanza para la predicción cuando el modelo plantea varias formas de hacerlo. Las deficiencias antes expuestas pudiesen servir como entrada para un Trabajo de Diploma posterior.

# REFERENCIAS


1. Verdecia, E., *Metodología para la formación formativa de roles desde la práctica profesional*. 2011, Universidad de las Ciencias Informáticas: La Habana.
2. André, A.M. *Un modelo para la asignación de recursos humanos a equipos de proyectos de software*. 2009. Ciudad de la Habana.
3. PMI, *A Guide to the Project Management Body of Knowledge* 2004, Pennsylvania: PMI Publications.
4. López Jimenez, T., *Selección de contenidos del Proyecto Estratégico de la UCI para el período 2008-2012*. 2008: Ciudad de La Habana.
5. *Métodos Teóricos*. Available from: http://www.buenastareas.com/ensayos/Metodos-Teoricos/136411.html.
6. Zayas, A.P.M., *EL ROMBO DE LAS INVESTIGACIONES DE LAS CIENCIAS SOCIALES*.
7. *Métodos empíricos*. Available from: http://es.scribd.com/doc/21229743/METODOS-EMPIRICOS.
8. *OBSERVACIÓN DESCRIPTIVA Y EXPERIMENTO*. Available from: HTTP://WWW2.UIAH.FI/PROJECTS/METODI/262.HTM.
9. Custodio, R.A. *MÉTODOS Y TÉCNICAS*. 2008; Available from: http://www.gestiopolis.com/economia/tecnicas-y-metodos-de-investigacion.htm.
10. SlideShare. *Tipos de Sistemas de Información*. 2009; Available from: http://www.slideshare.net/about.
11. Soto, C.E., *Decisiones en ambientes de incertidumbre*, Revista de Servicio Civil: San José, Costa Rica.
12. Gómez, A.; Available from: http://www.investigacion-operaciones.com/Curso_inv-Oper_carpeta/Clase21_II.pdf.
13. Simon, H., *"The New Science of Management Decision"*, in *Harper and Row*. 1960: New York.
14. Hurtado et al, T., Bruno, Gérard, *El proceso de analisis jerárquico (AHP) como herramienta para la toma de decisiones en la selección de proveedores*, Tesis Digitales UNMSM.
15. Finlay, P.N., *Introducing decision support systems*. 1994: Oxford, UK Cambridge.
16. Turban, E., *Decision support and expert systems: management support systems.* 1995.
17. Little, J.D.C., *"Models and Managers:The Concept of a Decision Calculus."*, in *Management Science*. 1970.
18. Keen, P.G.W., *Decision support systems: an organizational perspective*. 1978, ISBN 0-201-03667-3.
19. Moore, J.H., Chang, M.G., *Design of Decision Support Systems*, in *Data Base*. 1980.
20. Power, D.J., *What is a DSS? The On-Line Executive Journal for Data-Intensive Decision Support 1*. 1997.
21. Holsapple et al, C.W., Whinston, Andrew B., *Decision Support System: aknowledge- based approach*. 1996, Los Angeles.
22. Haettenschwiler, P., *Neues anwenderfreundliches Konzept der Entscheidungsunterstützung.*, in *Gutes Entscheiden in Wirtschaft*. 1999, 189-208.
23. Dominguez , A.J., Medina, Garrido Jose Aurelio, *El sistema de información como soporte a las decisiones*, in *La gestión de los sistemas de información en la empresa.* 2002, Ediciones Pirámide: Madrid.
24. Delfos, C.E., *SOFTWARE PARA LA IMPLEMENTACIÓN DE LOS SISTEMAS DE SOPORTE A LAS DECISIONES (SSD) A NIVEL INTERNACIONAL*. 2008.
25. Arza et al, P.L., Verdecia, Martínez Edistio Yoel , Lavandero, García José *PROCESO DE UBICACIÓN DE ESTUDIANTES EN UN ROL DEL PROCESO DE DESARROLLO DE SOFTWARE*, in *5ta Conferencia Internacional*. 2011: Holguín, Cuba.
26. Arza et al, P.L., Lavandero, García José , Verdecia, Edistio Yoel *EMPLEO DE MÉTODOS MÁTEMÁTICOS EN EL PROCESO DE UBICACIÓN DE ESTUDIANTES EN UN ROL DEL PROCESO DE DESARROLLO DE SOFTWARE EN LA UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS*, in *5ta Conferencia Científica Internacional*. 2011: Holguín, Cuba.
27. Arias, G., Fernando., *Administración de Recursos Humanos.* . 1979.
28. Chiavenato, I., *Administración de Recursos Humanos. 5ta Edición*. 1998, Colombia: Editorial Mc Graw Hill Interamericana S.A.
29. Chiavenato, I., *Gestión del Talento Humano*. 2002, Bogotá. Colombia: Editorial Mc Graw Hill Interamericana S.A.
30. Jofre, R., *PYMES y Gestión de los Recursos Humanos.* Ser Humano y Trabajo, 2010.
31. Martilla et. al., J.A., Jarnes. John C., *Importance - Performance Analysis.* Journal of Marketing, 1977: p. 77-79.

32. Pino et al, M.A.M., Sánchez, M.C., Pino, M.L.Q., *Recursos Humanos*, ed. Editex. 2008.
33. Puchol, L., *Dirección y gestión de recursos humanos*, D. Santos, Editor. 2007.
34. Rul et al, *Administración de Recursos Humanos. 3ra Edición*. 1996, Málaga: EDEA.
35. Wayne, R., Mondy, Noe, Robert M., *Administración de Recursos Humanos*. 1997, México: Prentice-Hal.
36. Wayne, R.M., Noe, R. M., *Administración de recursos humanos*. 9na Edición ed, ed. P. Educación. 2005, México.
37. Williamson et al., C., *Gestión de Recursos Humanos*. 2008.
38. Alles, M.A., *Selección por competencias*. 2006, Mexico: Ediciones Granica S.A. .
39. Gil et al, A.J., Kauffman, A., *Introducción de la teoría de los subconjuntos borrosos a la gestión de las empresas*. 3ra ed, ed. Milladoiro. 1993.
40. Zadeh, L.A., *"Fuzzy Sets."*. 1965.
41. Zadeh, L., *The concept of a linguistic variable and its application to approximate reasoning. Parte 1.* Information Sciences, 1975a. **8 no.3**: p. 199-249.
42. Zadeh, L., *The concept of a linguistic variable and its application to approximate reasoning. Parte 2.* Information Sciences, 1975b. **8 no.4**: p. 301-357.
43. Zadeh, L., *The concept of a linguistic variable and its application to approximate reasoning. Parte 3.* Information Sciences, 1975c. **9 no.1**: p. 43-85.
44. Wierman, M.J., *An Introduction to the Mathematics of Uncertainty*, ed. C. University. 2010.
45. Figueroa, R.G., Solís,Camilo J. , Cabrera,Armando A. , *Metodologías tradicionales vs metodologías ágiles*.
46. Infante, L. *Metodología Agil - Scrum*. 2009.
47. Canós, J.H., Letelier, Patricio, Panadés, M. Carmen. *Metodologías ágiles en el desarrollo de software*. Available from: http://www.willydev.net/descargas/prev/TodoAgil.Pdf.
48. Davis, A., *"A History of Agile Methods"*, in *JISBD 2002 Libertarios Fundamentalistas Tendencia global*. 2002.
49. *Rompiendo Paradigmas*. Available from: http://paradigmas14.blogspot.com/p/metodologia-utilizar.html.
50. *Ventajas de la metodología Scrum*. Available from: http://www.clubdesarrolladores.com/articulos/mostrar/63-metodologia-scrum/2.
51. Peñalver et al, G.M., A. García, S., *SXP, METODOLOGÍA ÁGIL PARA EL DESARROLLO DE SOFTWARE*, in *1er Congreso Iberoamericano de Ingeniería de proyectos*. 2010: Chile.
52. Zaho, J., Thomas, D., in *"Comparación de Herramientas de modelado UML: Enterprise Architect y Rational Rose."*. 2005.
53. *Visual Paradigm*. Available from: http://www.visual-paradigm.com.
54. Gómez, L.R., *PHP Vs Java*. 2010.
55. *Página oficial de JSF*.
56. Abreu, G.P.J. *Acerca de ExtJS*. 2011; Available from: http://www.ltu.jovenclub.cu/index.php?option=com_content&task=view&id=2001&Itemid=187.
57. *Introducción al SEAM framework*. Available from: http://www.google.com.cu/url?sa=t&rct=j&q=seam+framework+desventajas&source=web&cd=2&ved=0CCsQFjAB&url=http%3A%2F%2Fpfctikitaka.googlecode.com%2Ffiles%2F1%2520Introducci%25C3%25B3n.pdf&ei=60qhT4vSFMKKqwe6rq2gCQ&usg=AFQjCNFKnfwof7a8Txz9fJLKH_vhp0wmow.
58. *Java, write once, run away*. 2010; Available from: http://shervinasgari.blogspot.com/2010/10/why-you-should-not-use-seam-application.html
59. JOHNSON, R., *Professional Java Development with the Spring Framework*. 2005.
60. Areces, G.A., Diaz, Marquez Iskael, *"Diseño e implementación de las capas de negocio y acceso a datos de los módulos Planificación y Ejecución de Visitas Familiares"*. 2008.
61. *Oracle y su framework opensource de persistencia*. 2007; Available from: http://www.xperimentos.com/2007/03/29/oracle-y-su-framework-opensource-de-persistencia/.
62. Fernando, D.M. *Patrones de Diseño de Arquitecturas de Software Enterprise*. 2005; Available from: http://materias.fi.uba.ar/7500/montaldo-tesisdegradoingenieriainformatica.pdf.
63. *Relational Persistence for Java.*
64. *MySQL*.
65. Cameron, N., *PostgreSQL affiliates .ORG domain*. 2003.
66. Dominguez - Dorado, M., *NetBeans IDE 4.1. La alternativa a Eclipse.*, in *Todo Programación*. 2005, Editorial Iberprensa Madrid.
67. *Netbeans User FAQ*. 2007.
68. *The Eclpise Fundation open source community website*.

REFERENCIAS


69. *¿Qué es Glassfish?* 2009; Available from: http://bannysolano.wordpress.com/2009/08/23/%C2%BFque-es-glassfish/.
70. *Elección del servidor de aplicaciones web*. 2003; Available from: http://www.help400.es/asp/scripts/nwart.asp?Num=131&Pag=10&Tip=T.
71. *Servidor Web Tomcat*. 2011; Available from: (http://www.ecured.cu/index.php/Servidor_Tomcat.
72. Catrin, L.F.M. *Uso Práctico de CVS para control de versiones*. 2003; Available from: http://www.tuxpan.com/fcatrin/files/cvs.html.
73. *KDE migra de CVS a Subversion* 2005; Available from: http://barrapunto.com/articles/05/05/06/0652247.shtml.
74. *Subversion*. Available from: http://www.dosideas.com/wiki/Subversion.
75. Clements, P., *"A Survey of Architecture Description Languages"*, in *Proceedings of the International Workshop on Software Specification and Design*. 1996: Alemania.
76. *Arquitectura del Software*. Available from: http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/rivera_l_a/capitulo2.pdf.
77. *Patrones Grasp*. 2011; Available from: http://www.buenastareas.com/ensayos/Patrones-Grasp/1896730.html.
78. Nieto et al , D.J., Ramos, Fernández Pablo *El Patrón Fachada*: Universidad de Salamanca.
79. Lago, R. *Patrón "Data Access Object"*. 2007; Available from: http://www.proactiva-calidad.com/java/patrones/DAO.html.
80. *Diagrama de clases*. 2005; Available from: http://www-2.dc.uba.ar/materias/isoft1/is1-2005_2/apuntes/SlidesDC.pdf.
81. Fernández, R.J., *Modelo de Datos.*
82. Arizaca, R.E., *Artefacto: Diagrama de Componentes*. 2009: La Paz - Bolivia.
83. *Técnicas de Pruebas*. Available from: https://docs.google.com/viewer?a=v&q=cache:L0_moMWzYTYJ:indalog.ual.es/mtorres/LP/Prueba.pdf+pruebas+de+caja+negra&hl=es&gl=cu&pid=bl&srcid=ADGEEShyg3ux79zqPplD1O92UU_M3GdGAlm2GwqSjSr2t_-ZPK92JW5_hBCcumq1FcxUBeqrpgyuQ6QzRHEujVuzZ1T961tR-Omog9-fZtsh1JqIqFEOKiBLEYCPjgT-GyyMPmFmgdJ9&sig=AHIEtbQ_GHqnVZF5zNLQ7I7Epp26iQ3Bpg.
84. Driggs Vélez, J.C., *Diseño e Implementación de la nueva versión del Módulo Banco del Sistema Integral de Gestión Cedrux*. 2011.
85. EcuRed. *Métricas de diseño*. 2012; Available from: http://www.ecured.cu/index.php/M%C3%A9trica_de_dise%C3%B1o.
86. Romero, D.B., *Sistema para gestionar la actividad científica del Departamento de informática de la universidad de Guantánamo*, U.d.G.G. Departamento de informática Editor. 2010.

# ANEXOS

**Anexo 1 Plantilla de Concepción Inicial del Sistema**

1. **Polo productivo.**

Macro proyecto de investigación: Modelo de integración docencia-producción-investigación.

2. **Clasificación del proyecto.**

Desarrollo de aplicación

3. **Tipo de proyecto.**

Nacional

4. **Resumen:**

Este documento además de reflejar la visión general del producto a implementar, también recoge los diferentes roles que intervendrán en el desarrollo del software, así como las responsabilidades que tendrán en dicho proceso. Se documenta el tipo de proyecto al que pertenece así como la especificación del Polo Productivo y su clasificación. Se recoge además cuales herramientas serán utilizadas para el desarrollo de la aplicación, el alcance que va a tener, una descripción de los involucrados en el negocio, cuales son los motivos de la necesidad del desarrollo del software y la propuesta de solución.

5. **Surgimiento.**

El producto surge a raíz de entrevistas y encuestas realizadas donde se logra constatar que la manera en la que se realiza el proceso de ubicación de los estudiantes en un rol determinado no satisface totalmente las necesidades de la formación y la producción, existiendo información y evidencias del desarrollo del estudiante, pero se hace muy engorrosa su utilización, por el volumen de la información a procesar y por la forma en que se presentan los resultados. En la mayoría de los casos los líderes de proyecto guiándose solamente por los resultados académicos.

6. **¿Qué es?**

El sistema es un sistema de apoyo a la toma de decisiones para la ubicación de los estudiantes en un rol determinado, que a partir de la información que se posee del estudiante y los diferentes roles brinda un criterio que permite definir como esta cada estudiante con respecto a cada uno de los roles.

7. **Metodología a utilizar.**

**SXP** es la metodología a utilizar que consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar el éxito del proyecto. Basada completamente en los valores y principios de las metodologías ágiles expuestos en el Manifiesto Ágil.

La creación de SXP con la unión de las metodologías XP y Scrum se centra principalmente en que con la utilización de SCRUM para la gestión, se logra una correcta planificación y organización; mientras que XP respalda con sus prácticas todo el proceso de desarrollo y de esta forma se obtiene un proceso de software completo.

8. **Involucrados.**

Edistio Yoel Verdecia Martínez y Lizandra Arza Pérez expertos en la metodología y el modelo.

9. **Roles.**

| Rol | Responsabilidad | Nombre |
|---|---|---|
| Líder del Proyecto | Asegurar que el proyecto se está llevando a cabo de acuerdo con las prácticas y que todo funcione según lo planeado. Su principal trabajo es remover los impedimentos y definir, así como reducir los riesgos del producto. | Lizandra Arza Pérez |
| Gerente | Es el responsable de tomar las decisiones finales, acerca de estándares y convenciones a seguir durante el proyecto. Participa en la definición de objetivos y requerimientos, así como en la selección de los miembros del Equipo del Proyecto. Tiene la responsabilidad de controlar el progreso del software y trabaja junto con el Líder de Proyecto en la reducción de la Lista | Lizandra Arza Pérez |

| | | |
|---|---|---|
| | de reserva del producto, así como en la de riesgos. | |
| Cliente | El cliente contribuye a definir las historias de usuario y los casos de prueba de aceptación, para validar su implementación. Además, asigna la prioridad a las historias de usuario y decide cuáles se implementan en cada iteración centrándose en aportar mayor valor al negocio y participa en la concepción inicial del sistema. | Lizandra Arza Pérez<br>Edistio Yoel Verdecia Martínez |
| Programadores | El programador define las tareas de ingeniería y produce el código del sistema. Además selecciona el estándar de programación a utilizar, controlando incluso la gestión recambios. Debe existir una comunicación y coordinación adecuada entre los programadores y otros miembros del equipo. | Dianela Borrego Leon<br>Leanet Arza Pérez |
| Analista | Escribe la concepción del sistema y las historias de usuario. Crea el Modelo de historia de usuario del negocio y la LRP. Además, asigna la prioridad a las historias de usuario y decide cuáles se implementan en cada iteración centrándose en aportar mayor valor al negocio. | Leanet Arza Pérez |
| Diseñadores | Son los encargados del diseño del sistema, así como el de los prototipos de interfaces, máximos responsables de la realización del diseño de las metáforas y supervisan el proceso de construcción. | Dianela Borrego León |
| Encargado de Pruebas | Escribe los casos de prueba de aceptación. Ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para pruebas. | Leanet Arza Pérez<br>Dianela Borrego León |
| Arquitecto | Se vincula directamente con el analista y el diseñador debido a que su trabajo tiene que ver con la estructura y el diseño en grande del sistema. Ayuda en el diseño de las metáforas. | Leanet Arza Pérez |
| Consultor | Es un miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto, en el que puedan surgir problemas, además aportan ideas y experiencias para el beneficio del sistema en desarrollo. | Iskael Díaz Márquez |

**Tabla 17: Roles involucrados en el proceso de desarrollo del software.**

**10. Misión.**

Brindar los criterios para la toma de decisión para la ubicación de los estudiantes en un rol.

**11. Alcance**

El desarrollo de las funcionalidades necesarias para hacer el análisis de la información y dar los criterios para la toma de decisión para la ubicación de los estudiantes en un rol del proceso de desarrollo de software.

## Anexo 2 Modelo de historias de usuarios del negocio

### 1. Actores del negocio

| Actor | Descripción |
|---|---|
| Profesores vinculados a la producción | Estos profesores se benefician con el negocio al ser ellos los interesados de que los estudiantes sean ubicados correctamente en cada uno de los roles, permitiendo así que se realice de una forma ágil el trabajo en el proyecto. |

**Tabla 18: Actores del negocio.**

### 2. Trabajadores del negocio

| Trabajador | Descripción |
|---|---|
| Jefe de Práctica Profesional UCI | Será el encargado de cargar la información referente a los estudiantes y roles. |
| Jefe de Práctica Profesional Facultad | Serán los encargados de predecir el comportamiento de los estudiantes en cada uno de los roles |
| Profesores | Solo tendrán acceso a los reportes. |

**Tabla 19: Trabajadores del negocio.**

### 3. Diagrama de Historias de usuario del Negocio.

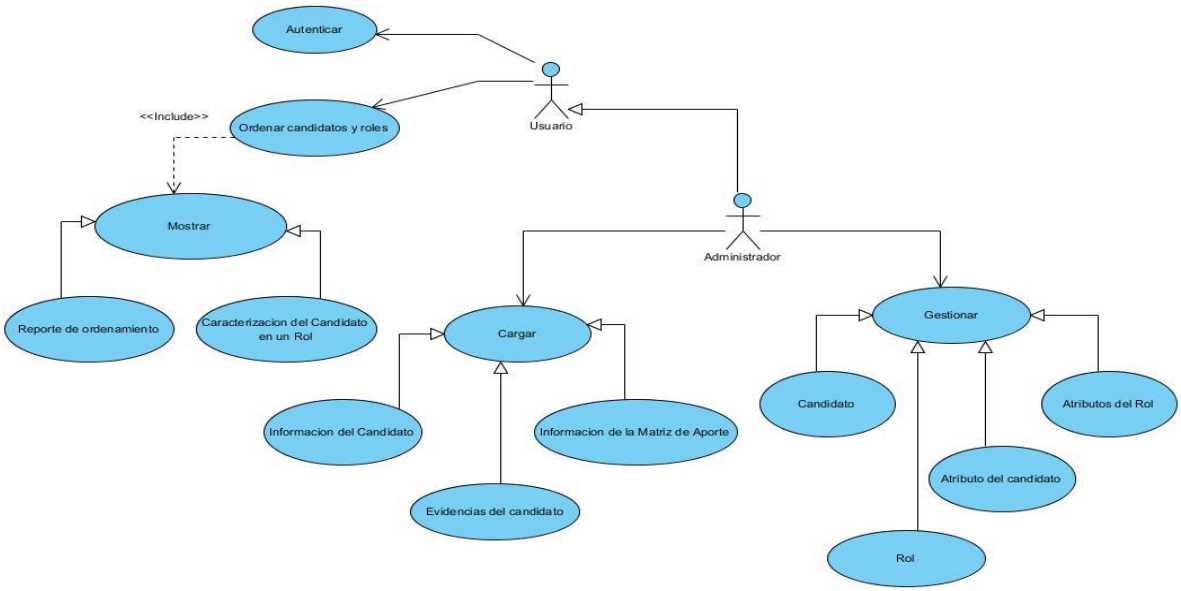



**Figura 20: Historias de Usuario del Negocio**

## Anexo 3 Plantilla de historias de usuarios

| **Historia de Usuario** | |
|---|---|
| **Número:** HU_3 | **Nombre Historia de Usuario:** Cargar información de la matriz de aporte |
| **Modificación de Historia de Usuario Número:** Ninguna | |
| **Usuario:** Dianela Borrego León | **Iteración Asignada:** Sprint 2 |
| **Prioridad en Negocio:** Muy Alta | **Puntos Estimados:** 3 |
| **Riesgo en Desarrollo:** Alto | **Puntos Reales:** 3 |
| **Descripción:** *El usuario administrador del sistema especifica el fichero Excel en el que se encuentra la información de la matriz de aporte, esta información se lee y se persiste en la base de datos para su posterior utilización.* | |
| **Observaciones:** | |
| **Prototipo de interfaces:**     | |

**Tabla 20: Historia de Usuario Cargar Información de la matriz de aporte.**

| **Historia de Usuario** | |
|---|---|
| **Número:** HU_4 | **Nombre Historia de Usuario:** Cargar evidencias del candidato |
| **Modificación de Historia de Usuario Número:** Ninguna | |
| **Usuario:** Leanet Arza Pérez | **Iteración Asignada:** Sprint 2 |
| **Prioridad en Negocio:** Muy Alta | **Puntos Estimados:** 3 |
| **Riesgo en Desarrollo:** Alto | **Puntos Reales:** 3 |
| **Descripción:** *El usuario administrador del sistema especifica el fichero Excel en el que se encuentra la información de las evidencias del candidato, esta información se lee y se persiste en la base de datos para su* | |

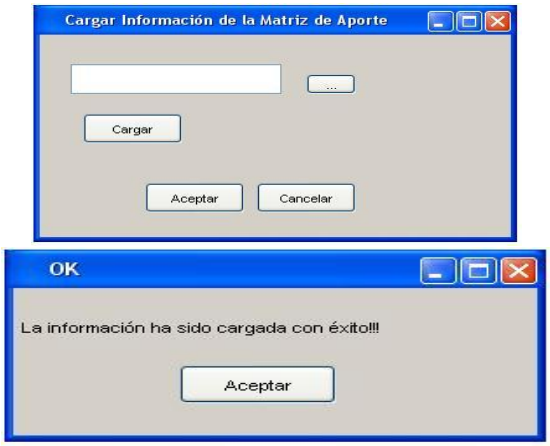
| |
|---|
| *posterior utilización.* |
| **Observaciones:** |
| **Prototipo de interfaces:**   |

**Tabla 21: Historia de Usuario Cargar Evidencias del candidato.**

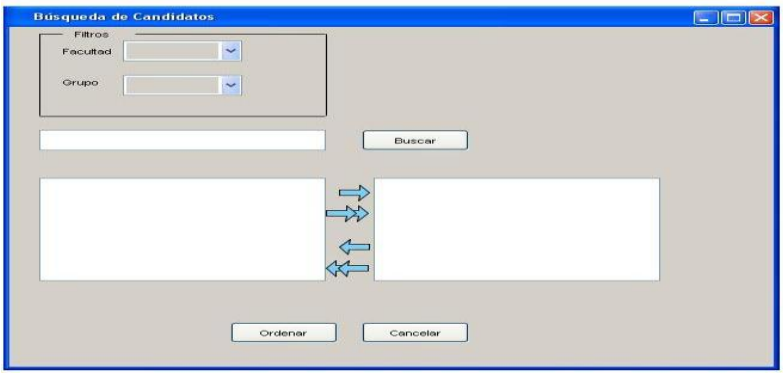
| **Historia de Usuario** | |
|---|---|
| **Número:** HU_6 | **Nombre Historia de Usuario:** Mostrar Reporte de Ordenamiento |
| **Modificación de Historia de Usuario Número:** Ninguna | |
| **Usuario:** *Leanet Arza Pérez* | **Iteración Asignada:** Sprint 3 |
| **Prioridad en Negocio:** *Alta* | **Puntos Estimados:**3 |
| **Riesgo en Desarrollo:** Alto | **Puntos Reales:** 3 |
| **Descripción:** *De acuerdo a los permisos del usuario se muestra una pantalla que permite escoger el grupo de usuarios del cual se quiere obtener el reporte.*<br>*El reporte debe mostrar para cada uno de los candidatos el orden de los roles ordenados de menor a mayor según el índice de acercamiento al rol. Se debe mostrar en el orden el rol y la etiqueta lingüística asociada a su IAR.* | |
| **Observaciones:** | |
| **Prototipo de interfaces:**   | |

**Tabla 22: Historia de Usuario Mostrar Reportes de Ordenamiento.**

| **Historia de Usuario** | |
|---|---|
| **Número:** HU_7 | **Nombre Historia de Usuario:** Mostrar Caracterización del candidato en un rol. |
| **Modificación de Historia de Usuario Número:** Ninguna | |
| **Usuario:** Dianela Borrego León | **Iteración Asignada:** Sprint 3 |
| **Prioridad en Negocio:** *Alta* | **Puntos Estimados:** 3 |
| **Riesgo en Desarrollo:** Alto | **Puntos Reales:** 3 |
| **Descripción:** *El usuario selecciona uno de los roles de un candidato, puede ser a partir de la tabla resultado de la HU_6, esta caracterización muestra el resultado de la evaluación del candidato en cada uno de los atributos del rol.* | |
| **Observaciones:** | |
| **Prototipo de interfaces:** | |


**Tabla 23: Historia de Usuario Mostrar Caracterización del candidato en un rol.**

| **Historia de Usuario** | |
|---|---|
| **Número:** HU_8 | **Nombre Historia de Usuario:** Gestionar atributos del candidato. |
| **Modificación de Historia de Usuario Número:** Ninguna | |
| **Usuario:** Dianela Borrego León | **Iteración Asignada:** Sprint 4 |
| **Prioridad en Negocio:** *Media* | **Puntos Estimados:** 1 |
| **Riesgo en Desarrollo:** Medio | **Puntos Reales:** 1 |
| **Descripción:** *Se realizan las funcionalidades de insertar, modificar y eliminar los atributos del candidato.* | |
| **Observaciones:** | |
| **Prototipo de interfaces:** | |

**Tabla 24: Historia de Usuario Mostrar Gestionar atributos del candidato.**

| **Historia de Usuario** | |
|---|---|
| **Número:** HU_9 | **Nombre Historia de Usuario:** Gestionar atributos del rol. |
| **Modificación de Historia de Usuario Número:** Ninguna | |
| **Usuario:** Leanet Arza Pérez | **Iteración Asignada:** Sprint 4 |
| **Prioridad en Negocio:** *Media* | **Puntos Estimados:** 1 |
| **Riesgo en Desarrollo:** Medio | **Puntos Reales:** 1 |
| **Descripción:** *Se realizan las funcionalidades de insertar, modificar y eliminar los atributos del rol.* | |
| **Observaciones:** | |
| **Prototipo de interfaces:** | |

**Tabla 25: Historia de Usuario Gestionar atributos del rol.**

| **Historia de Usuario** | |
|---|---|
| **Número:** HU_10 | **Nombre Historia de Usuario:** Gestionar candidatos. |
| **Modificación de Historia de Usuario Número: Ninguna** | |
| **Usuario: Dianela Borrego León** | **Iteración Asignada:** Sprint 4 |
| **Prioridad en Negocio: *Media*** | **Puntos Estimados:** 1 |
| **Riesgo en Desarrollo:** Medio | **Puntos Reales:** 1 |
| **Descripción:** *Se realizan las funcionalidades de insertar, modificar y eliminar candidatos.* | |
| **Observaciones:** | |
| **Prototipo de interfaces:** | |

**Tabla 26: Historia de Usuario Gestionar candidatos.**

| **Historia de Usuario** | |
|---|---|
| **Número:** HU_11 | **Nombre Historia de Usuario:** Gestionar roles. |
| **Modificación de Historia de Usuario Número:** Ninguna | |
| **Usuario:** Leanet Arza Pérez | **Iteración Asignada:** Sprint 4 |
| **Prioridad en Negocio: *Media*** | **Puntos Estimados:** 1 |
| **Riesgo en Desarrollo: Medio** | **Puntos Reales:** 1 |
| **Descripción:** *Se realizan las funcionalidades de insertar, modificar y eliminar roles.* | |
| **Observaciones:** | |
| **Prototipo de interfaces:** | |

**Tabla 27: Historia de Usuario Gestionar roles.**

## Anexo 4 Plantilla Tarea de Ingeniería

| **Tarea de Ingeniería** | |
|---|---|
| **Número Tarea:** T_4 | **Número Historia de Usuario:** HU_1 |
| **Nombre Tarea:** Diseñar las interfaces requeridas para la funcionalidad autenticación de usuario. | |
| **Tipo de Tarea :** Desarrollo | **Puntos Estimados:** 1 |
| **Fecha Inicio:** 1-marzo-2012 | **Fecha Fin:** 3-marzo-2012 |

| | |
|---|---|
| **Programador Responsable:** Dianela Borrego León | |
| **Descripción:** *Crear las interfaces necesarias, teniendo en cuenta los prototipos definidos en la HU* | |

**Tabla 28: Tarea de Ingeniería 4.**

| **Tarea de Ingeniería** | |
|---|---|
| **Número Tarea:** T_5 | **Número Historia de Usuario:** HU_1 |
| **Nombre Tarea:** Implementar funcionalidad de autenticación de usuarios | |
| **Tipo de Tarea :** Desarrollo | **Puntos Estimados:** 1 |
| **Fecha Inicio:** 3-marzo-2012 | **Fecha Fin:** 8-marzo-2012 |
| **Programador Responsable:** Dianela Borrego León | |
| **Descripción:** *Crear las clases, interfaces y algoritmos necesarios para la implementación de la funcionalidad de autenticación de usuario.* | |

**Tabla 29: Tarea de Ingeniería 5.**

| **Tarea de Ingeniería** | |
|---|---|
| **Número Tarea:** T_6 | **Número Historia de Usuario:** HU_2, HU_3, HU_4 |
| **Nombre Tarea:** Estudio relacionado con el manejo de ficheros Excel desde Java. | |
| **Tipo de Tarea :** Desarrollo | **Puntos Estimados:** 1 |
| **Fecha Inicio:** 1-marzo-2012 | **Fecha Fin:** 5-marzo-2012 |
| **Programador Responsable:** Leanet Arza Pérez | |
| **Descripción:** *Se estudian las formas de manejar información contenida en ficheros Excel. Diseñar el algoritmo y definir las sentencias a utilizar para implementar las funcionalidades que requieren de este conocimiento.* | |

**Tabla 30: Tarea de Ingeniería 6.**

| **Tarea de Ingeniería** | |
|---|---|
| **Número Tarea:** T_7 | **Número Historia de Usuario:** HU_2, HU_3, HU_4 |
| **Nombre Tarea:** Diseñar la interfaz necesarias | |
| **Tipo de Tarea :** Desarrollo | **Puntos Estimados:** 1 |
| **Fecha Inicio:** 10-marzo-2012 | **Fecha Fin:** 15-marzo-2012 |
| **Programador Responsable:** Leanet Arza Pérez | |
| **Descripción:** *Se diseñan las interfaces necesarias para la implementación de las funcionalidades relacionadas con la carga de la información, teniendo en cuenta los prototipos diseñados en las HU.* | |

**Tabla 31: Tarea de Ingeniería 7.**

| **Tarea de Ingeniería** | |
|---|---|
| **Número Tarea:** T_8 | **Número Historia de Usuario:** HU_2 |
| **Nombre Tarea: Implementar Cargar Información del** Candidato | |
| **Tipo de Tarea :** Desarrollo | **Puntos Estimados:** 1 |
| **Fecha Inicio:** 15-marzo-2012 | **Fecha Fin:** 20-marzo-2012 |
| **Programador Responsable:** Leanet Arza Pérez | |
| **Descripción:** *Se implementa la funcionalidad vinculada con la historia de usuario.* | |

**Tabla 32: Tarea de Ingeniería 8.**

| **Tarea de Ingeniería** | |
|---|---|
| **Número Tarea:** T_9 | **Número Tarea:** T_9 |
| **Nombre Tarea:** Implementar Cargar Información de la matriz de aporte | |
| **Tipo de Tarea :** Desarrollo | **Tipo de Tarea :** Desarrollo |
| **Fecha Inicio:** 20-marzo-2012 | **Fecha Inicio:** 20-marzo-2012 |
| **Programador Responsable:** Leanet Arza Pérez | |
| **Descripción:** *Se implementa la funcionalidad relacionada con la historia de usuario*. | |

**Tabla 33: Tarea de Ingeniería 9.**

| **Tarea de Ingeniería** | |
|---|---|
| **Número Tarea:** T_10 | **Número Historia de Usuario:** HU_4 |
| **Nombre Tarea:** Implementar Cargar Evidencias del Candidato | |
| **Tipo de Tarea :** Desarrollo | **Puntos Estimados:** 1 |
| **Fecha Inicio:** 25-marzo-2012 | **Fecha Fin:** 30-marzo-2012 |
| **Programador Responsable:** Leanet Arza Pérez | |
| **Descripción:** *Se implementa la funcionalidad vinculada con la historia de usuario.* | |

**Tabla 34: Tarea de Ingeniería 10.**

| **Tarea de Ingeniería** | |
|---|---|
| **Número Tarea:** T_13 | **Número Historia de Usuario:** HU_6 |
| **Nombre Tarea:** Diseñar las interfaces necesarias para mostrar reportes de ordenamiento | |
| **Tipo de Tarea :** Desarrollo | **Puntos Estimados:** 1 |

| | |
|---|---|
| **Fecha Inicio:** 1-abril-2012 | **Fecha Fin:** 5-abril-2012 |
| **Programador Responsable:** Leanet Arza Pérez | |
| **Descripción:** *Se diseñan las funcionalidades necesarias para la implementación de la HU_6, teniendo en cuenta los prototipos diseñados* | |

**Tabla 35: Tarea de Ingeniería 13.**

| Tarea de Ingeniería | |
|---|---|
| **Número Tarea:** T_14 | **Número Historia de Usuario:** HU_6 |
| **Nombre Tarea:** Implementar Mostrar reportes de ordenamiento | |
| **Tipo de Tarea : Desarrollo** | **Puntos Estimados:** 1 |
| **Fecha Inicio:** 5-abril-2012 | **Fecha Fin:** 15-abril-2012 |
| **Programador Responsable:** Leanet Arza Pérez | |
| **Descripción:** *Se implementa la funcionalidad vinculada con la historia de usuario.* | |

**Tabla 36: Tarea de Ingeniería 14.**

| Tarea de Ingeniería | |
|---|---|
| **Número Tarea:** T_15 | **Número Historia de Usuario:** HU_7 |
| **Nombre Tarea:** Diseñar las interfaces necesarias para mostrar caracterización del candidato en un rol | |
| **Tipo de Tarea :** Desarrollo | **Puntos Estimados:** 1 |
| **Fecha Inicio:** 11-abril-2012 | **Fecha Fin:** 15-abril 2012 |
| **Programador Responsable:** Dianela Borrego León | |
| **Descripción:** *Se diseñan las funcionalidades necesarias para la implementación de la HU_7, teniendo en cuenta los prototipos diseñados* | |

**Tabla 37: Tarea de Ingeniería 15.**

| Tarea de Ingeniería | |
|---|---|
| **Número Tarea:** T_16 | **Número Historia de Usuario:** HU_7 |
| **Nombre Tarea:** Implementar Mostrar caracterización del candidato en un rol | |
| **Tipo de Tarea :** Desarrollo | **Puntos Estimados:** 1 |
| **Fecha Inicio:** 16-abril-2012 | **Fecha Fin:** 25-abril 2012 |
| **Programador Responsable:** Dianela Borrego León | |
| **Descripción:** *Se implementa la funcionalidad vinculada con la historia de usuario.* | |

**Tabla 38: Tarea de Ingeniería 16.**

## Anexo 5 Descripción del Modelo de Datos

| **Nombre Tabla:** candidato | | |
|---|---|---|
| **Descripción:** En esta tabla se almacenan los datos correspondientes a un candidato. | | |
| **Atributos** | **Tipo** | **Descripción** |
| id_Candidato | Int | Es el identificador del candidato. Es la llave primario de la tabla. |
| ci | String | Es el carnet de identidad de un candidato. |
| primer_nombre | String | Es el primer nombre de un candidato. |
| segundo_nombre | String | Es el segundo nombre de un candidato en caso de que tenga. |
| primer_apellido | String | Es el primer apellido de un candidato. |
| segundo_apellido | String | Es el segundo apellido de un candidato. |
| id_expediente | String | Es el identificador del candidato en su área. |
| sexo | Char | Es el sexo del candidato, pudiendo se F o M. |
| id_area | Int | Es una llave foránea de la relación que existe entre la tabla candidato y la tabla área. |

**Tabla 39: Tabla candidato.**

| **Nombre Tabla:** variable_lingüística | | |
|---|---|---|
| **Descripción:** En esta tabla se almacenan los datos correspondientes a una variable lingüística. | | |
| **Atributos** | **Tipo** | **Descripción** |
| id_variable | Int | Es el identificador de la variable lingüística. Es la llave primaria de la tabla. |
| nombre_variable | String | Es el nombre de la variable lingüística. |
| descripcion | String | Permite almacenar la variable lingüística por la cual se va a representar el modelo. |
| sigla | String | Son las siglas del nombre de la etiqueta lingüística. |

**Tabla 40: Tabla variable_linguistica.**

| **Nombre Tabla:** etiqueta_lingüística | | |
|---|---|---|
| **Descripción:** En esta tabla se almacenan los datos correspondientes a una etiqueta lingüística. | | |
| **Atributos** | **Tipo** | **Descripción** |
| id_etiqueta | Int | Es el identificador de la etiqueta lingüística. Es la llave primaria de la tabla. |
| valor_minimo | Double | Es el valor mínimo que puede tener una etiqueta lingüística. |

| | | |
|---|---|---|
| valor_maximo | Double | Es el valor máximo que puede tener una etiqueta lingüística. |
| valor_modal | Double | Es el valor modal que puede tener una etiqueta lingüística. |
| nombre_etiqueta | String | Es el valor mínimo que puede tener una etiqueta lingüística. |
| sigla | String | Son las siglas del nombre de la etiqueta lingüística. |
| id_variable | Int | Es la llave foránea de la relación existente entre la tabla etiqueta_ lingüística y la tabla variable_ lingüística. |

**Tabla 41: Tabla etiqueta_linguistica.**

| **Nombre Tabla:** tipo | | |
|---|---|---|
| **Descripción:** En esta tabla se almacenan los datos correspondientes a tipo. | | |
| **Atributos** | **Tipo** | **Descripción** |
| id_tipo | Int | Es el identificador del tipo. Es la llave primaria de la tabla. |
| tipo | String | Es el tipo de atributo del rol. |

**Tabla 42: Tabla tipo.**

| **Nombre Tabla:** atributo_rol | | |
|---|---|---|
| **Descripción:** En esta tabla se almacenan los datos correspondientes a un atributo del rol. | | |
| **Atributos** | **Tipo** | **Descripción** |
| id_atributo_rol | Int | Es el identificador del atributo del rol. Es la llave primaria de la tabla. |
| sigla | String | Son las siglas del atributo del rol. |
| nombre_atributo_rol | String | Es el nombre del atributo del rol. |
| descripción | String | Describe a cada atributo del rol. |
| id_variable | String | Es la llave foránea de la relación existente entre la tabla atributo_rol y la tabla variable_ lingüística. |
| id_tipo | String | Es la llave foránea de la relación existente entre la tabla atributo_rol y la tabla tipo. |
| id_rol | Int | Es la llave foránea de la relación existente entre la tabla atributo_rol y la tabla rol. |

**Tabla 43: Tabla atributo_rol.**

| **Nombre Tabla:** atributo_candidato | | |
|---|---|---|
| **Descripción:** En esta tabla se almacenan los datos correspondientes a un atributo del candidato. | | |
| **Atributos** | **Tipo** | **Descripción** |
| id_atributo_candidato | Int | Es el identificador del atributo del candidato. Es la llave primaria de la tabla. |
| nombre_atributo_candidato | String | Es el nombre del atributo del candidato. |
| descripción | String | Describe a cada atributo del candidato. |
| valor_maximo | Int | Es el valor máximo que pueda tener el atributo del candidato. |
| activo | Booleano | Es para saber si el atributo del candidato está activo o no. |
| id_variable | Int | Es la llave foránea de la relación existente entre la tabla atributo_candidato y la tabla variable_ lingüística. |

**Tabla 44: Tabla atributo_candidato.**

| **Nombre Tabla:** caso | | |
|---|---|---|
| **Descripción:** En esta tabla se almacenan los datos correspondientes al caso. | | |
| **Atributos** | **Tipo** | **Descripción** |
| id_caso | Int | Es el identificador del caso. Es la llave primaria de la tabla. |
| caso | String | Cada caso representa un ideal que el candidato debe cumplir. |
| id_rol | Int | Es la llave foránea de la relación existente entre la tabla caso y la tabla rol. |

**Tabla 45: Tabla caso.**

| **Nombre Tabla:** valor_atributo_caso | | |
|---|---|---|
| **Descripción:** En esta tabla se almacenan los datos correspondientes del valor del atributo del rol con el caso. | | |
| **Atributos** | **Tipo** | **Descripción** |
| valor_minimo | Double | Es el valor mínimo que puede tener un caso en los atributos del rol. |
| valor_maximo | Double | Es el valor máximo que puede tener un caso en los atributos del rol. |
| valor_modal | Double | Es el valor modal que puede tener un caso en los atributos del rol. |
| id_caso | Int | Es la llave primaria por la relación existente entre la tabla atributo_candidato y la tabla atributo_rol. |
| id_atributo_rol | Int | Es la llave primariapor la relación existente entre la tabla atributo_candidato y la tabla atributo_rol. |

**Tabla 46: Tabla valor_atributo_caso.**

| **Nombre Tabla:** rol | | |
|---|---|---|
| **Descripción:** En esta tabla se almacenan los datos correspondientes al rol. | | |
| **Atributos** | **Tipo** | **Descripción** |
| id_rol | Int | Es el identificador del rol. Es la llave primaria de la tabla. |

| | | |
|---|---|---|
| descripción | String | Describe el rol. |
| nombre_rol | String | Es el nombre del rol. |

**Tabla 47: Tabla rol.**

| **Nombre Tabla:** atributo_candidato_atributo_rol | | |
|---|---|---|
| **Descripción:** En esta tabla se almacenan los datos correspondientes al atributo del candidato con el atributo del rol. | | |
| **Atributos** | **Tipo** | **Descripción** |
| id_ atributo_candidato | Int | Es la llave primaria por la relación existente entre la tabla atributo_candidato y la tabla atributo_rol. |
| id_ atributo_rol | Int | Es la llave primaria por la relación existente entre la tabla atributo_candidato y la tabla atributo_rol. |
| valor_minimo | Double | Es el valor mínimo que puede tener un caso en los atributos del rol. |
| valor_maximo | Double | Es el valor máximo que puede tener un caso en los atributos del rol. |
| valor_modal | Double | Es el valor modal que puede tener un caso en los atributos del rol. |
| Id_etiqueta | int | Es la llave foránea de la relación entre atributo_candidato_atributo_rol y etiqueta_linguistica. |

**Tabla 48: Tabla atributo_candidato_atributo_rol.**

| **Nombre Tabla:** candidato_atributo_rol | | |
|---|---|---|
| **Descripción:** En esta tabla se almacenan los datos correspondientes al candidato con el atributo del rol. | | |
| **Atributos** | **Tipo** | **Descripción** |
| id_ candidato | Int | Es la llave primaria por la relación existente entre la tabla candidato y la tabla atributo_rol. |
| id_ atributo_rol | Int | Es la llave primaria por la relación existente entre la tabla candidato y la tabla atributo_rol. |
| valor_minimo | Double | Es el valor mínimo que puede tener un candidato en los atributos del rol. |
| valor_maximo | Double | Es el valor máximo que puede tener un candidato en los atributos del rol. |
| valor_modal | Double | Es el valor modal que puede tener un candidato en los atributos del rol. |
| id_etiqueta | Int | Es la llave foránea de la relación existente entre la tabla candidato_atr_rol y la tabla etiqueta_ lingüística. |

**Tabla 49: Tabla candidato_atributo_rol.**

| **Nombre Tabla:** valor_candidato_atributo_candidato | | |
|---|---|---|
| **Descripción:** En esta tabla se almacenan los datos correspondientes al caso. | | |
| **Atributos** | **Tipo** | **Descripción** |
| id_ candidato | Int | Es la llave primariapor la relación existente entre la tabla candidato y la tabla atributo_candidato. |
| id_ atributo_candidato | Int | Es la llave primariapor la relación existente entre la tabla candidato y la tabla atributo_candidato. |
| valor | Double | Es el valor que tiene el candidato con el atributo del candidato. |
| id_etiqueta | Int | Es la llave foránea de la relación existente entre la tabla valor_candidato_atr_candidato y la tabla etiqueta_ lingüística. |

**Tabla 50: Tabla valor_candidato_atributo_candidato.**

| **Nombre Tabla:** area | | |
|---|---|---|
| **Descripción:** En esta tabla se almacenan los datos correspondientes al área. | | |
| **Atributos** | **Tipo** | **Descripción** |
| id_area | Int | Es el identificador del área. Es la llave primaria de la tabla. |
| nombre_area | String | Es el nombre del área. |

**Tabla 51: Tabla area.**

| **Nombre Tabla:** candidato_rol | | |
|---|---|---|
| **Descripción:** En esta tabla se almacenan los datos correspondientes al diagnóstico. | | |
| **Atributos** | **Tipo** | **Descripción** |
| id_rol | Int | Es la llave primariapor la relación existente entre la tabla candidato y la tabla rol. |
| id_candidato | Int | Es la llave primariapor la relación existente entre la tabla candidato y la tabla rol. |
| valor | Int | Es el valor que tiene el candidato en el rol. |
| id_etiqueta | Int | Es la llave foránea de la relación existente entre la tabla candidato_rol y la tabla etiqueta_ lingüística. |

**Tabla 52: Tabla candidato_rol.**

## Anexo 6 Descripción de las clases

| **Nombre Clase: ApoloServiceImpl** | |
|---|---|
| **Tipo de clase:** Controladora | |
| **Atributos** | **Tipo** |
| **Nombre Responsabilidad** | **Descripción** |

| | |
|---|---|
| indiceAcercamientoRol | Calcula cuán cerca está un candidato a cada uno de los roles. |
| ordenar | Ordena de menor a mayor los roles de acuerdo al índice de acercamiento de cada candidato a ese rol. |
| predecir | Persiste en la base de datos el índice de acercamiento del candidato en el rol. |
| evaluarCandidato | Persiste en la base de datos la evaluación del candidato en un atributo del rol. |
| recomendacion | Ejecuta los métodos evaluarCandidato, persistir y ordenar. |

**Tabla 53: Clase ApoloServiceImpl.**

| **Nombre Clase: Candidato** | |
|---|---|
| **Tipo de clase:** Entidad | |
| **Atributos** | **Tipo** |
| idCandidato | Int |
| Ci | String |
| primerNombre | String |
| segundoNombre | String |
| primerApellido | String |
| segundoApellido | String |
| idExpediente | String |
| Sexo | Char |

**Tabla 54: Clase Candidato.**

| **Nombre Clase: Rol** | |
|---|---|
| **Tipo de clase:** Entidad | |
| **Atributos** | **Tipo** |
| idRol | Int |
| nombreRol | String |
| descripción | String |

**Tabla 55: Clase Rol.**

| **Nombre Clase: AtributoCandidato** | |
|---|---|
| **Tipo de clase:** Entidad | |
| **Atributos** | **Tipo** |
| idAtributoCandidato | Int |
| nombreAtributoCandidato | String |
| Descripción | String |
| valorMaximo | Double |
| activo | String |

**Tabla 56: Clase AtributoCandidato.**

| **Nombre Clase: AtributoRol** | |
|---|---|
| **Tipo de clase:** Entidad | |
| **Atributos** | **Tipo** |
| idAtributoRol | Int |
| nombreAtributoRol | String |
| descripción | String |

**Tabla 57: Clase AtributoRol.**

| **Nombre Clase: Caso** | |
|---|---|
| **Tipo de clase:** Entidad | |
| **Atributos** | **Tipo** |
| idCaso | Int |
| caso | String |

**Tabla 58: Clase Caso.**

| **Nombre Clase: Area** | |
|---|---|
| **Tipo de clase:** Entidad | |
| **Atributos** | **Tipo** |
| idArea | Int |
| nombreArea | String |

**Tabla 59: Clase Area.**

| **Nombre Clase: Tipo** | |
|---|---|
| **Tipo de clase:** Entidad | |
| **Atributos** | **Tipo** |
| idTipo | Int |
| tipo | String |

**Tabla 60: Clase Tipo.**

| **Nombre Clase: EtiquetaLinguistica** | |
|---|---|
| **Tipo de clase:** Entidad | |
| **Atributos** | **Tipo** |
| idEtiqueta | Int |
| valorMaximo | Double |
| valorModal | Double |
| valorMinimo | Double |
| nombreEtiqueta | String |
| sigla | String |

**Tabla 61: Clase EtiquetaLinguistica.**

| **Nombre Clase: VariableLinguistica** | |
|---|---|
| **Tipo de clase:** Entidad | |
| **Atributos** | **Tipo** |
| idVariable | Int |
| descripcion | String |
| nombreVariable | String |
| sigla | String |

**Tabla 62: Clase VariableLinguistica.**

## Anexo 7 Plantilla Caso de Prueba de Aceptación

| Caso de Prueba de Aceptación | |
|---|---|
| **Código Caso de Prueba:** A-3-1 | **Nombre Historia de Usuario:** Cargar Información de la matriz de aporte. |
| **Nombre de la persona que realiza la prueba:** Leanet Arza Pérez | |
| **Descripción de la Prueba:** Se ejecuta la prueba y se verifica que el administrador del sistema cargue correctamente la información de la matriz de aporte. | |
| **Condiciones de Ejecución:**<br>• Se comprueba que el archivo a cargar esté en el formato establecido. | |
| **Entrada / Pasos de ejecución:** Al elegir la opción de examinar, el sistema debe dar la posibilidad de buscar el lugar donde se encuentra el archivo que se decida cargar. | |
| **Resultado Esperado:** Que el administrador del sistema cargue correctamente la información referente a la matriz de aporte. | |
| **Evaluación de la Prueba:** | |

**Tabla 63: Caso de Prueba de Aceptación 3-1.**

| Caso de Prueba de Aceptación | |
|---|---|
| **Código Caso de Prueba:** A-4-1 | **Nombre Historia de Usuario:** Cargar Evidencias del Candidato. |
| **Nombre de la persona que realiza la prueba:** Dianela Borrego León | |
| **Descripción de la Prueba:** Se ejecuta la prueba y se verifica que el administrador del sistema cargue correctamente las evidencias de los candidatos. | |
| **Condiciones de Ejecución:**<br>• Se comprueba que el archivo a cargar esté en el formato establecido. | |
| **Entrada / Pasos de ejecución:** Al elegir la opción de examinar, el sistema debe dar la posibilidad de buscar el lugar donde se encuentra el archivo que se decida cargar. | |

| |
|---|
| **Resultado Esperado:** Que el administrador del sistema cargue correctamente las evidencias de los candidatos. |
| **Evaluación de la Prueba:** |

**Tabla 64: Caso de Prueba de Aceptación 4-1.**

| Caso de Prueba de Aceptación | |
|---|---|
| **Código Caso de Prueba:** A-6-1 | **Nombre Historia de Usuario:** Mostrar reporte de ordenamiento. |
| **Nombre de la persona que realiza la prueba:** Dianela Borrego León | |
| **Descripción de la Prueba:** Se ejecuta la prueba y se verifica que se muestre correctamente la matriz ordenada de los candidatos según acercamiento a cada uno de los roles. | |
| **Condiciones de Ejecución:**<br>• Que se haya seleccionado algún candidato.<br>• Que se haya seleccionado la opción de predecir. | |
| **Entrada / Pasos de ejecución:** Para la ejecución de esta historia se debe pasar a la selección del o de los candidatos de interés, luego se selecciona la opción de predecir. | |
| **Resultado Esperado:** Se muestra una tabla con los candidatos seleccionados ordenados según su acercamiento a cada uno de los roles. | |
| **Evaluación de la Prueba:** | |

**Tabla 65: Caso de Prueba de Aceptación 6-1.**

| Caso de Prueba de Aceptación | |
|---|---|
| **Código Caso de Prueba:** A-7-1 | **Nombre Historia de Usuario:** Mostrar caracterización del candidato en un rol. |
| **Nombre de la persona que realiza la prueba:** Leanet Arza Pérez | |
| **Descripción de la Prueba:** Se ejecuta la prueba y se verifica que se muestre correctamente la caracterización del candidato en un rol específico. | |
| **Condiciones de Ejecución:**<br>• Que se haya seleccionado un candidato en un rol específico.<br>• Que se haya seleccionado la opción de mostrar caracterización. | |
| **Entrada / Pasos de ejecución:** Para la ejecución de esta historia se debe pasar a la selección de un rol correspondiente a algún candidato de interés, luego se selecciona la opción de mostrar caracterización. | |
| **Resultado Esperado:** Se muestra una interfaz con la caracterización del candidato en un rol seleccionado. | |
| **Evaluación de la Prueba:** | |

**Tabla 66: Caso de Prueba de Aceptación 7-1.**

| Caso de Prueba de Aceptación | |
|---|---|
| **Código Caso de Prueba:** A-8-1 | **Nombre Historia de Usuario:** Gestionar atributos del candidato. |
| **Nombre de la persona que realiza la prueba:** Dianela Borrego León | |
| **Descripción de la Prueba:** Se ejecuta la prueba y se verifica que los atributos del candidato sean insertados, modificados y eliminados correctamente. | |
| **Condiciones de Ejecución:**<br>• Que exista el atributo de candidato. | |
| **Entrada / Pasos de ejecución:** Para la ejecución de esta historia se debe obtener la información requerida y pasar a insertar, modificar y eliminar esta información. | |
| **Resultado Esperado:** Comprobar en la base de datos que se haya insertado, modificado y eliminado la información correctamente. | |
| **Evaluación de la Prueba:** | |

**Tabla 67: Caso de Prueba de Aceptación 8-1.**

| Caso de Prueba de Aceptación | |
|---|---|
| **Código Caso de Prueba:** A-9-1 | **Nombre Historia de Usuario:** Gestionar atributos del rol. |
| **Nombre de la persona que realiza la prueba:** Leanet Arza Pérez | |
| **Descripción de la Prueba:** Se ejecuta la prueba y se verifica que los atributos del rol sean insertados, modificados y eliminados correctamente. | |
| **Condiciones de Ejecución:**<br>• Que exista el atributo de rol. | |
| **Entrada / Pasos de ejecución:** Para la ejecución de esta historia se debe obtener la información requerida y pasar a insertar, modificar y eliminar esta información. | |
| **Resultado Esperado:** Comprobar en la base de datos que se haya insertado, modificado y eliminado la información correctamente. | |
| **Evaluación de la Prueba:** | |

**Tabla 68: Caso de Prueba de Aceptación 9-1.**

| Caso de Prueba de Aceptación |
|---|
| **Código Caso de Prueba:** A-10-1 **Nombre Historia de Usuario:** Gestionar candidato. |
| **Nombre de la persona que realiza la prueba:** Leanet Arza Pérez |
| **Descripción de la Prueba:** Se ejecuta la prueba y se verifica que los candidatos sean insertados, modificados y eliminados correctamente. |
| **Condiciones de Ejecución:**<br>• Que exista el candidato. |
| **Entrada / Pasos de ejecución:** Para la ejecución de esta historia se debe obtener la información requerida y pasar a insertar, modificar y eliminar esta información. |
| **Resultado Esperado:** Comprobar en la base de datos que se haya insertado, modificado y eliminado la información correctamente. |
| **Evaluación de la Prueba:** |

**Tabla 69: Caso de Prueba de Aceptación 10-1.**

| Caso de Prueba de Aceptación |
|---|
| **Código Caso de Prueba:** A-11-1 **Nombre Historia de Usuario:** Gestionar roles. |
| **Nombre de la persona que realiza la prueba:** Dianela Borrego León |
| **Descripción de la Prueba:** Se ejecuta la prueba y se verifica que los roles sean insertados, modificados y eliminados correctamente. |
| **Condiciones de Ejecución:**<br>• Que exista el rol. |
| **Entrada / Pasos de ejecución:** Para la ejecución de esta historia se debe obtener la información requerida y pasar a insertar, modificar y eliminar esta información. |
| **Resultado Esperado:** Comprobar en la base de datos que se haya insertado, modificado y eliminado la información correctamente. |
| **Evaluación de la Prueba:** |

**Tabla 70: Caso de Prueba de Aceptación 11-1.**

## Anexo 8 Interfaces de Usuario

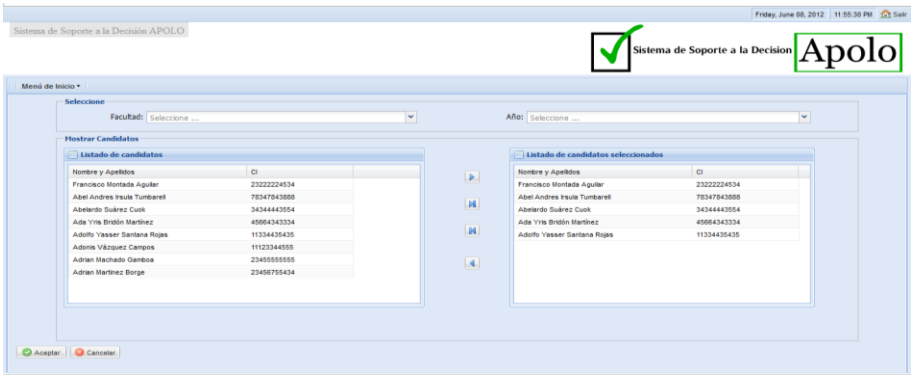



**Figura 21: Interfaz relacionada con la HU Mostrar Caracterización del candidato en un rol.**

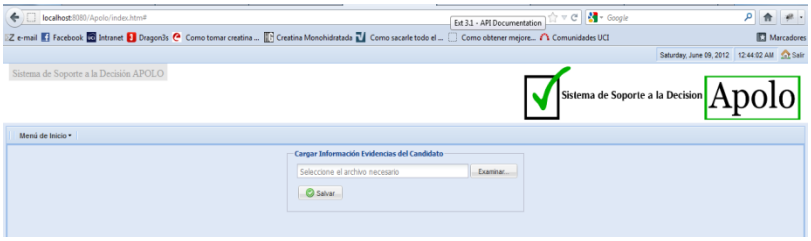



**Figura 22: Interfaz relacionada con la HU Cargar evidencias del candidato**


**Figura 23: Interfaz relacionada con la HU Cargar información del candidato**

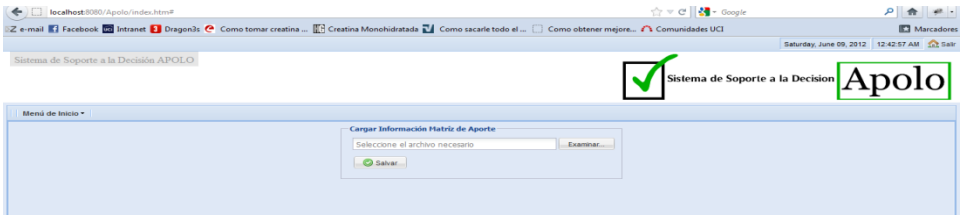



**Figura 24: Interfaz relacionada con la HU Cargar información de la matriz de aporte**

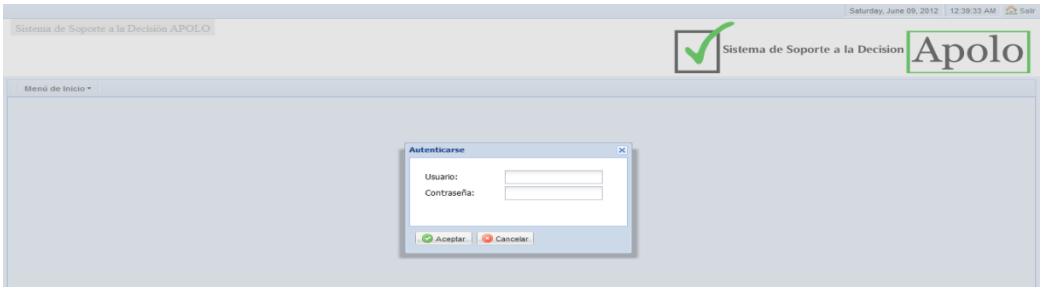



**Figura 25: Interfaz relacionada con la HU Autenticación de usuarios**

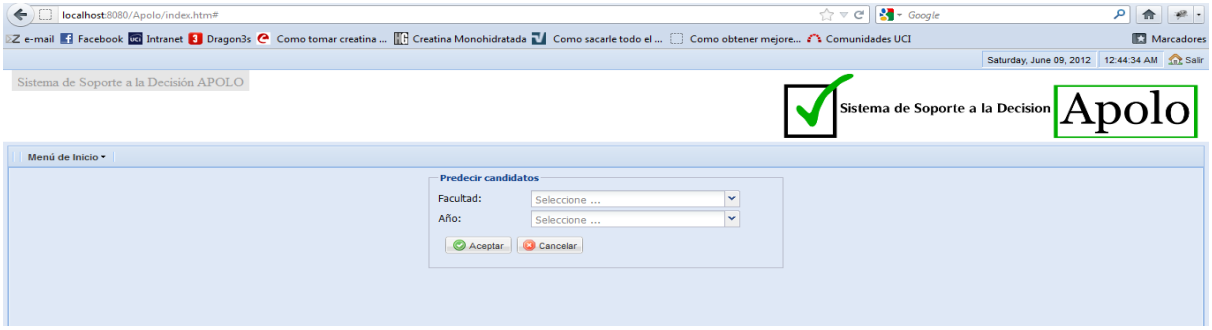



**Figura 26: Interfaz relacionada con las HU Ordenar roles y candidatos y Mostrar Reporte de Ordenamiento**

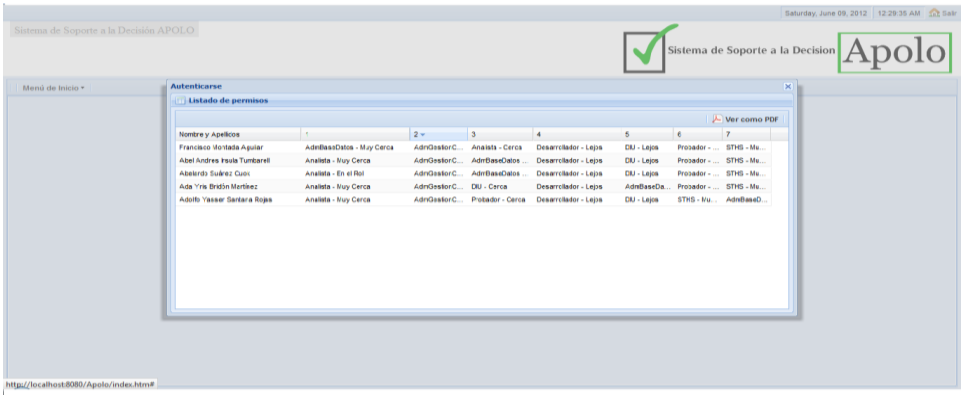



**Figura 27: Interfaz relacionada con la HU Autenticación de usuarios**