Universidad de las Ciencias Informáticas

Facultad 3

Título: *Sistema Informático para la Gestión de Currículum Vítae basados en modelos internacionales.*

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

Autores: *Dariel Rojas Mantilla*

Rey Manuel Lazo Brito

Tutores: *Ing. Zénel Reyes Pérez*

Ing. Yasel Antonio Romero Piñeiro

Co-Tutor: *Ing. Nolvis Pérez Franco*

"La Habana, Mayo de 2015"

DECLARACIÓN DE AUTORÍA

Por este medio declaramos que Dariel Rojas Mantilla y Rey Manuel Lazo Brito somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas (UCI) para que haga el uso que estime pertinente con este trabajo.

Para que así conste firmamos la presente a los __ días del mes de ___ del año ___.

_____	_____
Firma del Autor	Firma del Autor
Dariel Rojas Mantilla	Rey Manuel Lazo Brito
_____	_____
Firma del Tutor	Firma del Tutor
Ing. Zénel Reyes Pérez	Ing. Yasel Antonio Romero Piñeiro

AGRADECIMIENTOS

Dariel Rojas

Primero que todo agradezco a mi compañero de tesis Rey Manuel, que en estos últimos tiempos ha sido compañero y amigo al mismo tiempo. Sin él no estaría aquí parado.

A mis padres, por entregarme su vida y su amor desmedido, por estar pendientes de mí, y ser los guías y educadores de mi vida. Les agradezco por todo su apoyo y todo el esfuerzo que han hecho por darme lo que he necesitado y mucho más.

A mi otra mamá que también me ha brindado su apoyo sin condiciones, lo ha dado todo por mí. A mi hermano Dainiel que sé que me quiere y que sigue mi ejemplo.

A mi Familia, que de una forma u otra han colaborado porque este momento se hiciera realidad; a mi abuela que siempre ha creído en mí y en todo lo que he sido capaz de hacer.

A mi novia Marlen, que ha sido mi punto de apoyo todo este tiempo en la ausencia de mis progenitores, brindándome su apoyo desmedido y aguantando mis malacrianzas; a su familia, que ya es mi familia.

A mis compañeros por ser tan paciente conmigo, por soportarme todo este tiempo que hemos pasado juntos, al Danny, Barbaro, Zamora, Ebrik, Sandy y si sigo mencionando es por gusto nunca termino.

Muchas gracias a todos mis amigos, compañeros de aula y profesores que de una forma u otra me acompañaron hasta aquí.

Agradecimientos

Rey Manuel

Por sobre todo quiero agradecer a mis padres Marisel y Reinaldo, a mis hermanos el Lachy y Luis May, a mi abuelo Manolete y a mi novia Anayelis por su apoyo y por siempre creer en mí.

Agradecer también a mi amigo y compañero de tesis por la confianza depositada al creer en mí cuando le dije "de que esto sale puedes estar seguro" y aquella frase que siempre nos caracterizó, "Esto son 5 segundos".

A mi familia en toda si totalidad, por estar siempre pendiente de mí y de mis estudios. A mis suegros Carlos y Ana, por el regalo y la confianza que depositaron en mí para cuidarle a su pequeña.

Agradecer a mi grupo 7 completo, a mis amigos de ahora y de siempre, pero en especial a Yani quien a pesar de sus berrinches supo ser una amiga incondicional, Abel, Abdiel, Christiano, Jose Armando, al Migue, Jose el millonario, al Sabro, a Pablito el superfly y a Ivaniet.

A mis tutores gracias por guiarnos durante todo el proceso y el apoyo que nos brindaron. En fin a todo el que se dedicó un pedacito de su tiempo para compartirlo con nosotros, a todos,

GRACIAS.

DEDICATORIA

Dedico este trabajo a todas las personas que siempre han estado a mí lado, en especial a mis padres, hermanos, abuelo, tías, primos y mi novia

por ser ustedes una fuente de inspiración y apoyo.

A todos los amigos y personas que he conocido en el transcurso

de estos años de mi vida por su apoyo cuando lo he

necesitado.

Rey Manuel Lazo Brito.

Dedico este trabajo a todas las personas que siempre me han apoyado y estado a mí lado, a mí familia en general y a mi novia.

Dariel Rojas Mantilla

RESUMEN

La gestión de la información curricular de los trabajadores de la facultad 3 en la Universidad de las Ciencias Informáticas (UCI) se realiza de manera manual, lo que trae consigo que el trabajo sea engorroso y en ocasiones conduzca a la pérdida de información, no se sigue ningún estándar o norma para la redacción del mismo. La búsqueda de esta información depende de la consulta a varios documentos que se encuentran en diferentes formatos. Por tal motivo el presente trabajo propone un sistema que permita gestionar la información curricular de los trabajadores de la facultad 3, basados en los modelos internacionales a partir del registro de información personal y la certificación de las evidencias reflejadas en la misma. Se obtuvo como resultado una aplicación web que gestiona la información curricular de los trabajadores de esta entidad, garantizando una mayor accesibilidad, organización y toma de decisiones oportuna cuando se consulta un currículum vítae.

Palabras claves: certificación de evidencias, currículum vítae, estándares, gestión de información.

ÍNDICE DE CONTENIDOS

Índice de Contenidos

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

A nivel mundial existe una necesidad imperante en todas las empresas e instituciones de gestionar la información de manera correcta. Para ello, el hombre con el transcurso de los años ha perfeccionado y modificado la forma en que la misma es almacenada, procesada y consultada, ya sea con el uso de medios de cómputo o métodos manuales. El carácter competitivo que caracteriza a las organizaciones ha exigido que estas dediquen gran esfuerzo a la búsqueda de una plantilla laboral más calificada, de ahí que las subdirecciones o departamentos de recursos humanos hagan énfasis en la gestión de la información personal de sus trabajadores haciendo uso del currículum vítae. El currículum vitae (CV) es el documento en el cual se expone el historial académico y profesional de un trabajador o aspirante a un puesto de trabajo, de una manera clara, concisa y ordenada. (1)

Para la conformación y creación de dicho documento, se emplean varios modelos entre los que se encuentran: el modelo americano y el europeo. Dichos modelos se definen a partir de un conjunto de normas o estándares que se establecen en la región o país donde se encuentra la institución en la cual se opta por el empleo. (2)

En Cuba aunque la estandarización del CV no está establecida, sí se siguen una serie de normas y especificaciones que deben estar presentes como son: nombre completo, carnet de identidad, dirección y teléfono, según la Gaceta Oficial de la República de Cuba en su número 29; junio de 2014. En algunas entidades no se hace necesario la presentación del mismo, puesto que con solo un título que avale un nivel de estudio en correspondencia con la plaza que se oferte, es suficiente para optar por ella.

En la Universidad de las Ciencias Informáticas (UCI), el currículum es utilizado en disímiles trámites referentes principalmente en la superación profesional de los profesores y especialistas. Los mismos deben presentar dicho documento cuando son convocados a participar en acreditaciones de eventos, cursos de superación tanto nacionales como internacionales y viajes a otros países para el cumplimiento de misiones.

En la Facultad 3 de la UCI, no existe un sistema que registre la información curricular de los trabajadores; lo que hace que esta se realice de manera manual a través de las herramientas Ofimáticas. Ello trae consigo que el trabajo sea engorroso y en ocasiones conduce a pérdidas de documentación, debido a que la información se encuentra en disímiles documentos y en diferentes formatos. Los sistemas informáticos actualmente creados sólo registran los datos de los profesores y especialistas, pues no incluyen otros trabajadores no docentes; esto provoca que no se posea la

información curricular de todo el personal y que no se brinde la posibilidad de que un experto evalúe las evidencias reflejadas en el currículum.

Por lo antes expuesto, se deriva el siguiente **problema científico:** ¿cómo gestionar el currículum vítae de los trabajadores de la facultad 3 garantizando que cumpla con modelos internacionales?

Se define como **objeto de estudio:** sistemas informáticos para la gestión de información. Enmarcado en el **campo de acción:** sistemas informáticos para la gestión de información curricular.

Para dar solución al problema científico planteado se define como **objetivo general** de la investigación**:** desarrollar un sistema informático que permita gestionar los currículos de los trabajadores de la facultad 3, basados en los modelos internacionales a partir del registro de información personal.

Objetivos específicos:

1. Conformar el marco teórico asociado a sustentar los conceptos y la propuesta de solución.
2. Implementar el sistema informático para la generación del CV de los trabajadores de la Facultad 3.
3. Validar técnicamente la propuesta de solución en la Facultad 3 de la UCI.

Tareas de la investigación:

1. Definir el marco teórico para dejar definidos los principales conceptos asociados al campo de acción y al objetivo de investigación.
2. Analizar la información existente relacionada con los modelos internacionales existentes de CV.
3. Analizar la información existente relacionada con las normas nacionales e internacionales de conformación de los CV.
4. Analizar la información existente relacionada con el proceso de certificación de evidencias.
5. Describir el dominio identificado para el sistema a desarrollar.
6. Diseñar el dominio identificado del sistema informático en cuestión.
7. Integrar las herramientas estudiadas para la construcción del sistema diseñado.
8. Implementar el sistema informático para la gestión del CV.
9. Validar el sistema informático para la gestión del CV a través de la personalización del mismo.
10. Realizar pruebas al sistema desarrollado.

Idea a defender:

Si se implementa un sistema informático para la gestión de currículum vítae basado en modelos internacionales, se podrá garantizar una correcta conformación de currículos de los trabajadores en la Facultad 3.

La investigación será guiada por los siguientes métodos de investigación:

Métodos de Investigación:

Métodos Teóricos

Análisis Histórico-Lógico: este método se utilizará para analizar los antecedentes, la evolución y el desarrollo del CV, los principales conceptos asociados al mismo y los sistemas que existen en la actualidad.

Analítico-Sintético: este método permitirá el análisis de la información acerca de las tecnologías, metodologías y herramientas posibles a ser utilizadas en el desarrollo del sistema, seleccionando los principales elementos y características.

Modelado: realizada la investigación y analizada la información, este método es muy útil para realizar los modelos correspondientes al ciclo de vida del sistema, esto permitirá facilidades a la hora de cumplir con las tareas de análisis y diseño de los procesos que intervendrán en la aplicación, así como para implementar el sistema.

Métodos Empíricos

Entrevista: este método tiene como objetivo obtener información sobre las necesidades reales que tiene la Facultad 3, para que el sistema cumpla de forma completa con las mismas.

Observación: este método se utilizará para observar cómo funciona el proceso de certificación de evidencias y la gestión curricular de los trabajadores en la Facultad 3.

El presente trabajo de diploma está estructurado en 3 capítulos:

-**Capítulo 1. Fundamentación teórica**.

En este capítulo se mostrará el resultado de la investigación enfocada al objeto de estudio y contiene una base teórica para entender el problema a solucionar. Conceptos fundamentales, técnicas,

tendencias, metodologías y herramientas utilizadas, así como los elementos para definir las que se utilizarán en el desarrollo del trabajo. Además se incluye el estado del arte del tema.

-**Capítulo 2. Análisis y diseño del sistema**.

En este capítulo se realiza la descripción del negocio propuesto, se diseña la solución mediante el modelo de dominio y se describen los patrones que se utilizarán en el diseño e implementación del sistema a desarrollar. También se determinan las funcionalidades de la aplicación que serán descritas a través de las historias de usuarios, incluye el plan de iteraciones con la planificación, prioridad y duración de desarrollo de cada funcionalidad. Se describen las tarjetas clase-responsabilidad-colaboración, se muestra el diseño del modelo de datos y además se confecciona el plan de entrega de la aplicación.

-**Capítulo 3. Implementación y prueba**.

En este capítulo se describe la implementación del sistema, para cumplimentar los requisitos especificados por el cliente. Para un mejor entendimiento del sistema se realizará los diagramas de componentes y el diagrama de despliegue pertenecientes a la aplicación. De la misma manera se define los estándares de codificación a utilizar. Por último se realizarán las pruebas unitarias y pruebas de aceptación para validar el software.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Introducción

Este capítulo abarcará todo lo relacionado con el estado del arte de los distintos sistemas de gestión curricular, así como el análisis de las herramientas y tecnologías posibles a usar para desarrollar el sistema. Se realizará el estudio de la metodología a utilizar en el desarrollo del sistema. Se definirán conceptos necesarios como por ejemplo los conceptos de información, currílucum vítae, gestión, gestión de la información, certificación de evidencias, entre otros que permitirán entender y realizar el trabajo.

1.1 Marco teórico, conceptos y definiciones

Currículum vítae

El currículum vítae es una frase del latín; significa carrera de la vida, que por simplificación a nivel mundial se utiliza solo el término currículo. Esto no es más que el conjunto de experiencias, laborales, educacionales y vivenciales de una persona, así como la relación de títulos, honores, cargos, trabajos realizados, datos biográficos, etc., que califican a una persona o individuo. Currículo es la acepción en singular de currículum. (3)

El currículum vítae o CV es el documento en el cual se expone el historial académico y profesional de un candidato a un puesto, de una manera clara, concisa y ordenada. No se trata de un relato de la vida del candidato, ni de una base de datos de sus cualidades, ni un inventario de competencias, es un arma para vender la posibilidad de obtener una entrevista de trabajo ante el seleccionador, que será el verdadero momento de convencer y vencer. (1)

Dado el conjunto de conceptos asociado al currículum vítae, los autores del presente trabajo seleccionaron el definido por Dr. José Antonio Fernández Sánchez. “El currículum vitae o CV es el documento en el cual se expone el historial académico y profesional de un individuo, de una manera clara, concisa y ordenada. No se trata de un relato de la vida del candidato, ni de una base de datos de sus cualidades, ni un inventario de competencias, es un arma para vender la posibilidad de obtener una entrevista de trabajo ante el seleccionador, que será el verdadero momento de convencer y vencer.” (1)

Los modelos de currículum son americanos y europeos. El modelo europeo es más largo y detallado. Tiene varias secciones y los datos personales suelen situarse en el apartado del CV. (2) Mientras que el americano suele ser más corto y conciso.

Cuando se elabora un currículum se deben tener en cuenta algunos aspectos importantes que no pueden faltar dentro de la información del mismo, como son: los datos personales, el historial tanto académico como profesional, cualidades profesionales y personales, por ejemplo los idiomas que domina, el conocimiento de informática y otros datos de interés. Para la elaboración del currículum es necesario tener un orden y limpieza adecuados ya que se está redactando la información personal que será leída por otras personas. (4)

Existen varios tipos de currículum vítae, como son el cronológico, el inverso, mixto y el temático o funcional. El cronológico es cuando la información curricular es ordenada de la más antigua a la más reciente, este método tiene como propiedad principal que resalta la evolución seguida del individuo, la estabilidad y la evolución ascendente de la persona. El inverso es el que permite destacar la experiencia reciente, que en algunas ocasiones y para algunas empresas es la información más importante a considerar. El mixto, es aquel en el que la información se organiza temáticamente aunque se mantiene la distribución cronológica (o inversa) dentro de cada bloque (5).El temático o funcional ordena el currículum en bloques temáticos y tiene como ventaja que al no seguir un orden cronológico se pueden abordar los puntos más importantes de la información curricular de un individuo y omitir aquella información que indique errores de recorrido. (6)

Modelo Americano o currículum vítae en inglés

El americano es aquel que ocupa como máximo dos hojas, pone los datos personales en el encabezado de la página a modo de membrete, y hace una selección antecedente para limitar el espacio y dar al entrevistador sólo lo que necesita para la oferta de empleo específica.

El CV de modelo americano es usado básicamente por candidatos que postulan un puesto en las siguientes áreas:

- ✓ Internacional
- ✓ Académico
- ✓ Educación
- ✓ Científica
- ✓ Investigación
- ✓ Salud
- ✓ Premios

En el Anexo 5: Modelos de currículums se encuentran varios ejemplos de CV correspondiente al modelo americano.

Modelo Europeo

El CV Europeo es un modelo estandarizado que posibilita a los individuos presentar de modo sistemático y cronológico sus cualificaciones y competencias. El CV incluye informaciones sobre datos personales, nivel de estudios y formación, experiencia laboral, competencias lingüísticas y otras competencias adicionales.

En el Anexo 5: Modelos de currículums se encuentran varios ejemplos de CV correspondiente al modelo europeo.

Gestión

Del latín *gestĭo*, el concepto de gestión hace referencia a la acción y a la consecuencia de administrar o gestionar un objeto, una empresa, etc. (3)
Conjunto de trámites que se llevan a cabo para resolver un asunto. Dirección, administración de una empresa, negocio, etc. (7)

Gestión curricular

Los autores del presente trabajo definieron como Gestión curricular: la acción o secuencia de acciones que permiten manipular el documento en el cual se expone el historial académico y profesional de un individuo, de una manera clara, concisa y ordenada.

Información

La información puede entenderse como la significación que adquieren los datos como resultado de un proceso consciente e intencional de cuatro elementos: los datos del entorno, los propósitos y el contexto de aplicación, así como la estructura de conocimiento del sujeto. (8)

Gestión de información

La gestión de información es el proceso que se encarga de suministrar los recursos necesarios para la toma de decisiones, así como para mejorar los procesos, productos y servicios de una organización. La gestión de la información no es un proceso que se pueda llamar nuevo, es algo que preocupó al hombre desde el inicio de los tiempos, desde que el hombre se dio cuenta que aprovechar solamente su experiencia lo limitaba con respecto a aquellos que aprovechaban las experiencias colectivas, y más aún, se trazaban estrategias para poder adquirir más conocimiento. (9)

La gestión de información es todo lo que tiene que ver con obtener la información correcta, en la forma adecuada, para la persona indicada, al costo correcto, en el momento oportuno, en el lugar indicado para tomar la acción precisa. (10)

La coordinación eficiente y eficaz de la información procedente de fuentes internas y externas. (11)

El proceso mediante el cual se obtienen, despliegan o utilizan recursos básicos (económicos, físicos, humanos, materiales) para manejar información dentro y para la sociedad a la que sirve. Tiene como elemento básico la gestión del ciclo de vida de este recurso y ocurre en cualquier organización. Es propia también de unidades especializadas que manejan este recurso en forma intensiva, llamadas unidades de información. (12)

Sistemas de gestión de información

Sistema integrado y automatizado para proveer la información que sostenga las funciones de operatividad, gestión y toma de decisiones en una organización. (13)

El conjunto de políticas y normas relacionadas entre sí que se establecen para el acceso y tratamiento de los recursos de información, incluye los registros administrativos y los archivos, el soporte tecnológico de los recursos y el público a que se destina. En su evolución el sistema puede manejar la función de inteligencia corporativa y generar productos de inteligencia. (14)

Los autores del presente trabajo seleccionaron como definición de sistema de gestión de información el definido por David y Olson (13), el cual plantea: “Sistema integrado y automatizado para proveer la información que sostenga las funciones de operatividad, gestión y toma de decisiones en una organización.”

Certificación de evidencia

La certificación tiene como objetivo reconocer las competencias prácticas, individuales y sociales obtenidas por el trabajador a lo largo de la vida profesional, documentar y señalar las competencias exigidas en la realidad práctica del trabajo. Procedimiento por el cual un tercero verifica públicamente que una persona cumple con los requisitos de una norma, independientemente de la forma de cómo lo haya adquirido. (15)

Dado que el concepto antes mencionado no se ajusta al contexto de la facultad 3, los autores del presente trabajo defininen como certificación de evidencias:

Es el procedimiento por el cual un tercero verifica públicamente las competencias prácticas, individuales y sociales obtenidos por un individuo a lo largo de su vida profesional.

Certificación bibliográfica

Proceso mediante el cual una tercera parte emite una declaración por escrito, basada en una decisión tomada después de la revisión de que se ha demostrado que se cumplen los requisitos especificados para un producto, proceso, servicio o persona conforme con los requisitos especificados para el alcance cubierto por la misma. (16)

1.2 Análisis de las soluciones existentes

1.2.1 Sistemas informáticos para la certificación de evidencia

Publifirma

Publifirma es un sistema para certificar la publicación de un documento en la web en un día y hora exacta, sabiendo el tiempo que ha permanecido publicado y garantizando que no ha sufrido modificaciones durante el periodo de publicación. (17)

Características

- Aporta evidencia legal de la publicación de documentos en una web.
- Proceso de certificación de publicaciones puede ser manual o automático.

Sistema Informático para la Certificación en el Centro de Recursos para el Aprendizaje y la Investigación (CRAI)

Es una aplicación web con el fin de acreditar o validar una publicación, permitiendo mejorar y facilitar el trabajo de los especialistas de la entidad, reduciendo considerablemente el tiempo de procesamiento de los datos que a su vez repercute positivamente en el funcionamiento del centro. La herramienta contribuye a la eliminación de los errores que se comenten a diario, permitiendo gestionar la información del proceso de certificación de manera rápida y confiable. (16)

Características

- Permite gestionar revistas científicas, monografías, artículos, reportes y libros.

1.2.2 Sistemas que dentro de sus funcionalidades permiten gestionar CV

CurriculumQ

Es un sistema de gestión curricular desarrollado para ambiente desktop, es ejecutable solo en Windows, actualmente se encuentra en su versión 5.0, el mismo permite crear currículum de forma fácil y dinámica. Dentro de sus principales características se encuentran:

- Facilita la tarea de tener que realizarlo personalmente.
- Crea tu carta de presentación de forma rápida y profesional.
- El CV que elabora este software es el más aceptado en el mundo laboral.
- Utiliza el formato de currículum de orden cronológico.
- Incluye también un corrector ortográfico.
- Plantillas personalizables, exportarlos a PDF, impresión directa en DIN A4[1] y personalización de textos. (18)

CurrículumFácil

CurrículumFácil es un programa de diseño y redacción de currículos con las siguientes características:

- Propone el tipo de currículo, adecuándose a la situación personal.
- Sugiere frases y expresiones predeterminadas para definir habilidades.
- Ayuda durante el proceso de creación.
- Sugerencia de formas de redacción.
- Sugerencias de modelos de cartas de presentación.
- Incluye más de 25.000 modelos y plantillas de currículos.
- Gestión de envío de currículos y cartas de presentación.
- Aprobado por miles de usuarios a nivel mundial. (19)

NitroCV

NitroCV es una aplicación que brinda ayuda para confeccionar y distribuir currículos. Permite crear un Currículum Vitae, publicarlo en Internet, enviarlo por correo electrónico, utilizar plantillas a la hora de crear el currículo, almacenar un número ilimitado de currículum, entre otras funcionalidades.

NitroCV requiere como primer paso crear un currículum vítae nuevo, como segundo paso se introducen todos los datos basándose en unas pautas que el software tiene definidas. Se selecciona una plantilla y se crea el currículum, siendo este publicado. Se crea una página en Internet con el Currículum Vitae funcionando al instante y con la posibilidad de ser actualizado. Por otro lado también

[1] Es el estándar que define el popular tamaño de papel A4.

brinda la posibilidad de enviar el currículo directamente desde el programa NitroCV (requiere parámetros de configuración del correo electrónico que se utilice). (20)

LinkedIn

LinkedIn es un sitio web orientado a negocios. Fue fundado en diciembre de 2002 y lanzado en mayo de 2003 (comparable a un servicio de red social), principalmente para red profesional. Fue fundada por Reid Hoffman, Allen Blue, Konstantin Guericke, Eric Ly y Jean-Luc Vaillant. (21)

Permite las siguientes funcionalidades:

- Una red de contactos se construye mediante conexiones directas, las conexiones de cada una de estas conexiones (denominado conexiones de segundo grado) y también las conexiones de conexiones de segundo grado (denominadas conexiones de tercer grado). Esto se puede utilizar para introducirse en la red a través de un contacto mutuo, favoreciendo la interactividad.
- Los usuarios pueden subir su currículum vítae o diseñar su propio perfil con el fin de mostrar experiencias de trabajo y habilidades profesionales.
- Se puede utilizar para encontrar puestos de trabajo y oportunidades de negocio recomendados por alguien de la red de contactos.
- Los empleadores pueden enumerar puestos de trabajo y la búsqueda de posibles candidatos.
- Los solicitantes de empleo pueden revisar el perfil de los directores de recursos humanos y descubrir cuáles de sus contactos existentes pueden presentarse.
- Los usuarios pueden subir sus propias fotos y ver las fotos de los demás para ayudar a identificarlo.
- Los usuarios pueden seguir diferentes empresas y pueden recibir una notificación acerca de las posibles fusiones y ofertas disponibles.
- Los usuarios pueden guardar trabajos que les gustaría solicitar.

ResearchGate

Es una plataforma de investigación y colaboración en línea, totalmente gratuita, y está diseñada para la cooperación y el intercambio de conocimientos científicos en el mundo. Dirigida a estudiantes, profesores e investigadores de cualquier ámbito.

Actualmente hay más de un millón de miembros activos en ella, que utilizan la plataforma para comunicar y dejar comentarios, compartir y discutir cuestiones científicas, participar activamente en los congresos y acontecimientos científicos así como encontrar nuevas oportunidades de empleo

y programas de postgrado. Asimismo es una herramienta de ayuda a los investigadores para la publicación de sus investigaciones y para potenciar su difusión en todo el mundo. (22)

Pone a disposición del investigador diferentes aplicaciones para impulsar de manera efectiva el desarrollo de la cooperación y el intercambio de conocimientos:

- Perfil científico: los investigadores que deseen formar parte de esta red social han de crear un perfil personal, semejante a un CV de investigación, que reúne los intereses del investigador, educación, proyectos, experiencias, datos de contacto, un índice de publicación personal, opción de presentar y compartir documentos, blog personal, posibilidad de exportar a su software de gestión de referencias.
- Grupos de discusión: los investigadores pueden crear grupos con el fin de discutir con personas de ideas afines y trabajar en colaboración en un entorno interactivo.
- Publicaciones: incluye un motor de búsqueda que permite al usuario buscar información simultáneamente a través de siete bases de datos principales y más de 1.000 bases de datos de acceso libre, incluyendo Pubmed, ArXiv, IEEE y CiteSeer entre otros.
- Publicaciones Autoarchivo: los investigadores pueden subir directamente en su perfil personal sus artículos publicados en revistas. Compara automáticamente el índice de revistas con los acuerdos de autoarchivo de revistas y editoriales de SHERPA ROMEO, de forma que los autores pueden saber que versiones de sus artículos pueden subir legalmente a la red.

Sistema de gestión universitaria

Sistema informático desarrollado por la universidad de las ciencias informáticas y puesto a disposición el 9 de febrero del 2015, actualmente se encuentra en su versión 1.0.0.2. Es un sistema en el cual se expone el recorrido académico de los docentes y el desempeño laboral de los profesionales de dicha institución.

Después de haber estudiado los diferentes sistemas de gestión curricular antes expuestos, se conformó una tabla comparativa de los mismos. Para ello se tomaron en cuenta los siguientes aspectos: Sistema Operativo (SO) que soporta, disponibilidad, accesibilidad y si permiten certificar evidencias.

Sistemas	SO que soportan	Disponibilidad	Accesibilidad	Certifican evidencias

CurriculumQ	Windows	La aplicación, es un instalable local que provee al usuario de las herramientas necesarias para la confección del CV.	Puede ser accedido solo por el usuario local de la PC donde se encuentre instalada la aplicación, en el momento que este lo requiera.	No
CurrículumFácil	Windows	La aplicación, es un instalable local que provee al usuario de las herramientas necesarias para la confección del CV.	Puede ser accedido solo por el usuario local de la PC donde se encuentre instalada la aplicación, en el momento que este lo requiera.	No
NitroCV	Windows	La aplicación, es un instalable local que provee al usuario de las herramientas necesarias para la confección del CV.	Puede ser accedido solo por el usuario local de la PC donde se encuentre instalada la aplicación, en el momento que este lo requiera.	No
LinkedIn	Linux, Windows, Mac OS	Disponible para ser accedido las 24 horas del día.	Es accesible para todo usuario que cuente con conexión a internet.	Parcialmente
ResearchGate	Linux, Windows, Mac OS	Disponible para ser accedido las 24 horas del día.	Es accesible para todo usuario que cuente con conexión a internet.	Parcialmente

Sistema de gestión universitaria	Linux, Windows, Mac OS	Disponible para ser accedido las 24 horas del día.	Es accesible para todos los usuarios pertenecientes a la Universidad de Ciencias Informáticas.	Parcialmente

Tabla 1. Comparativa entre los sistemas que dentro de sus funcionalidades permiten la gestión curricular

Como resultado del análisis de la tabla anterior, se pudo arribar a la conclusión que aunque los sistemas cumplen con los parámetros y estándares de redacción del CV; la mayoría no son ejecutables en los diferentes sistemas operativos, necesitan de conexión a internet para su publicación o redacción, no poseen un certificador de evidencias que confirme los resultados alcanzados expuestos por los propietarios de los currículum. Por tales motivos se propone un sistema que permita la centralización de la información de los trabajadores de la Facultad 3, así como la creación y publicación de sus CV. Además se certificará la evidencia presentada en el currículo de cada individuo.

1.3 Metodologías de desarrollo de software

Un proceso de software detallado y completo suele denominarse metodología de desarrollo de software[2]. Las metodologías se basan en una combinación de los modelos de proceso genéricos (cascada, evolutivo, incremental, etc.). Adicionalmente una metodología debería definir con precisión los artefactos, roles y actividades involucrados, junto con prácticas y técnicas recomendadas, guías de adaptación de la metodología al proyecto, guías para uso de herramientas de apoyo, etc. (23)

Extreme Programming (XP)

La Programación Extrema es una metodología ligera de desarrollo de software formulada por Kent Beck, autor del primer libro sobre el tema, *Extreme Programming Explained: Embrace Change* (1999). Se basa en la simplicidad, la comunicación y la realimentación continua entre el cliente y el equipo de desarrollo. Promueve el trabajo en equipo preocupándose por el aprendizaje de sus colaboradores.

[2] Una metodología de desarrollo de software es un conjunto de procedimientos, técnicas y actividades a seguir para en un fin convertir los requisitos de usuario en un producto de software.

Está definida y especialmente adecuada para proyectos con requisitos imprecisos y cambiantes, y donde existe alto riesgo técnico. (24)

Características

- Es capaz de adaptarse a los cambios de requisitos.
- Colaboración continua entre, el cliente y el equipo de desarrollo.
- Está enfocado hacia equipos de trabajos pequeños y medianos.
- El equipo de desarrollo sigue estrictamente el orden de prioridad de las tareas definidas por el cliente.
- Los miembros del equipo trabajan en parejas en un proyecto XP.

Según Kent Beck, XP precisa de cuatro fases de desarrollo las cuales son (ver Anexo 4: Fase de desarrollo de la metodología XP):

1ra Fase: Planificación del proyecto.

Es la fase en la que se define el alcance general del proyecto. El cliente define lo que necesita mediante la redacción de sencillas “historias de usuarios”. Los programadores estiman los tiempos de desarrollo en base a esta información. Las estimaciones realizadas son primarias (ya que estarán basadas en datos de muy alto nivel), y podrían variar cuando se analicen en detalle en cada iteración. Obteniendo como resultado una visión general del sistema y un Plan de Entregas.

2da Fase: Diseño.

En esta fase la metodología sugiere que hay que conseguir diseños simples, fácilmente entendibles e implementables. Utilizar glosarios de términos para una correcta especificación de los nombres y métodos de cada una de las clases.

3ra Fase: Codificación

Es la fase principal en el ciclo de desarrollo de XP. Las funcionalidades son desarrolladas, generando al final de cada una un entregable funcional que implementa las historias de usuario asignadas a la iteración. El cliente, también debe participar activamente durante este período del ciclo.

4ta Fase: Pruebas.

Al final de cada iteración se entregan módulos funcionales y sin errores, puede ser deseable por parte del cliente no poner el sistema en producción hasta tanto no se tenga la funcionalidad completa. En esta fase no se realizan más desarrollos funcionales, pero pueden ser necesarias tareas de ajuste, dentro de las que se encuentran la realización de pruebas unitarias y pruebas de aceptación. (25)

Después de haber analizado algunas de las metodologías de desarrollo, se determinó que dado el tiempo, las características y la forma de interacción entre los distintos roles que conforman la metodología se seleccionó Extreme Programming (XP).

1.4 Herramientas, tecnologías y lenguajes a utilizar en la solución

Para desarrollar el sistema se definen las herramientas, tecnologías y lenguajes a utilizar. Con este objetivo, se hace necesario realizar un estudio de las mismas analizando sus características, ventajas y desventajas, así como las tendencias actuales en el desarrollo de sistemas para definir cuáles serán las que se utilizarán para desarrollar el sistema. Se analizarán las herramientas y tecnologías de desarrollo que sean libres o permitan implementaciones sobre software libre.

1.4.1 Herramientas CASE

Visual Paradigm 8.0

Visual Paradigm es una herramienta para desarrollo de aplicaciones utilizando modelado UML ideal para Ingenieros de Software, ha sido concebida para soportar el ciclo de vida completo del proceso de desarrollo de software mediante presentación de diagramas. La misma está diseñada para ayudar al desarrollo ágil, lo que reduce en gran medida los costes de desarrollo y mejorar la calidad del producto final. Es una herramienta colaborativa, pues soporta múltiples usuarios trabajando sobre el mismo proyecto. (26)

Ventajas

- Disponible en múltiples plataformas.
- Modelo y código permanece sincronizado en todo el ciclo de desarrollo.

Esta herramienta permite aumentar la calidad del software, portabilidad y estandarización de la documentación. Se decidió utilizar Visual Paradigm 8.0 facilitando la modelación de los artefactos.

1.4.2 Lenguaje de modelado

Lenguaje Unificado de Modelado 2.1 (UML)

Este lenguaje prescribe un conjunto de notaciones y diagramas estándares para modelar sistemas orientados a objetos, y describe la semántica esencial de lo que estos diagramas y símbolos significan; posibilitando así visualizar, especificar y documentar los artefactos o toda información que se obtiene o modifica durante un proceso de desarrollo de software. Además de poder utilizarse para modelar distintos tipos de sistemas de software, hardware y organizaciones del mundo real. (27)

Características

- Posibilita mejores tiempos totales de desarrollo.
- Tecnología orientada a objetos.
- Mejor soporte a la planeación y al control de proyectos.

1.4.3 Lenguajes de programación

PHP 5.6

PHP es un lenguaje interpretado de propósito general ampliamente usado, diseñado especialmente para desarrollo web y que puede ser incrustado dentro de código HTML5[3]. Generalmente se ejecuta en un servidor web, tomando el código en PHP como su entrada y creando páginas web como salida. Puede ser desplegado en la mayoría de los servidores web y en casi todos los sistemas operativos y plataformas sin costo alguno. (28)

Ventajas

- Permite involucrarse con aplicaciones de contenido dinámico sin tener que aprender todo un nuevo grupo de funciones.
- PHP permite la conexión a diferentes tipos de servidores de bases de datos dentro del cual se encuentra PosgreSQL.
- PHP también tiene la capacidad de ser ejecutado en la mayoría de los sistemas operativos, tales como UNIX y Windows.

1.4.4 Marco de Trabajo (Framework)

Es un conjunto de funciones, procedimientos y herramientas destinadas a la construcción de un determinado tipo de aplicaciones, simplificando el desarrollo de las mismas mediante la automatización de algunos de los patrones utilizados para resolver tareas comunes. Es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Los frameworks hacen el trabajo más fácil, presentando funcionalidades y componentes reutilizables que brindan mayor eficiencia a la hora de desarrollar una aplicación**.** (29)

Symfony 2.3.28

Es un completo *framework* diseñado para optimizar el desarrollo de las aplicaciones web, emplea el tradicional patrón de diseño MVC (modelo-vista-controlador) el cual separa la lógica del negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases

[3] Lenguaje de marcado de hipertexto, para la elaboración de páginas web. http://www.w3c.es/Divulgacion/html/logo/

encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. (30)

Ventajas

- Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.
- Preparado para aplicaciones empresariales y adaptables a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo y corto plazo.

Bootstrap 3.1

Bootstrap es el *framework* de Twitter[4] que permite crear interfaces web con CSS 3[5] avanzado y JavaScript que adaptan la interfaz dependiendo del tamaño del dispositivo en el que se visualice la información, es decir, automáticamente se adapta al tamaño de la pantalla del dispositivo sin que el usuario tenga que hacer nada, esto se denomina diseño adaptativo o diseño responsable. (31)

Ventajas

- Agilidad a la hora de cargar la interfaz y al adaptarse a los diferentes tipos de dispositivos.
- Posee varios elementos con estilos predefinidos fáciles de configurar.
- Compatible con la mayoría de los navegadores web.

1.4.5 Sistemas Gestores de Base de Datos (SGBD)

Los sistemas gestores de bases de datos son un tipo de software muy específico que sirve de interfaz entre el usuario, las aplicaciones y la base de datos. Están compuestos por un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta. Su propósito general es manejar de forma sencilla y ordenada los datos que posteriormente se convertirán en información relevante para el manejo correcto de información. Los objetivos fundamentales de los SGBD son: independencia de los datos y los programas de aplicación, minimización de la redundancia, integración y sincronización de las bases de datos, integridad de los datos, seguridad y protección de los datos, facilidad de manipulación de la información y control centralizado. (32)

[4] Es un framework o conjunto de herramientas de software libre para diseño de sitios y aplicaciones web.

[5] Lenguaje usado para definir la presentación de un documento estructurado escrito en HTML o XML.

PostgreSQL 9.3

Es un sistema de gestión de base de datos relacional orientada a objetos y libre, publicado bajo la licencia BSD. Como muchos otros proyectos de código abierto, el desarrollo de PostgreSQL no es manejado por una sola empresa sino que es dirigido por una comunidad de desarrolladores y organizaciones comerciales las cuales trabajan en su desarrollo. Se ejecuta en la mayoría de los sistemas operativos más utilizados en el mundo como Linux, varias versiones de UNIX y Windows. (33)

Características

- Multiplataforma.
- Extensible.
- Estabilidad y Confiabilidad.
- Instalación ilimitada.

1.4.6 Herramienta para administrar el gestor de base de datos

pgAdmin3

Es una aplicación de diseño y manejo de bases de datos para su uso con PostgreSQL. Este software fue diseñado para responder a las necesidades de todos los usuarios, desde la escritura de simples consultas SQL a la elaboración de bases de datos complejas. (34)

Responde a las necesidades de la mayoría de los usuarios, desde escribir simples consultas SQL hasta desarrollar bases de datos complejas. La interfaz gráfica soporta todas las características de PostgreSQL y hace simple la administración. (35)

1.4.7 Entorno de desarrollo (IDE)

Netbeans 8.0

NetBeans IDE es un entorno de desarrollo, una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender el NetBeans IDE. Es un producto libre y gratuito sin restricciones de uso. Permite desarrollar aplicaciones web de forma fácil y rápida, así como aplicaciones HTML5 con HTML, JavaScript y CSS. El IDE también proporciona un gran conjunto de herramientas para desarrolladores de PHP y C / C ++. El mismo tiene una gran integración con el *framework* Symfony2. (36)

1.4.8 Servidor de Aplicaciones

Apache 2.2.22-13

Apache 2.2 es un servidor web de software libre aunque ha sido ampliado su uso a los diferentes sistemas operativos. Desarrollado por la Apache *Software Foundation* (Fundación de software Apache) cuyo objetivo es servir o suministrar páginas web (en general, hipertextos) a los clientes web o navegadores que las solicitan. (37)

Características

- Facilidad de configuración.
- Código abierto y altamente configurable.
- Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor y es posible configurarlo para que ejecute un determinado script cuando esto suceda.

1.4.9 Herramienta para el control de versiones

Git

Es un sistema gratuito y de código abierto distribuido de control de versiones diseñado para manejar todo, desde pequeños a grandes proyectos con rapidez y eficiencia. Diseñado por Linus Torvalds[6], pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de código fuente. El mismo tiene como objetivo controlar los cambios en el desarrollo de cualquier tipo de software, permitiendo conocer el estado actual de un proyecto, los cambios que se le han realizado a cualquiera de sus piezas y las personas que intervinieron en las modificaciones del software. (38).

Para llevar a cabo un control de versiones de sistema a desarrollar se utilizará Git[7].

1.4.10 Herramienta para las pruebas

PHPUnit

PHPUnit es una herramienta que permite realizar las pruebas pertinentes al código de las aplicaciones desarrolladas en el lenguaje PHP, verificando que el funcionamiento de las mismas es el

[6] Linus Benedict Torvalds es un ingeniero de software finlandés estadounidense, conocido por iniciar y mantener el desarrollo del "kernel" Linux, basándose en el sistema operativo libre Minix creado por Andrew S.

[7] Es una aplicación para administrar proyectos con el control de versiones.

deseado y en su caso encontrando bugs y errores que una vez solucionados mejorarán la calidad del desarrollo web. Este software basado en la familia de *frameworks* xUnit[8] constituye junto con alternativas como SimpleTest o phpt[9], los principales *frameworks* de pruebas unitarias en PHP. (39)

Conclusiones parciales

Luego del estudio realizado de los principales conceptos y el marco teórico asociados a la investigación se concluye que:

- Los sistemas para la gestión curricular analizados no permiten su adaptación a las especificaciones para la gestión de los CV de la Facultad 3.
- Las herramientas, tecnologías y metodología propuestas son suficientes para el desarrollo del sistema a desarrollar, atendiendo a la política de la universidad sobre la migración a software libre.

[8] Entorno de pruebas, basado en el diseño de Kent Beck.

[9] El sistema de prueba SimpleTest PHP es un marco de prueba de unidad PHP y prueba de web. (http://www.simpletest.org/)

CAPÍTULO 2: ANÁLISIS Y DISEÑO DEL SISTEMA

INTRODUCCIÓN

En este capítulo se propone la solución a la problemática presentada y se realiza el modelado de la misma a través del modelo de dominio. Se confeccionan las historias de usuarios donde se describen las funcionalidades a desarrollar otorgando la prioridad a cada una. A partir de las mismas se generan las iteraciones planificadas para realizar la aplicación y se diseña el plan de entregas que permite analizar el tiempo de concepción del sistema. Además se muestra la vista lógica de la arquitectura, así como los patrones de diseño a utilizar en la solución.

2.1 Propuesta del sistema

El sistema propuesto debe manejar la información curricular de los trabajadores de la facultad 3, permitiendo al trabajador crear su CV y actualizarlo en función o no a la presentación de evidencias. La evidencia reflejada en el CV podrá ser validada por un usuario responsable de esta tarea. El sistema debe garantizar que la información esté disponible para que el usuario propietario o el certificador de evidencia puedan hacer uso de la misma cuando la requiera ya sea para su certificación o modificación, deberá garantizar además la confidencialidad de dicha información para que solo pueda ser manipulada por las personas con el privilegio para hacerlo. La seguridad de la información es otro de los aspectos importantes a garantizar, puesto que esta puede servir para tomar decisiones importantes no solo en la facultad sino también en la universidad; debe además permitir generar reportes con la información guardada en el CV, así como la búsqueda de trabajadores registrados en el sistema.

2.2 Fase de planificación del proyecto

Es la fase en la que se define el alcance general del proyecto. El equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. El cliente define lo que necesita mediante la redacción de sencillas “historias de usuarios”. Los programadores estiman los tiempos de desarrollo en base a esta información. Las estimaciones realizadas son primarias (ya que estarán basadas en datos de muy alto nivel), y podrían variar cuando se analicen en detalle en cada iteración. Obteniendo como resultado una visión general del sistema y un Plan de Entregas. (25)

2.2.1 Modelo de dominio

Un modelo de dominio es un artefacto construido con las reglas de UML durante la fase de concepción. El modelo de dominio puede ser tomado como el punto de partida para el diseño del sistema. Es una representación visual de clases conceptuales o de objetos reales en un dominio de interés. (40)

2.2.2 Diagrama del modelo de dominio

Al no estar bien definido el proceso de negocio con respecto al currículo de los trabajadores de la Facultad 3, se lleva a cabo el modelo conceptual o modelo de dominio con los principales conceptos y eventos que suceden en el entorno en el que se trabaja.

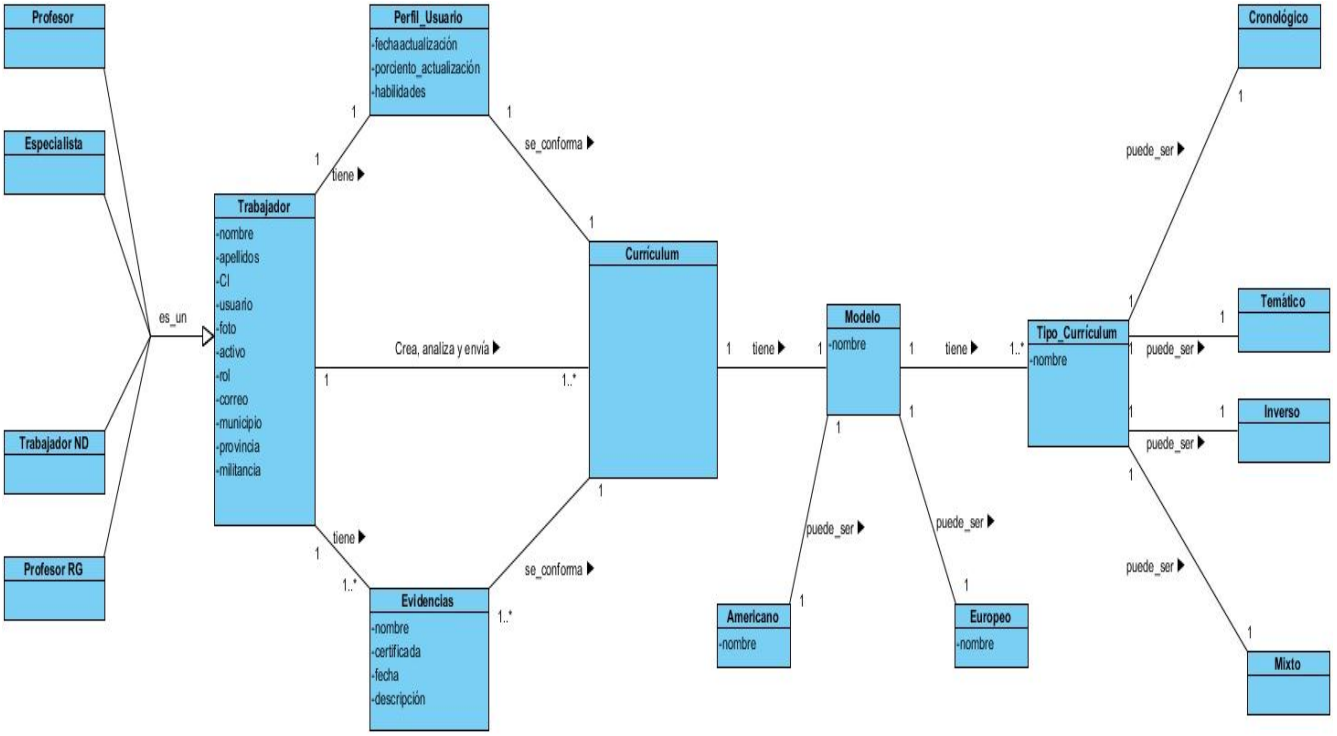

Figura 1. Diagrama del modelo de dominio

2.2.3 Glosario de términos del modelo de dominio

Currículum: contiene la información curricular del trabajador registrado en el sistema.

Profesor: es un profesor graduado no necesariamente de la UCI, y que tiene una hoja de vida desarrollada antes de entrar a la UCI o después.

Profesor RG: es el profesor recién egresado de la universidad, que cuando comienza su trabajo, su currículo se encuentra en blanco.

Trabajador ND: persona que no se encuentra vinculada ni a la docencia, ni a proyectos productivos.

Especialista: persona que se encuentra vinculada a la producción y no a la docencia.

Certificador Evidencias: persona o grupo de personas encargada de verificar las evidencias reflejadas en un CV.

Tipo Currículum: forma en la que se confecciona un CV.

Cronológico: organiza la información cronológicamente, partiendo de los logros más antiguos y llegando hasta los más recientes. (41)

Temático: permite proporcionar un conocimiento rápido de tu formación y experiencia en un ámbito determinado. (41)

Inverso: comienza reflejando los datos más recientes, la última titulación y las últimas experiencias profesionales. Este es el modelo de currículum vitae que prefieren las empresas cuando buscan experiencia profesional. (42)

Mixto: es el más completo de los cuatro tipos de currículum, aunque, por ello, también es el más complejo de elaborar. El currículum combinado es una mezcla del cronológico y del temático. Parte siempre del modelo funcional, organizando la información por áreas temáticas o profesionales, para llegar después a la organización en el tiempo. (41)

Modelo: modelos o estándares para confeccionar un CV internacionalmente.

Americano: modelo o plantilla para conformar un CV**.**

Europeo: modelo o plantilla para conformar un CV**.**

2.2.4 Actores que interactúan con el sistema

Usuarios	Descripción
Trabajadores	Personas que van a interactuar con el sistema (profesor, profesor rg, especialista, trabajador nd).
Administrador	Puede realizar todas las funcionalidades definidas para el sistema. Administra elementos propios del sistema y los permisos de los usuarios.
Certificador	Persona encargada de certificar las evidencias reflejadas en el currículum de los usuarios.

Figura 2. Actores del sistema

2.2.5 Historias de usuario (HU)

En el contexto de una metodología ágil, popularmente se asocia el concepto “historia de usuario” con el de “requisito funcional”. Una historia de usuario (HU) describe funcionalidad que será útil para el usuario, o cliente, de un sistema software. Una historia de usuario debería ser pequeña, memorizable, y que pudiera ser desarrollada por una pareja de programadores en una semana. Pero claro, al tener una única frase, es imposible que una historia de usuario contenga toda la información necesaria para desarrollar. En tan reducido espacio no pueden contener el diseño, las pruebas, normativa, detalles para codificar. (43)

2.2.5.1 Clasificación de las HU

La prioridad en el negocio:

Alta: se le otorga a las HU que resultan fundamentales en el desarrollo y control integral del sistema.

Media: se le otorga a las HU que resultan para el cliente funcionalidades a tener en cuenta, sin que estas tengan una afectación sobre el sistema que se esté desarrollando.

Baja: se le otorga a las HU que sirven de ayuda al control de elementos asociados al desarrollo del sistema.

El riesgo en su desarrollo:

Alta: cuando en la implementación de las HU se consideran la posible existencia de errores que lleven la inoperatividad del código.

Media: cuando en la implementación de la HU pueden aparecer errores que puedan retrasar la entrega del sistema.

Baja: cuando pueden aparecer errores que serán tratados con relativa facilidad sin que traigan perjuicios para el desarrollo del sistema.

Las HU serán representadas mediante tablas divididas por las siguientes secciones:

Número: número de la historia de usuario incremental en el tiempo.

Nombre de la historia de usuario: el nombre de la historia de usuario a desarrollar.

Usuario: involucrados en la funcionalidad de la HU.

Iteración asignada: número de la iteración.

Prioridad en negocio: Alta, Media o Baja.

Riesgo en desarrollo: Alta, Media o Baja.

Estimación técnica: tiempo estimado que se demorará el desarrollo de la HU.

Descripción: breve descripción de la HU.

Durante todo el proceso se identificaron 81 HU, a continuación se muestran las principales historias de usuario del sistema desarrollado, la descripción de las restantes se encuentran en los anexos (ver Anexo 1: historias de usuario).

HISTORIA DE USUARIO	
Número: HU-01	**Usuario:** trabajadores
Nombre de Historia: Confeccionar perfil de usuario	
Prioridad: alta	**Riesgo en desarrollo:** alta
Estimación Técnica: 3	**Iteración asignada:** 3
Descripción: una vez el usuario registrado en el sistema, se adiciona los datos referentes al perfil del usuario.	

Tabla 2. Historia de usuario Confeccionar perfil de usuario

HISTORIA DE USUARIO	
Número: HU-04	**Usuario:** trabajadores
Nombre de Historia: Confeccionar CV	
Prioridad: alta	**Riesgo en desarrollo:** alta
Estimación Técnica: 3	**Iteración asignada:** 3
Descripción: una vez autenticado en el sistema el usuario podrá introducir sus datos generales para la confección de su currículum.	

Tabla 3.Historia de usuario Confeccionar CV

HISTORIA DE USUARIO	
Número: HU-08	**Usuario:** certificador
Nombre de Historia: Certificar evidencia	
Prioridad: alta	**Riesgo en desarrollo:** alta
Estimación Técnica: 3	**Iteración asignada:** 3
Descripción: el certificador podrá certificar las evidencias reflejadas en cada currículum de los trabajadores.	

Tabla 4. Historia de usuario Certificar evidencia

2.2.6 Estimación de esfuerzo por HU

El método de puntos de historia se desarrolló no para obtener un valor en horas de esfuerzo para el desarrollo de una historia de usuario (cada una de las partes en que se divide la funcionalidad a desarrollar en las metodologías ágiles) sino como una manera de dimensionar y relacionar la complejidad de las historias de usuario con respecto a otras. (44)

Para el desarrollo de la aplicación se realizó una estimación de cada una de las HU identificadas en el proceso, la misma se encuentra en los anexos (ver Anexo 2: tabla de estimación de esfuerzo por historia de usuario).

2.2.7 Plan de iteraciones

Después de ser descritas y estimadas las HU se procede a la planificación de la implementación del sistema, quedando definida la implementación en 3 iteraciones: en la primera iteración 29 HU, en la segunda iteración 25 HU y en la tercera iteración 27 HU, las cuales se describen a continuación:

1era iteración:

En esta iteración se implementan las HU de prioridad baja para el sistema, creando así la base para la implementación de las principales funcionalidades. Estas hacen referencia a los diferentes nomencladores que ayudarán al desarrollo de los elementos asociados a la conformación del perfil de usuario.

2da iteración:

En esta iteración se implementan las restantes HU de prioridad baja y las HU de prioridad media del sistema, permitiendo culminar las funcionalidades básicas y desarrollar algunas de las funcionalidades más importantes del sistema. Las mismas hacen alusión a los diferentes formularios donde se insertará información relevante para la creación de un perfil de usuario y donde se iniciará la conformación el CV. Al finalizar se tendrá una versión simplificada de la aplicación.

3era iteración:

En esta iteración se implementan las HU de prioridad alta para el sistema, al finalizarla se tendrán implementadas las peticiones descritas por el cliente y se contará con una versión final del sistema. A partir de ese momento será puesto a prueba durante un período de tiempo para evaluar el desempeño del mismo.

2.2.7.1 Plan de duración de las iteraciones

Iteraciones	Historias de usuario	Estimación
1era iteración	Adicionar revista Modificar revista Eliminar revista Seleccionar idioma Adicionar idioma Cambiar idioma Seleccionar proyecto Adicionar proyecto Modificar proyecto Eliminar proyecto Seleccionar categoría Adicionar categoría Seleccionar departamento Adicionar departamento Modificar departamento Eliminar departamento Seleccionar línea de investigación Adicionar línea de investigación Modificar línea de investigación Eliminar línea de investigación Seleccionar ocupación Adicionar ocupación Modificar ocupación Eliminar ocupación Adicionar graduación Modificar graduación Adicionar publicación Modificar publicación Eliminar publicación	4 semanas

2da iteración	Adicionar artículo Modificar artículo Eliminar artículo Adicionar evento Modificar evento Eliminar evento Adicionar libro Modificar libro Eliminar libro Adicionar premios Modificar premios Eliminar premios Adicionar título académico Modificar título académico Adicionar tesis tutoradas Modificar tesis tutoradas Eliminar tesis tutoradas Adicionar tesis realizadas Modificar tesis realizadas Eliminar tesis realizadas Adicionar registro Modificar registro Eliminar registro Seleccionar modelo de currículo Seleccionar tipo currículo	5 semanas
3era iteración	Confeccionar perfil de usuario Modificar perfil de usuario Buscar usuario Confeccionar CV Modificar CV Eliminar CV Visualizar CV Certificar evidencia Adicionar evidencia Modificar evidencia Eliminar evidencia Autenticación	4 semanas

	Permitir el acceso a los usuarios autorizados Denegar el acceso a los usuarios Asignar rol Adicionar rol Modificar rol Eliminar rol Generar currículo en formato PDF Adicionar diplomado Modificar diplomado Eliminar diplomado Adicionar asignatura Modificar asignatura Eliminar asignatura Adicionar reconocimiento Modificar reconocimiento Eliminar reconocimiento	

Figura 3. Plan de duración de las iteraciones

2.2.8 Requisitos no funcionales

Los requisitos no funcionales son restricciones de los servicios o funciones ofrecidas por el sistema. Incluyen restricciones de tiempo sobre el proceso de desarrollo y estándares. (45)

Los siguientes requisitos no funcionales son importantes para el correcto funcionamiento del sistema propuesto:

- **Disponibilidad**

RNF-1: el sistema debe estar disponible para todos los usuarios al menos durante toda la jornada laboral.

- **Seguridad**

RNF-3: el sistema brinda control de acceso, garantizando que solo los profesionales que radican en la universidad sean capaces de acceder al mismo, así como permitir las funcionalidades permitidas a cada usuario según su rol dentro de la aplicación.

RNF-4: el sistema deberá garantizar la protección ante acciones no autorizadas como las inyecciones SQL.

- **Portabilidad**

RNF-5: el sistema podrá ejecutarse desde los sistemas operativos Linux y Windows.

- **Tipo de aplicación informática**

NF-6: el producto constituye una aplicación web enfocada en las necesidades del sistema de gestión de información curricular de la facultad 3.

- **Finalidad:**

RNF-7: este sistema está enfocado a la gestión de información relacionada con el perfil del usuario y las evidencias reflejadas en el currículum de los trabajadores de la facultad 3.

- **Ambiente:**

RNF-8: el sistema se desarrollará con tecnología PHP versión 5.6.

RNF-9: se empleará como Gestor de Base de Datos PostgreSQL en su versión 9.3

RNF-10: se empleará como Servidor de Aplicaciones Web Apache en su versión 2.2.

RNF-11: las computadoras de los clientes solo requerirán de un navegador web.

- **Usabilidad:**

RNF-12: en el sistema no existirán más de 3 interfaces para lograr una funcionalidad completa.

RNF-13: la tipografía y colores serán estándares en toda la aplicación.

- **Hardware**

RNF-15: se requiere para el servidor una PC con requisitos mínimos, procesador Dual-Core a 2.7GHz, con una memoria RAM de 1GB o superior, y 250 GB o más de espacio en el disco duro.

RNF-16: se requiere para la PC cliente como mínimo, un procesador Pentium 3 a 800MHz, con una memoria RAM de 256 MB o superior, y 40 GB o más de espacio en el disco duro.

- **Software**

RNF-17: se requieren las librerías de php5 como son ldap, mcrypt y curl para poder ejecutar la aplicación en el servidor.

2.3 Fase de Diseño

En esta fase la metodología XP establece prácticas especializadas que inciden directamente en la realización del diseño para lograr un sistema robusto y reutilizable tratando de mantener su simplicidad, es decir, crear un diseño evolutivo que se va mejorando incrementalmente y que permite hacer entregas pequeñas y frecuentes de valor para el cliente. A la hora de darle cumplimiento a la actividad de diseñar, XP no especifica ninguna técnica de modelado. Pueden utilizarse indistintamente sencillos esquemas en una pizarra, diagramas de clases utilizando UML o tarjetas CRC (Clase, Responsabilidad y Colaboración) siempre que sean útiles, tributen a la comprensión y no requieran mucho tiempo en su creación. Utilizar glosarios de términos para una correcta especificación de los nombres y métodos de cada una de las clases. (25)

2.3.1 Tarjetas Clase – Responsabilidad – Colaborador

Las tarjetas CRC tienen como principal funcionalidad, convertir las clases del sistema en objeto y así cada tarjeta representa una clase. El nombre de la clase se coloca en la parte superior, en la sección inferior izquierda las funcionalidades de la misma (responsabilidades) y a la derecha las clases que le sirven de soporte (colaboradoras). (46)

A continuación se muestran las principales trajeras CRC identificadas en el sistema, las demás se describen en los anexos (ver Anexo 3: Tarjetas Clase – Responsabilidad - Colaborador).

Clase: PerfilUsuario	
Responsabilidad	**Colaboración**
Confeccionar perfil de usuario	Usuario
Modificar perfil de usuario	Municipio
Eliminar Usuario	Provincia
Buscar Usuario	Cargo
Generar currículo en formato PDF	Departamento
	Ocupación
	Militancia
	Evidencia
	TítuloAcadémico
	LíneaInvestigación
	Idioma
	Publicación

Tabla 5. Tarjeta CRC PerfilUsuario

Clase: Currículum	
Responsabilidad	**Colaboración**
Confeccionar CV	Perfil
Modificar CV	Modelo
Eliminar CV	TipoCurrículum
Visualizar CV	

Tabla 6. Tarjeta CRC Currículum

Clase: Evidencia	
Responsabilidad	**Colaboración**
Adicionar evidencia Modificar evidencia Eliminar evidencia	Publicación Registro TesisTutoradas Evento Premio Curso

Tabla 7. Tarjeta CRC Evidencia

2.3.2 Modelo de Datos Relacional (Diagrama Entidad – Relación)

Es la representación abstracta de los datos en un sistema gestor de base de datos. Básicamente el modelo de datos está formado por tres elementos fundamentales que son:

- Objetos
- Atributos
- Relaciones

El modelo de datos relacional del sistema consta de un total de 37 entidades para el almacenamiento de toda la información. Las entidades de color azul son las encargadas recoger la información correspondiente al negocio, las de color rosado son los nomencladores del sistema, las naranjas son las relaciones de mucho a mucho que se forman entre diferentes entidades y la de color verde es la encargada de la seguridad.

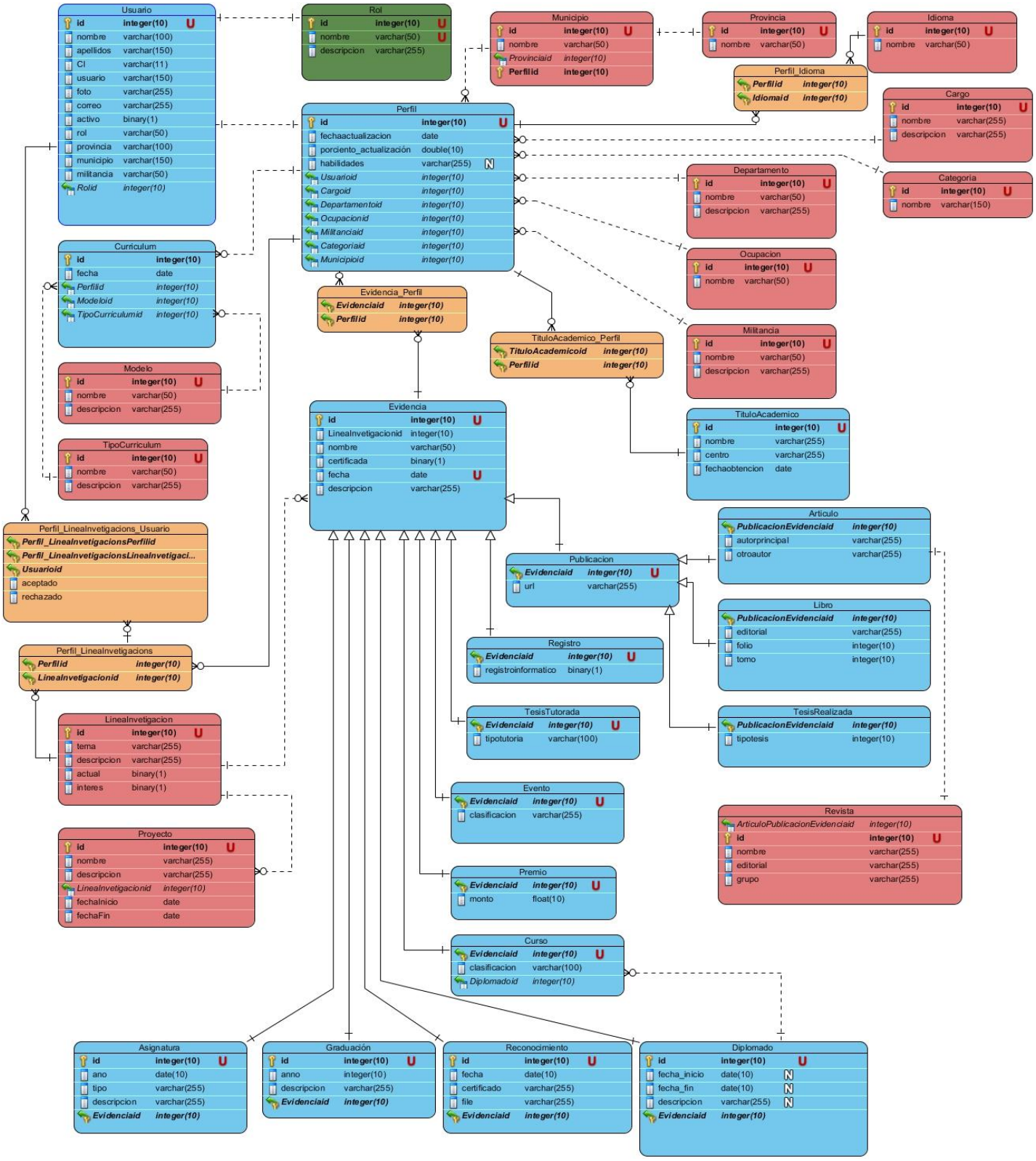

Figura 4. Modelo Entidad - Relación

A continuación se muestran las principales entidades que se utilizan en el desarrollo del sistema.

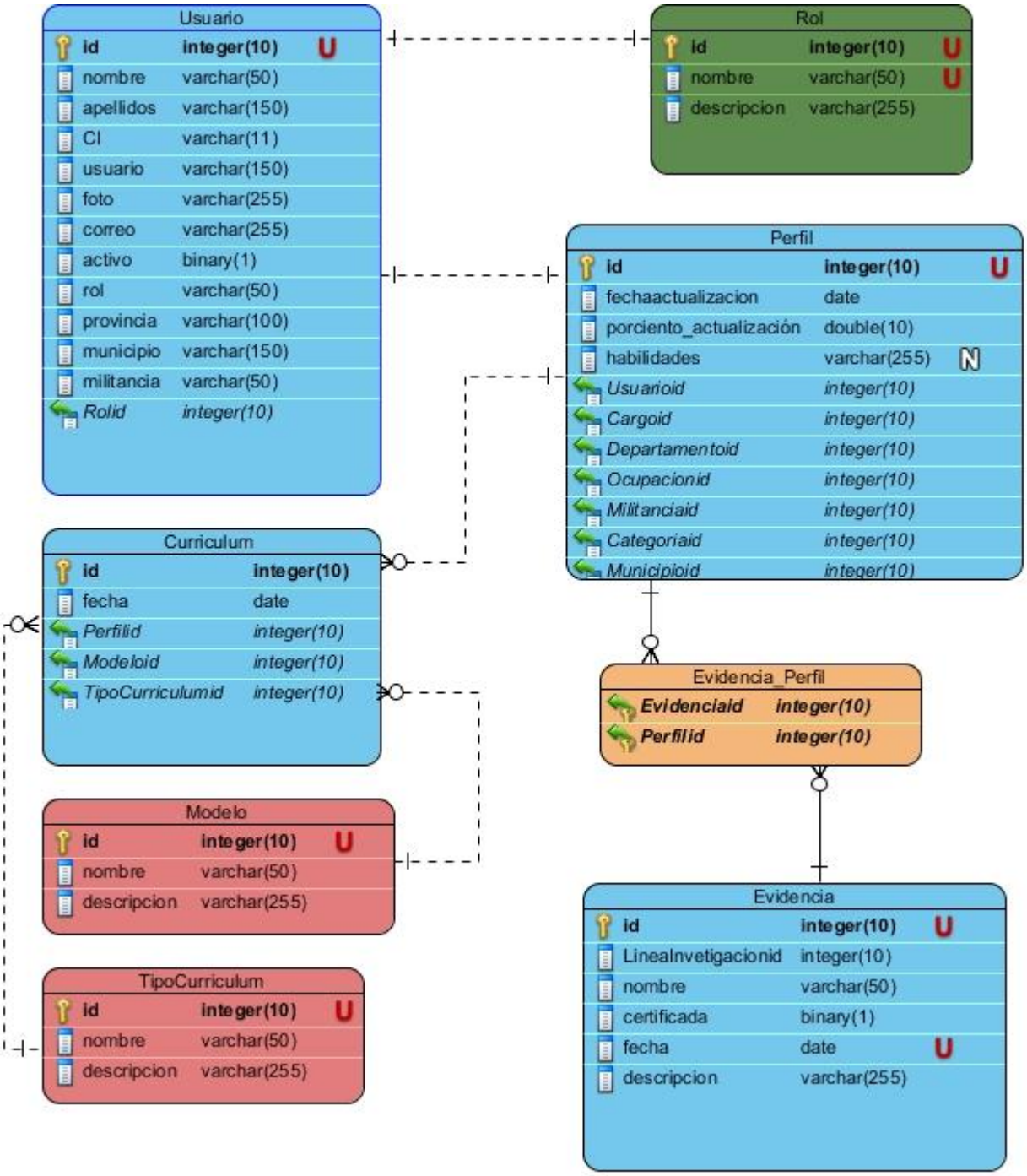

Figura 5. Principales entidades del sistema

2.3.3 Patrones de base de datos

Patrón Representación de Objetos como Tablas

El patrón Representación de Objetos como Tablas propone definir una tabla (si se emplea una base de datos relacional) para cada clase de objeto persistente. Los atributos de objetos que contienen tipos primitivos de datos (número, cadena, booleano y otros) se mapean en las columnas. (47)

En la Figura 6. **Representación del patrón Representación de objetos como tablas** se expone un ejemplo de cómo se evidencia el patrón en el modelo entidad-relación.

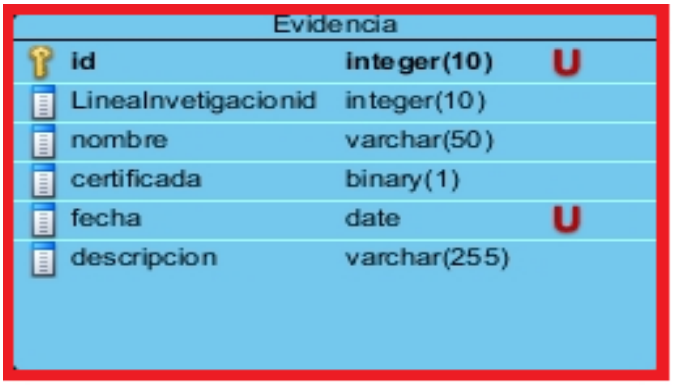

Figura 6. Representación del patrón Representación de objetos como tablas

Patrón Identificador de Objetos (IDO)

El patrón Identificador de Objetos (IDO) propone asignar un ID0 a cada registro y objeto. Un identificador de objetos suele ser un valor alfanumérico; es único para un objeto específico. (47)

En la Figura 7. **Representación del patrón Identificador de objetos** se expone un ejemplo de cómo se evidencia el patrón en el modelo entidad-relación.

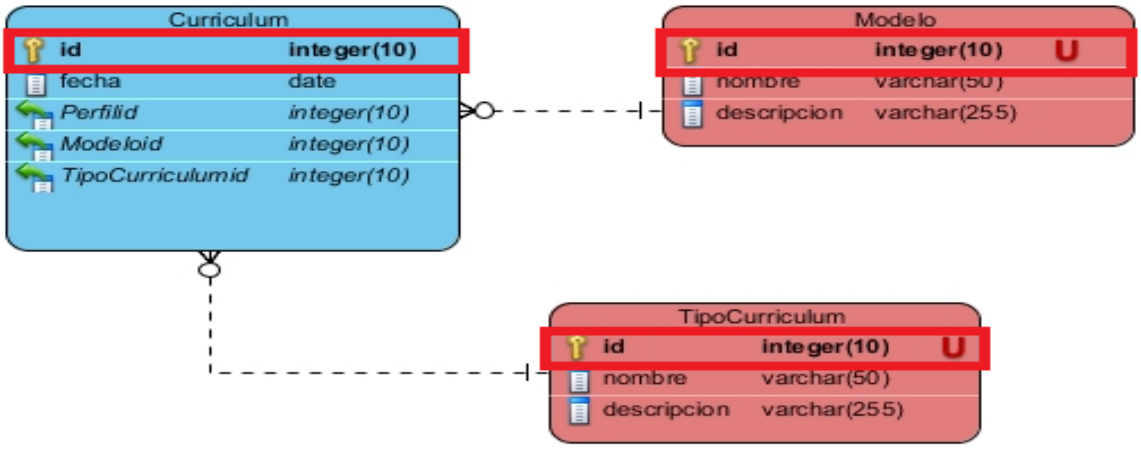

Figura 7. Representación del patrón Identificador de objetos

Patrón Clave subrogada

Una clave subrogada es un identificador único que se asigna a cada registro de una tabla de dimensión. Esta clave, generalmente, no tiene ningún sentido específico de negocio. Son siempre de tipo numérico. Preferiblemente, un entero autoincrementar. (48)

En la Figura 8. **Representación del patrón Clave subrogada** se expone un ejemplo de cómo se evidencia el patrón en el modelo entidad-relación.

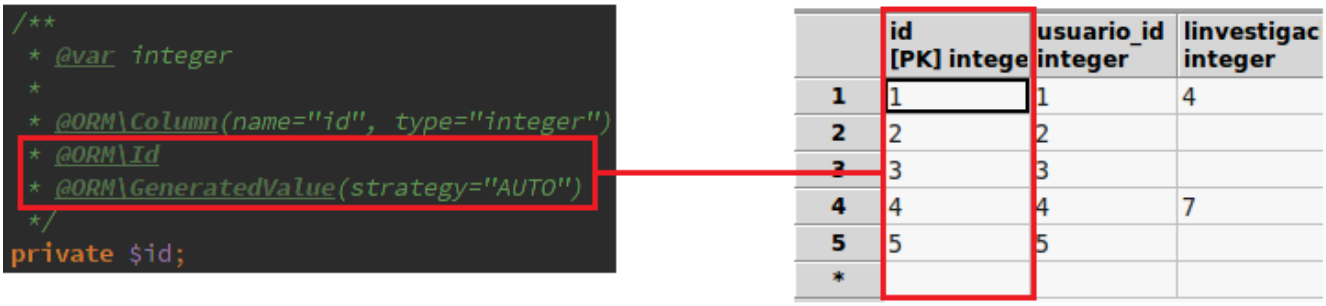

Figura 8. Representación del patrón Clave subrogada

2.3.4 Plan de Entrega

El plan de entregas es un documento que especifica exactamente que HU serán implementadas en cada entrega del sistema y sus prioridades, de modo que también permita conocer con exactitud qué HU serán implementadas en la próxima liberación. Debe ser negociado y elaborado en forma conjunta entre el cliente y el equipo desarrollo durante las reuniones de planificación de entregas.

A continuación se muestra una tabla con el resumen del plan de entregas, mostrando la planificación de la implementación de cada HU.

1era Entrega Iteración 1	2da Entrega Iteración 2	3era Entrega Iteración 3
8 de marzo	13 de abril	21 de mayo

Tabla 8. Cronograma de entrega de la solución

2.3.5 Patrones arquitectónicos y de diseños utilizados

Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el desarrollo de software. Estos brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares. (49)

Los patrones arquitectónicos expresan un esquema de organización estructural esencial para un sistema de software, que consta de subsistemas, sus responsabilidades e interrelaciones. El mismo resuelve problemas arquitectónicos, adaptabilidad a requerimientos cambiantes, modularidad, acoplamiento, etc. (50)

Patrón Modelo Vista Controlador (MVC)

El patrón de llamada y retorno MVC (según CMU[10]), se ve frecuentemente en aplicaciones web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página. El modelo es el Sistema de Gestión de Base de Datos y la Lógica de negocio, y el controlador es el responsable de recibir los eventos de entrada desde la vista. (51)

Modelo: esta es la representación específica de la información con la cual el sistema opera. En resumen, el modelo se limita a lo relativo de la vista y su controlador facilitando las presentaciones visuales complejas. El sistema también puede operar con más datos no relativos a la presentación, haciendo uso integrado de otras lógicas de negocio y de datos afines con el sistema modelado.

Vista: este presenta el modelo en un formato adecuado para interactuar, usualmente la interfaz de usuario.

Controlador: este responde a eventos, usualmente acciones del usuario, e invoca peticiones al modelo y, probablemente, a la vista. (51)

En la Figura 9. **Estructura de directorios del patrón MVC en Symfony2** se muestra la estructura de directorios del sistema, incluyendo la representación del patrón arquitectónico MVC.

[10] Universidad Carnegie Mellon (en inglés: Carnegie Mellon University, CMU).

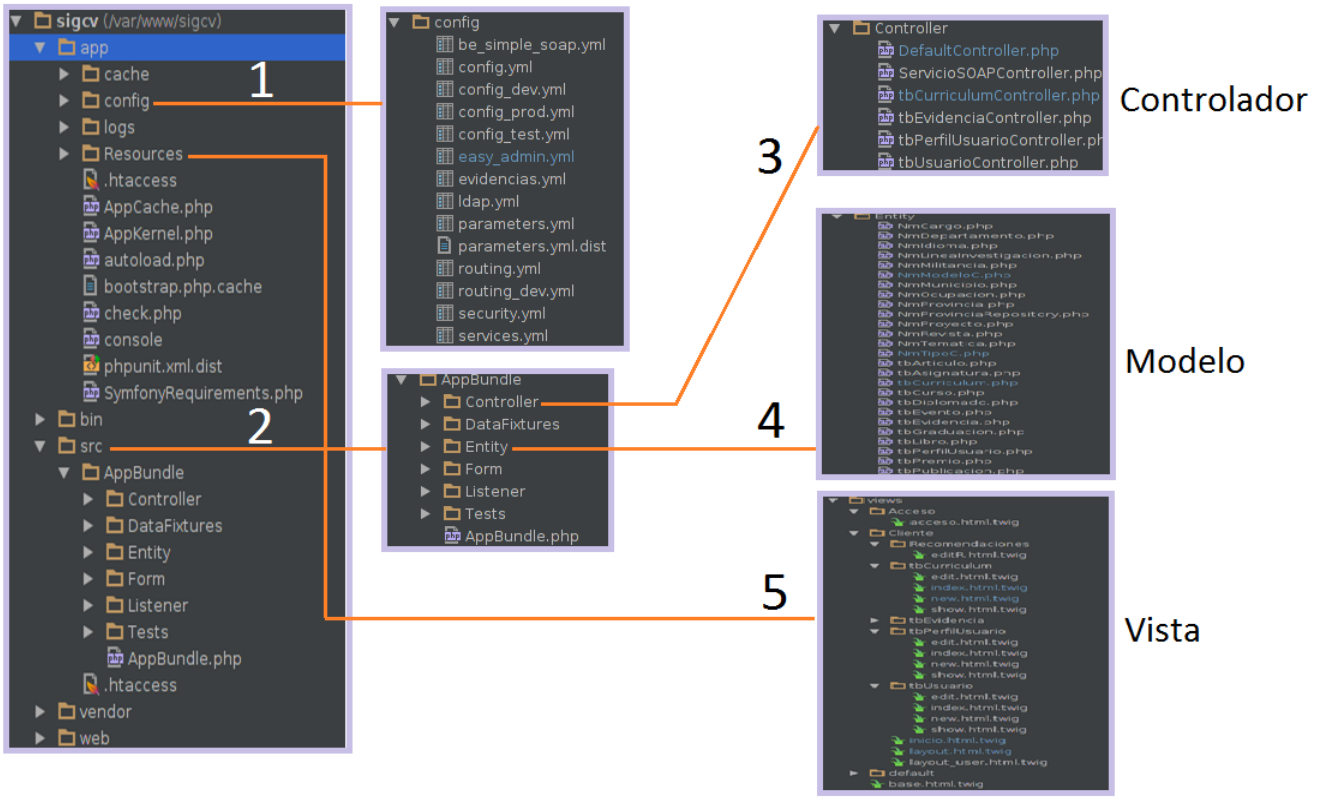

Figura 9. Estructura de directorios del patrón MVC en Symfony2

2.3.6 Patrones GRASP

Los patrones GRASP[11] describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en formas de patrones.

GRASP es un acrónimo que significa General *Responsibility Assignment Software Patterns.* El nombre se eligió para indicar la importancia de captar estos principios, si se quiere diseñar eficazmente el software orientado a objetos. (52)

Experto en información

La responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo. (53)

En el sistema desarrollado Symfony2 utiliza el mapeador de objetos y relaciones por sus siglas en inglés (ORM) Doctrine2 para:

[11] Patrones Generales De Software Para Asignación De Responsabilidades.

- Realizar su capa de abstracción en el modelo.
- Encapsular toda la lógica de los datos.
- Generar las clases con todas las funcionalidades comunes de las entidades.

Las clases de abstracción de datos poseen un grupo de funcionalidades que están relacionadas directamente con la entidad que representan y contienen la información necesaria.

Este patrón se evidencia en la clase ***tbUsuarioController*** permitiendo crear un nuevo y manipular el usuario autenticado.

```
class tbUsuarioController extends Controller
{
    public function indexAction()
    {
     $em = $this->getDoctrine()->getManager();
       $entities = $em->getRepository('AppBundle:tbUsuario')->findAll();
        return $this->render(':Cliente:tbUsuario/index.html.twig', array(
            'entities' => $entities,
        ));
    }
```

Creador

Es el responsable de la creación o instanciación de nuevos objetos o clases. Éste patrón refleja que la nueva instancia podrá ser creada por una clase si:

- Clase B contiene o agrega la clase A.
- Clase B es una agregación o composición de la clase A.
- Almacena la instancia en algún sitio (por ejemplo una base de datos).
- Tiene la información necesaria para realizar la creación del objeto (es 'Experta').
- Usa directamente las instancias creadas del objeto. (52)

Este patrón se evidencia en la clase ***PerfilUsuarioController*** permitiendo crear un nuevo perfil para el usuario autenticado.

```
class tbPerfilUsuarioController extends Controller
{
    public function createAction(Request $request)
    {
        $entity = new tbPerfilUsuario();
        $form = $this->createCreateForm($entity);
        $form->handleRequest($request);

        if ($form->isValid()) {
            $em = $this->getDoctrine()->getManager();
            $em->persist($entity);
            $em->flush();

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

        return $this->render('AppBundle:tbPerfilUsuario/new.html.twig', array(
            'entity' => $entity,
            'form' => $form->createView(),
        ));
    }
```

Controlador

Es un patrón que sirve como intermediario entre una determinada interfaz y el algoritmo que la implementa, de tal forma que es el controlador quien recibe los datos del usuario y quien los envía a las distintas clases según el método llamado. (54)

Este patrón se ve reflejado en la clase ***PerfilUsuarioController*** ya que se controla la información a almacenar en el perfil de usuario.

```
class tbPerfilUsuarioController extends Controller
{
    public function createAction(Request $request)
     public function showAction($id)
     {

     public function editAction($id)
     {

     private function createEditForm(tbPerfilUsuario $entity)
     {

     public function updateAction(Request $request, $id)
     {
     public function showAction($id)
     {

     public function editAction($id)
     {

     private function createEditForm(tbPerfilUsuario $entity)
     {

     public function updateAction(Request $request, $id)
```

Alta cohesión

El patrón Alta Cohesión mejora la claridad y facilidad con que se entiende el diseño. Se simplifica el mantenimiento y las mejoras de funcionalidad. A menudo se genera un bajo acoplamiento. Soporta mayor capacidad de reutilización. La cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas para que no realicen un trabajo enorme. (55)

En el sistema propuesto se evidencia la utilización del patrón alta cohesión al asignar a cada clase las responsabilidades que le corresponde y se establecen las condiciones para que cada clase colabore con las demás en la resolución de tareas que las implican a todas y que no son capaces de resolver por sí solas.

Bajo acoplamiento

El patrón Bajo Acoplamiento permite que al realizar cambios en un componente no se afecten otros. Son diseños fáciles de entender por separado y fáciles de reutilizar. El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, las conoce y recurre a ellas. Acoplamiento alto significa que una clase recurre a muchas otras clases. (55)

El bajo acoplamiento se evidencia en el sistema propuesto en las clases que implementan la lógica del negocio y de acceso a datos en el modelo, puesto que estas clases no tienen relación con las de la vista o el controlador por lo que la dependencia en este caso es baja.

2.3.7 Patrones GOF

Decorador

La plantilla base base.html.twig que se encuentra en el directorio /app/Resources/, padre de todas las vistas, conocida como plantilla global, contiene el código HTML, así como las referencias a los ficheros CSS y JavaScript que es tradicional en todas las páginas del sistema, para no tener que repetirlo. El contenido de la plantilla se integra en las demás vistas del sistema, a través de la utilización de los bloques los cuales se heredan jerárquicamente a medida que desciende el nivel de la página.

```
<html>
    <head>
        <meta charset="UTF-8" />
        <title>{% block title %}MaiJOB | {% block pagina %} {% endblock %}{% endblock %}</title>
        {% block stylesheets %}
            <link rel="stylesheet" type="text/css" href="{{asset('bundles/comun/css/bootstrap.css')}}" />
            <link rel="stylesheet" type="text/css" href="{{asset('bundles/comun/css/sb-admin.css')}}" />
            <link rel="stylesheet" type="text/css" href="{{asset('bundles/comun/font-awesome/css/font-awesome.min.css')}}" />
            <link rel="stylesheet" type="text/css" href="{{asset('bundles/comun/css/morris.min.css')}}" />
        {% endblock %}
       <link rel="icon" type="image/x-icon" href="{{ asset('bundles/nomenclador/images/logo_app.png') }}" />
    </head>
    <body style="background-image:url(../{%block background%}{% endblock %});background-repeat: no repeat;background-attachment
webkit-background-size: cover;-moz-background-size: cover;-o-background-size: cover;background-size: cover;">
        {% block body %}{% endblock %}
        {% block javascripts %}
            <script type="text/javascript" src="{{asset('bundles/comun/js/jquery-2.0.0.min.js')}}"></script>
            <script type="text/javascript" src="{{asset('bundles/comun/js/bootstrap.js')}}"></script>
            <script type="text/javascript" src="{{asset('bundles/comun/js/raphael-2.1.0.min.js')}}"></script>
            <script type="text/javascript" src="{{asset('bundles/comun/js/morris/morris.min.js')}}"></script>
            <script type="text/javascript" src="{{asset('bundles/comun/js/morris/chart-data-morris.js')}}"></script>
            <script type="text/javascript" src="{{asset('bundles/comun/js/tablesorter/jquery.tablesorter.js')}}"></script>
            <script type="text/javascript" src="{{asset('bundles/comun/js/tablesorter/tables.js')}}"></script>
        {% endblock %}
    </body>
</html>
```

Inyección de dependencias

Inyección de dependencias, es un patrón de diseño, que permite "inyectar" objetos a una clase, en vez de que la clase cree el objeto por sí solo. Esto permite que la clase que utilice el objeto no requiera especificar como crearlo, sino que otra clase se ocupará en crear este objeto según sus especificaciones y simplemente será agregado en la clase.

Típicamente este contenedor es implementado por un marco de trabajo externo a la aplicación por lo cual en la aplicación también se utilizará inversión de control al ser el contenedor quien invoque el código de la aplicación. Esta es la razón por la que los términos de inversión de control e inyección de dependencias se confunden habitualmente entre sí. (56)

Dentro del marco de trabajo Symfony2 cada bundle[12] proporciona un fichero de configuración llamado ***services.yml***. Este contiene dos secciones ***parameters*** y ***services***. En el primero se declaran todos los parámetros de configuración y en el segundo todos los servicios que se utilizarán.

Los controladores heredan de la clase *Controller*, la cual proporciona un atributo público llamado ***container*****,** una instancia del contenedor de dependencias. Este permite que desde cualquier clase derivada de la anteriormente mencionada se pueda obtener una instancia del contenedor de dependencias por lo que se puede instanciar cualquier servicio existente haciendo uso de la función ***get***.

[12] Es un conjunto estructurado de archivos que se encuentran en un directorio y que implementan una sola característica.

Conclusiones parciales

Del presente capítulo se concluye que:

- Las historias de usuarios y los RNF permiten obtener un sistema multiplataforma, fácil de usar y con disponibilidad de información todo el tiempo. Así como la flexibilidad del mismo a partir de los patrones usados en su desarrollo.
- Con la realización de los artefactos propuestos por la metodología se modeló el proceso de negocio y se obtuvo la propuesta de solución del sistema.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBAS

INTRODUCCIÓN

En este capítulo se implementan las tres iteraciones llevadas a cabo por la aplicación, además se muestra el diagrama de componentes y el diagrama de despliegue de la solución. Se describen los estándares de codificación utilizados. Por último, se le realizan las validaciones y pruebas pertinentes al producto final, verificando de forma unitaria las funcionalidades de la aplicación y validando junto al usuario la solución desarrollada.

3.1 Diagrama de componentes

El diagrama de componente permite visualizar con más facilidad la estructura general del sistema y el comportamiento del servicio que estos componentes proporcionan y utilizan a través de las interfaces. El mismo representa cómo un sistema de *software* es dividido en componentes y muestra las dependencias entre estos componentes. Los elementos de un diagrama de componente son: componentes, interfaces, relaciones de dependencias y los paquetes o subsistemas. (57)

A continuación se muestra el diagrama de componente correspondiente al sistema propuesto.

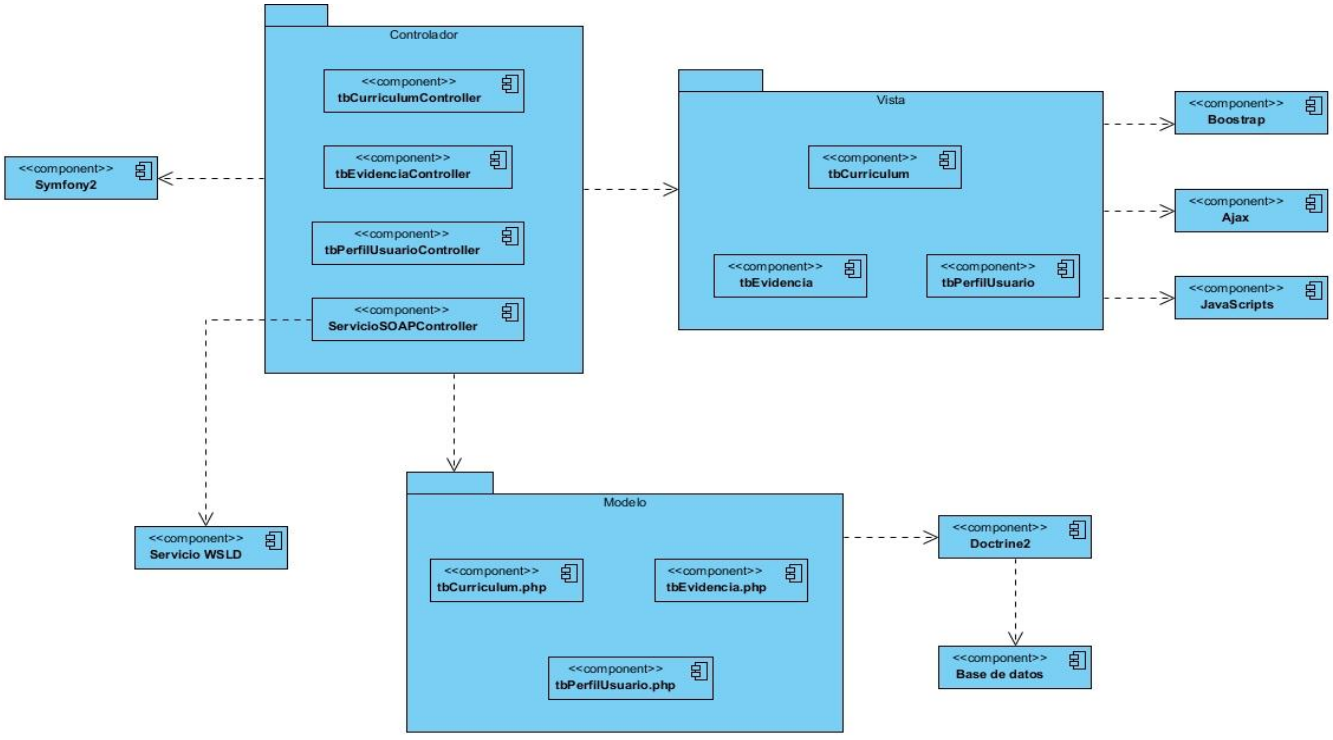

Figura 10. Diagrama de Componentes

El componente ***CurrículumController*** es el encargado de conformar el currículo del usuario autenticado, el mismo consume servicio de los componentes ***PerfilUsuario*** y ***Evidencias*** reflejadas hasta el momento y que hayan sido aceptadas. El paquete ***Vista*** es el encargado de manejar todas la interfaces del sistema, así como las peticiones de los usuarios y los componentes antes descritos, para luego enviarlas al paquete ***Controlador*** el cual se encarga de relacionar estas peticiones con el paquete ***Modelo***, este último se encarga de gestionar el acceso a datos.

3.2 Diagrama de despliegue

El diagrama de despliegue muestra las relaciones físicas de los distintos componentes *hardware* y *software* que componen un sistema. En el diagrama se observa que los nodos representan un recurso de cómputo o un dispositivo de *hardware* similar. Los mismos son conectados por el protocolo seguro HTTPS. (58)

Figura 11. Diagrama de despliegue

Para el despliegue de la aplicación se debe contar con una ***PC Cliente*** conectada a un ***Servidor de Base de datos*** y ***aplicaciones*** al cual el cliente tendrá acceso por el protocolo ***https:443***, el mismo consume servicio del ***Servidor SOAP*** para la autenticación de los usuarios, la comunicación será por el protocolo seguro ***https:443***.

3.3 Fase de implementación

En esta fase la metodología XP define que la implementación de las HU se haga de manera iterativa de forma tal que al final de cada iteración se obtenga un producto funcional siendo este probado y mostrado al cliente. Se define los estándares de codificación a utilizar en el sistema.

3.3.1 Estándares de codificación

XP promueve la programación basada en estándares, de manera que sea fácilmente entendible por todo el equipo, y que facilite la recodificación. Un estándar de codificación comprende todos los aspectos de la generación de código. Un código fuente debe reflejar un estilo armonioso, como si un único programador hubiera escrito todo el código de una sola vez. Usar técnicas de codificación sólidas y realizar buenas prácticas de programación con vistas a generar un código de alta calidad es de gran importancia para la calidad del *software* y para obtener un buen rendimiento. (59)

Estilos de codificación utilizados:

- Los *namespaces*[13] y las clases deben tener la siguiente estructura <*Vendor name*> *(<Namespace>)*<Class Name>.*
- Los nombres de los *namespaces* o clases deben ser ordenadas alfabéticamente.
- Los archivos sólo deben utilizar una codificación UTF-8[14].
- El número de caracteres por línea deben ser de 80 columnas aunque también esta aceptado que sean hasta 120.
- Las constantes de PHP true, false y null deben estar en minúsculas. (60)
- Notación CamelCasing: en la redacción de las clases la letra inicial del identificador no debe estar en mayúscula.

3.4 Fase de pruebas

En esta fase se realizan las pruebas planteadas por la metodología XP. Las mismas son utilizadas para identificar posibles fallos de diseño e implementación, calidad o usabilidad del sistema informático desarrollado. La metodología XP divide las pruebas en dos grupos: las pruebas unitarias, que son realizadas por los programadores, encargadas de verificar el código y las pruebas de aceptación, destinadas a verificar si al final de cada iteración se obtuvo la funcionalidad que se requiere, además de comprobar que dicha funcionalidad haga lo que el cliente quiere. (61)

[13] En su definición más aceptada, los espacios de nombres son una manera de encapsular elementos.

[14] UTF-8. (8-bit *Unicode Transformation Format*). Es una norma de transmisión de longitud variable para codificación de caracteres codificados utilizando *Unicode*, creada por Rob Pike y Ken Thompson.

3.4.1 Validación del diseño

Para medir el diseño se decidió utilizar métricas básicas inspiradas en el estudio de la calidad del diseño orientado a objeto, principalmente la métrica Relaciones entre Clases (RC) y la métrica Tamaño Operacional de Clase (TOC). Estas fueron seleccionadas para la validación del diseño dado el nivel de aceptación que tienen en el ambiente donde son aplicadas:

3.4.1.1 Métrica Relaciones entre Clases

La métrica fue propuesta por Shyam R Chidamber y Chris F Kemerer en 1994, y es conocida en inglés como *Coupling between Object Classes.* La misma relaciona la noción de que una clase está acoplada a otra si instancias de la primera clase, utilizan métodos o variables de instancia de los objetos de otra clase, lo cual no es deseable en el paradigma orientado a objetos por las siguientes causas:

- Cuanto más independiente es una clase, es más fácil para otra aplicación reutilizar dicha clase.
- Si hay muchas dependencias entre las clases que conforma nuestro sistema, es difícil ver qué cambios en una parte tendrán consecuencias.
- Cuanto más alto sea el acoplamiento más riguroso ha de ser las pruebas requeridas sobe las clases involucradas. (62)

Esta métrica está definida por el número de relaciones de uso de una clase con otra, en esta se evalúan varios atributos de calidad:

- **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.
- **Reutilización:** un aumento del RC implica una disminución en el grado de reutilización de la clase.
- **Cantidad de pruebas:** un aumento del RC implica un aumento de la Cantidad de pruebas de unidades necesarias para probar una clase.

El resultado obtenido al aplicar cada uno de los atributos de calidad de la métrica se le denomina umbral, el cual determina la categoría óptima para cada uno de los atributos de calidad.

Atributos de calidad	Categoría	Criterio

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

Tabla 9. Rango de valores para medir la afectación de los atributos de calidad RC

Nota: el umbral obtenido para que sea óptimo tiene que ser el valor resaltado en azul en cada una de las categorías de cada uno de los atributos de calidad.

A continuación se muestra el resultado de aplicar las métricas de diseño.

Criterio	Categoría	Cantidad de clases	Promedio
0 dependencias	Muy Bueno	0	0
1 dependencias	Bueno	27	72,97297297
2 dependencias	Regular	7	18,91891892
3 dependencias	Malo	2	5,405405405
> 3 dependencias	Muy Malo	1	2,702702703
Total		37	100

Tabla 10. Promedio por cantidad de clases

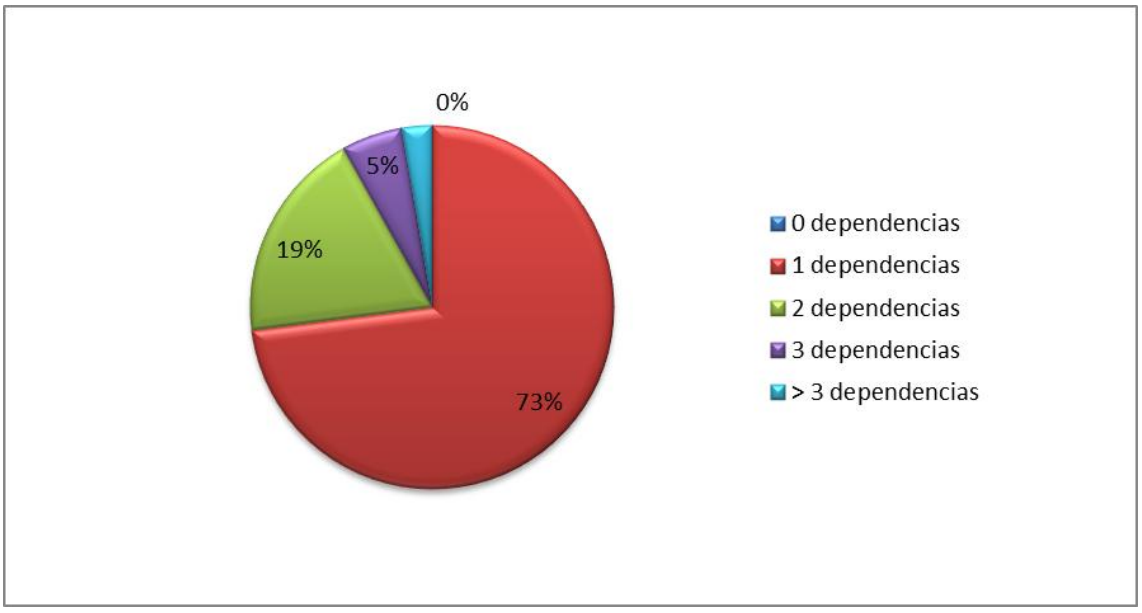

Figura 12. Dependencia entre clases

Acoplamiento

Acoplamiento	Cantidad de clases	Promedio
Ninguno	0	0
Bajo	27	72,97297297
Medio	7	18,91891892
Alto	3	8,108108108

Tabla 11. Acoplamiento

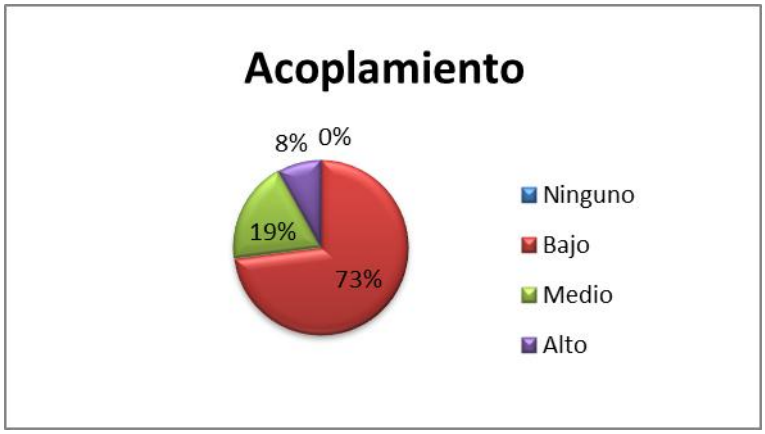

Figura 13. Evaluación del atributo acoplamiento

Complejidad de mantenimiento

Complejidad de Mantenimiento	Cantidad de clases	Promedio
Baja	27	72,97297297
Media	9	24,32432432
Alta	1	2,702702703

Tabla 12. Complejidad de mantenimiento

Figura 14. Evaluación del atributo complejidad de mantenimiento

Cantidad de pruebas

Cantidad de Pruebas	Cantidad de clases	Promedio
Baja	27	72,97297297
Media	9	24,32432432
Alta	1	2,702702703

Tabla 13. Cantidad de pruebas

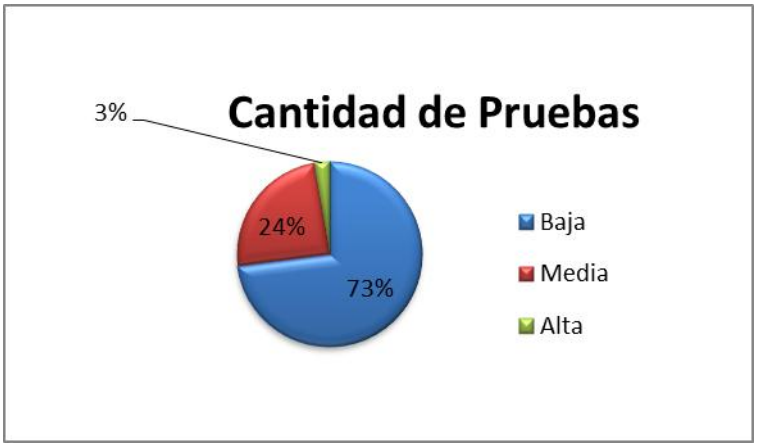

Figura 15. Evaluación del atributo cantidad de pruebas

Reutilización

Reutilización	Cantidad de clases	Promedio
Baja	1	2,702702703
Media	9	24,32432432
Alta	27	72,97297297

Tabla 14. Reutilización

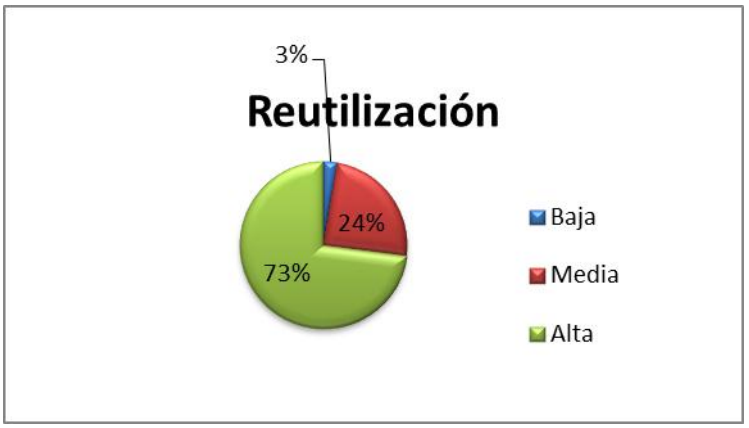

Tabla 15. Evaluación del atributo reutilización

Los resultados obtenidos demuestran que el diseño propuesto para el sistema informático de gestión curricular se encuentra dentro de los niveles de calidad requeridos. Los atributos de calidad fueron evaluados satisfactoriamente confirmando que las clases están diseñadas correctamente, presentan una elevada reutilización, un bajo acoplamiento, poca complejidad de mantenimiento y poca cantidad de pruebas que implica que se necesitará menos esfuerzo a la hora de realizar pruebas unitarias a estas clases.

3.4.1.2 Métrica Tamaño Operacional de Clase

En el libro de métricas *Object-Oriented Systems* realizado por Lorenz y Kidd, dividen las métricas basadas en clases en cuatro categorías: tamaño, herencia, valores internos y valores externos. Las métricas orientadas a tamaños para una clase orientada a objetos (OO) se centran en cálculos de atributos y de operaciones para una clase individual, y promedian los valores para el sistema OO en su totalidad. Las métricas basadas en herencia se centran en la forma en que se reutilizan las operaciones a lo largo y ancho de la jerarquía de clases. Las métricas para valores internos de clase examinan la cohesión y asuntos relacionados con el código, y las métricas orientadas a valores externos examinan el acoplamiento y la reutilización. (63)

La métrica tamaño de clase (TOC), define que el tamaño general de una clase se puede determinar empleando las medidas siguientes:

- El número total de operaciones (tanto operaciones heredadas como privadas de la instancia) que están encapsuladas dentro de la clase.
- El número de atributos (tanto atributos heredados como atributos privados de la Instancia) que están encapsulados en la clase.

Al aplicar la métrica se tendrá en cuenta varios atributos de calidad que se relacionan a continuación:

- **Responsabilidad:** es la responsabilidad asignada a una clase en un marco de modelado de un dominio.
- **Complejidad de implementación:** consiste en el grado de dificultad que tiene que implementar un diseño de clases determinado.
- **Reutilización:** consiste en la reutilización presente en una clase o estructura de clases, dentro de un diseño de software.

El TOC se puede determinar por el número operaciones que el mismo posee, el resultado obtenido se denomina umbral, el cual determina la categoría óptima para cada uno de los atributos de calidad. Si el valor de TOC es grande, demuestra que la clase puede tener demasiada responsabilidad, lo cual reduce su reutilización y complejiza su implementación.

Atributos de calidad	Categoría	Criterio
Responsabilidad	Baja	<= Prom.
	Media	Entre Prom. y 2*Prom.
	Alta	>2*Prom.
Complejidad de Implementación	Baja	<=Prom.
	Media	Entre Prom. y 2*Prom.
	Alta	>2*Prom.
Reutilización	Baja	>2*Prom.
	Media	Entre Prom. y 2*Prom.
	Alta	<=Prom.

Tabla 16. Rango de valores para medir la afectación de los atributos de calidad TOC

Nota: el umbral obtenido para que sea óptimo tiene que ser el valor resaltado en azul en cada una de las categorías de cada uno de los atributos de calidad.

A continuación se muestra el resultado de aplicar las métricas de diseño.

Responsabilidad

Responsabilidad	Cantidad de clases	Promedio
Baja	25	67,56756757
Media	9	24,32432432
Alta	3	8,108108108

Tabla 17. Responsabilidad

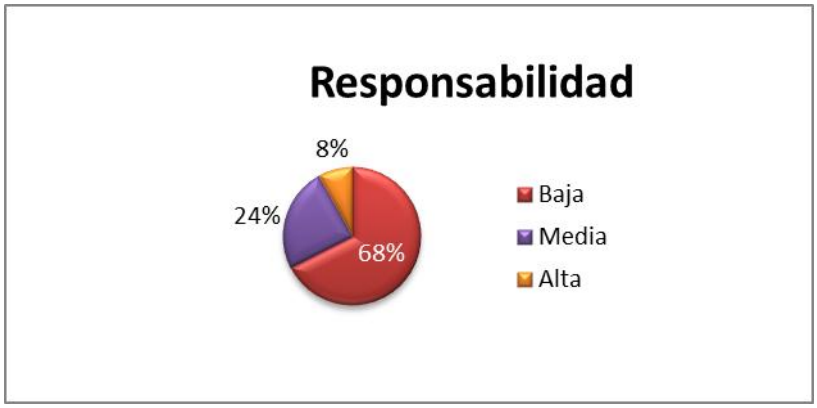

Figura 16. Evaluación del atributo de calidad Responsabilidad

Complejidad de Implementación

Complejidad	Cantidad de clases	Promedio
Baja	25	67,56756757
Media	9	24,32432432
Alta	3	8,108108108

Tabla 18. Complejidad

Figura 17. Evaluación del atributo Complejidad

Reutilización

Reutilización	Cantidad de clases	Promedio
Alta	25	67,56756757
Media	9	24,32432432
Baja	3	8,108108108

Tabla 19. Reutilización

Figura 18. Evaluación del atributo Reutilización

Una vez realizada la aplicación de la métrica TOC se obtuvo como resultado que el diseño cumple con los requerimientos inicialmente propuestos, arrojando como resultado una baja responsabilidad, complejidad y una alta reutilización, permitiendo la eliminación de clases repetidas, teniendo en cuenta que más de la mitad de las clases poseen baja dependencia respecto a otras, y el sistema no tendrá una compleja implementación.

3.4.2 Pruebas

Las pruebas de *software* se puede definir como un conjunto de actividades y técnicas en la cual un sistema o uno de sus componentes se ejecuta en circunstancias previamente especificadas (configuración de la prueba), registrándose los resultados obtenidos. Dentro de las técnicas se encuentran: las técnicas de evaluación estática y técnicas de evaluación dinámicas. (64)

Técnicas de evaluación estática

Esta técnica consiste en realizar revisiones y análisis estáticos a la documentación del proyecto. Las revisiones pretenden detectar ***manualmente*** defectos en cualquier producto del desarrollo. Los revisores analizan el sistema mediante la lectura del mismo, sin ejecutarlo. (64)

Técnicas de evaluación dinámica

Las técnicas de evaluación dinámica son todas aquellas pruebas que para su realización ***requieren la ejecución de la aplicación***. Las mismas proporcionan distintos criterios para generar casos de pruebas que provoquen fallos al sistema. Estos criterios son:

- **Técnicas de caja blanca o estructural:** se basan en un minucioso examen de los detalles procedimentales del código a evaluar, por lo que es necesario conocer la lógica del sistema.
- **Técnicas de caja negra o funcional:** realizan pruebas sobre la interfaz del programa a probar, entendiendo por interfaz las entradas y salidas de dicho programa. No es necesario conocer la lógica del sistema, únicamente la funcionalidad que debe realizar. (64)

Después de haber terminado cada iteración, se procede a aplicarles técnicas y pruebas de validación al código fuente y a las interfaces definidas hasta el momento, las mismas se validan por parte del cliente. Symfony2 permite realizar estas pruebas mediante la herramienta PHPUnit, la cual se utiliza para realizar las pruebas unitarias al sistema.

3.4.2.1 Pruebas unitarias

Las pruebas unitarias verifican que cada método y función está trabajando correctamente. Cada prueba deberá ser lo más independiente posible de las demás. Las pruebas unitarias garantizan que un único componente produce una salida correcta para una determinada entrada. Las mismas validan la forma en la que las funciones y métodos trabajan en cada caso particular. El conjunto de pruebas debe ser guardado junto con el código, para que pueda ser utilizado por otros desarrolladores, en caso de tener que corregir, cambiar o recodificar parte del mismo. (39)

En la Figura 19. **Resultado de las pruebas** se muestra la estructura de directorios donde se encuentran las pruebas dentro del proyecto Symfony2, el cual está constituido por las carpetas *Controller* y *Entity*, donde se ubican las pruebas realizadas a los controladores y entidades respectivamente. Se realizaron un total de 65 pruebas a las entidades y a los controladores del sistema, arrojando como resultado que las 65 fueron satisfactorias, para un 100%.

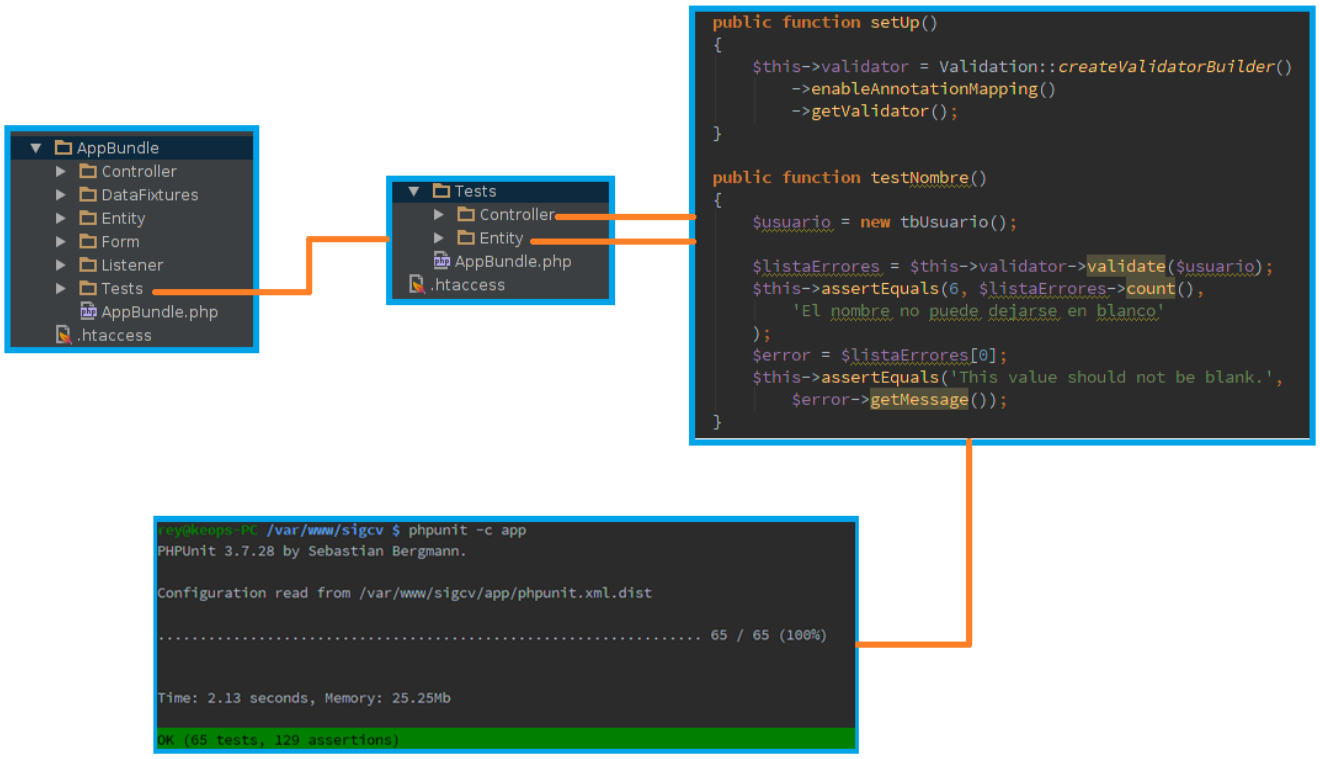

Figura 19. Resultado de las pruebas

3.4.2.2 Pruebas de aceptación

Las pruebas de aceptación son realizadas por el cliente, ayudado por el equipo de pruebas siendo deseable que el cliente establezca los casos de pruebas. Los mismos están enfocados para probar los requerimientos del usuario. Es recomendable que se haga en el entorno en el que se va explotar el sistema (incluyendo el personal que lo maneje). En caso de un producto de interés general, se realizan pruebas con varios usuarios que reportarán sus valoraciones sobre el producto. (65)

A continuación se muestra los principales casos de pruebas.

Caso de prueba aceptación	
Código: CPA-1	**Número de HU:** 01
Nombre: Confeccionar perfil de usuario.	
Descripción: demostrar que el sistema crea un perfil para ser actualizado.	
Resultado esperado: confeccionar un perfil.	
Evaluación de la prueba: prueba satisfactoria	

Tabla 20. Caso de prueba de aceptación Confeccionar perfil de usuario

Caso de prueba aceptación	
Código: CPA-4	**Número de HU:** 04
Nombre: Confeccionar CV.	
Descripción: demostrar que una vez registrado en el sistema se puede confeccionar un CV.	
Resultado esperado: mostrar que el CV se confecciono.	
Evaluación de la prueba: prueba satisfactoria	

Tabla 21. Caso de prueba de aceptación Confeccionar Currículum vítae

Caso de prueba aceptación	
Código: CPA-8	**Número de HU:** 08
Nombre: Certificar evidencia.	
Descripción: demostrar que una vez reflejada evidencias, esta puede pasar por 3 estados: rechazada, en espera y aceptada.	
Resultado esperado: mostrar la evidencia rechazada, en espera o aceptada.	
Evaluación de la prueba: prueba satisfactoria	

Tabla 22. Caso de prueba de aceptación Certificar evidencia

El desarrollo del sistema estuvo dividido en 3 iteraciones, de las cuales 29 HU se realizaron en la primera iteración concluyendo de forma total 23 HU y quedando inconformidades en las restantes 7 HU. En la segunda iteración se corrigieron las no conformidades detectadas en la iteración anterior y se implementaron 25 nuevas HU, obteniendo como resultado 3 no conformidades. La tercera iteración estuvo marcada por un trabajo exhaustivo por parte del equipo de desarrollo logrando corregir las no conformidades de la iteración anterior e implementando de forma correcta y satisfactoria las 27 HU restantes, para un total de 81 HU.

En la Figura 20. **Resultado de las pruebas de aceptación** se muestra una gráfica en la que se describe el proceso de desarrollo concerniente a lo antes descrito.

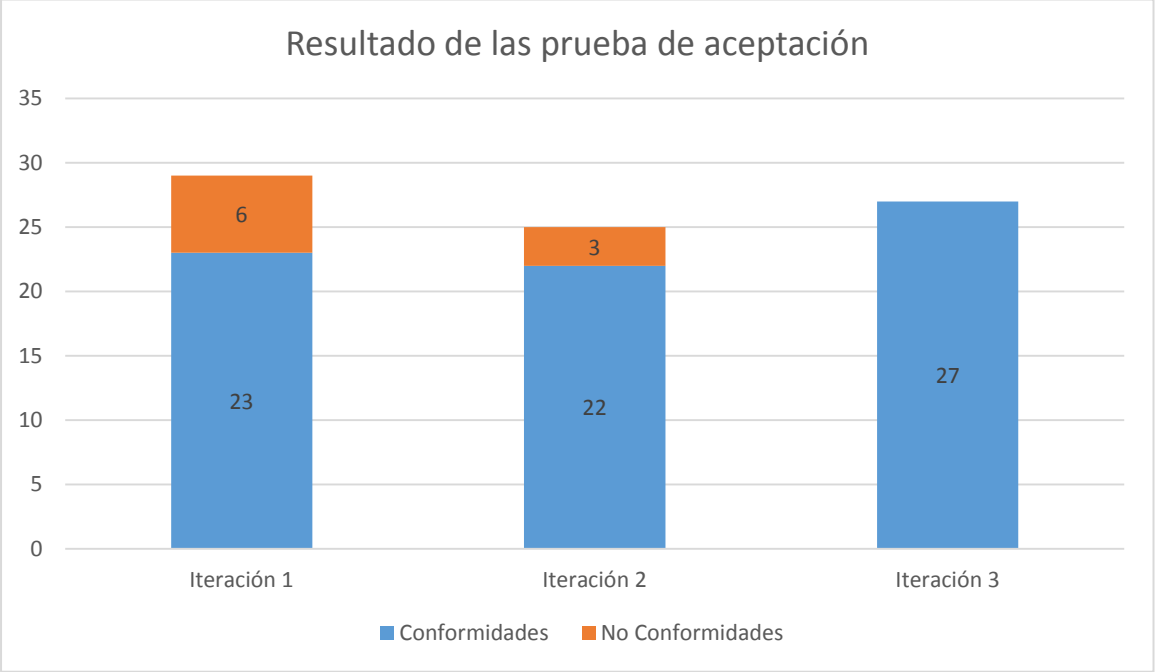

Figura 20. Resultado de las pruebas de aceptación

Conclusiones parciales

A partir de la implementación y validación del sistema propuesto se puede concluir que:

- Con la aplicación de las métricas RC y TOC se validó el diseño de la solución, comprobando que los atributos cumplieron con los parámetros establecidos para asegurar que estaban correctos para obtener un diseño responsable adaptable a cualquier entorno.
- Las pruebas unitarias del sistema permitieron validar que las entidades creadas y sus métodos funcionaban correctamente.
- Las pruebas funcionales permitieron verificar que la validación de los datos utilizados en los casos de pruebas se realizó de forma correcta permitiendo informar al usuario en todo momento de la ausencia de datos que son de interés e importancia para la conformación de las evidencia.

CONCLUSIONES

- Al finalizar la presente investigación se lograron cumplir los objetivos planteados de manera satisfactoria, obteniendo como resultado un sistema que permite gestionar la información curricular de los trabajadores de la facultad 3 basado en los modelos internacionales.
- Las validaciones realizadas sobre el diseño de la aplicación muestra la flexibilidad del sistema para adaptarse a otros entornos.
- Las pruebas realizadas al sistema muestran un producto que cumple con las historias de usuarios, requisitos no funcionales y la satisfacción del cliente.

RECOMENDACIONES

Después de alcanzados los objetivos trazados al inicio del trabajo los autores del mismo, proponen las siguientes recomendaciones:

- Estudiar la factibilidad de adecuar el sistema para gestionar el CV del trabajo de los estudiantes, más allá del expediente que obtiene hoy de otros sistemas.
- Realizar un estudio de la factibilidad de incluir el sistema en entornos laborales donde no se posean los datos y que estos no sean consumidos mediante servicios.

REFERENCIAS BIBLIOGRÁFICAS

1. Sánchez, José Antonio Fernández, y otros. Métodos para la búsqueda de empleo. Alicante : Publicaciones de Universidad de Alicante, 2014.

2. Zeroo, Beastieux. Scribd. [En línea] 22 de 11 de 2008. [Citado el: 9 de 12 de 2014.] http://es.scribd.com/doc/8255409/Metodologias-para-la-geston-y-desarrollo-de-Software.

3. Definicion. Definicion.de. Curriculum Vítae. [En línea] 2008. [Citado el: 2 de 11 de 2014.] http://definicion.de/curriculum/.

4. Dirección de Prensa y Comunicaciones. Ministerio de Trabajo, Empleo y Seguridad Social de la Nación, Republica de Argentina. Ministerio de Trabajo, Empleo y Seguridad Social de la Nación, Republica de Argentina. [En línea] 1996-2015. [Citado el: 9 de 11 de 2014.] http://www.trabajo.gov.ar/buscastrabajo/cv/.

5. Cabrera, Teresa Sanz, y otros. Curriculum y formación profesional. La Habana : Departamento de Ediciones e Imprenta ISPJAE - CUJAE, 2003.

6. Algunas consideraciones sobre la confección del Curriculum Vitae. Miranda, MsC. Daniel Reyes, Pellón, Lic. Raiza Texidor y Pérez, Dra. Alina María Segredo. 2004;20(5-6), Ciudad de La Habana : Rev Cubana Med Gen Integr, 27 de 12 de 2004, págs. 5-6.

7. Diccionario de la lengua española Espasa-Calpe. Diccionario de la lengua española. 2005.

8. Información: Una nueva propuesta conceptual. Angulo, Marcial N. No.4, México : Revistas Ciencias de la Información, diciembre de 1996, Vol. CD Ciencias de la Informaci?n Textos completos.

9. Delgado, M.C. Gabino Estevez y Hernández, Ing. Eduardo Ochoa. CIE iUMSNH. [En línea] 9 de 2001. [Citado el: 2 de 11 de 2014.] http://dieumsnh.qfb.umich.mx/gesinfo/.

10. Information management in large organizations. Woodman, L. [ed.] B CRONIN. London : ASLIB, 1985, Information management from strategies to action., págs. 95-114.

11. Intelligence management. White. [ed.] B. Cronin. London : s.n., 1985, Information management from strategies to action.

12. PONJUAN D., Gloria. Gestión de Información: dimensiones e implementación para el éxito organizacional. Rosario : Nuevo Paradigma, 2004. ISBN: 987-96536-6-1.

13. Gordon B. Davis, Margrethe H. Olson. Management information systems: conceptual foundations, structure, and development (2nd ed.) . New York : McGraw-Hill Inc., 1985. ISBN-10: 007066241X / ISBN-13: 978-0070662414.

14. Introducción al estudio de la información y la documentación. González, José Antonio Moreiro. 027, Mexico : IBÁÑEZ MARMOLEJO, 1999, Vol. 13. ISSN: 0187-358X.

15. Meléndrez, Lic. Ederlys Hernández y Gómez, Lic. Adelkys Rosa Sánchez. GestioPolis. GestioPolis. [En línea] 4 de 2006. [Citado el: 28 de 11 de 2014.] http://www.gestiopolis.com/canales6/fin/evidencias-y-papeles-de-una-auditoria.htm.

16. Linares Armas, MSc. Dayamy , Serralvo Cala, MSc. Miriam y Valdés Parada, Lic. Luisa Elena. Sistema informático para la certificación de publicaciones en el Centro de Recursos para el Aprendizaje y la Investigacion(CRAI). Cienfuegos : Universidad y Sociedad, 2014.

17. Firma Profesional. FirmaProfesional. [En línea] https://www.firmaprofesional.com/esp/soluciones-y-servicios/2014-02-13-11-59-39/publifirma-2.

18. CurriculumQ.com. http://curriculumq.uptodown.com/. http://curriculumq.uptodown.com/. [En línea] 04 de 03 de 2013. [Citado el: 03 de 12 de 2014.] http://curriculumq.uptodown.com/.

19. AGBO Business Architecture S.L. http://www.curriculumfacil.es/. http://www.curriculumfacil.es/. [En línea] 2010. [Citado el: 03 de 12 de 2014.] http://www.curriculumfacil.es/.

20. NitroCV. http://nitrocv-curriculum-vitae.uptodown.com/. http://nitrocv-curriculum-vitae.uptodown.com/. [En línea] 20 de 05 de 2003. [Citado el: 03 de 12 de 2014.] http://nitrocv-curriculum-vitae.uptodown.com/.

21. Hoffman, Reid, y otros. Linkedin. Linkedin. [En línea] 2011. [Citado el: 3 de 12 de 2014.] http://www.ecured.cu/index.php/LinkedIn.

22. UCA, Universidad de Cádiz. http://biblioteca.uca.es. http://biblioteca.uca.es. [En línea] [Citado el: 26 de 04 de 2015.] http://biblioteca.uca.es/aprendizajeinvestigacion/research_gate/.

23. Samada, Lisandra Isabel Leyva. Flujo de Investigación para la Metodología. Ciudad de la Habana : s.n., 2009.

24. Escribano, Gerardo Fernández. Introducción a Extreme Programming. 2002.

25. Universidad Mayor de San Andres. Sistema académico escolar caso colegio BOLIVIANO ALEMAN AVE MARIA. [En línea] 8 de 04 de 2012. [Citado el: 08 de 12 de 2014.] http://grupo13sistemaacademicoescolar.files.wordpress.com/2012/06/marcoxp.jpg.

26. Visual-paradigm. Visual-paradigm. [En línea] [Citado el: 17 de 11 de 2014.] http://www.visual-paradigm.com/.

27. Manuel Márquez, Juan. El Lenguaje Unificado de Modelado. Resumen del libro. México : s.n., 2003.

28. UBM plc company. PR Newswire:PhpStorm 2.0: Conquiste PHP . PR Newswire:PhpStorm 2.0: Conquiste PHP . [En línea] 2014. [Citado el: 6 de 11 de 2014.] http://www.prnewswire.com/news-releases/phpstorm-20-conquiste-php-124241809.html.

29. Ecured. Enciclopedia colaborativa en la red cubana. Framework. [En línea] 2008. [Citado el: 30 de 11 de 20014.] http://www.ecured.cu/index.php/Framework.

30. Aguiluz, Javier. Symfony. Symfony. [En línea] 2013. [Citado el: 12 de 11 de 2014.] http://symfony.es/que-es-symfony.

31. Sanchez, Alvaro Fontela. OpenWebCMS: Que es bootstrap. OpenWebCMS: Que es bootstrap. [En línea] 2013. [Citado el: 28 de 11 de 2014.] http://openwebcms.es/2013/que-es-bootstrap/.

32. CAVSI. CAVSI. CAVSI. [En línea] 2014. [Citado el: 04 de 02 de 2015.] http://www.cavsi.com/preguntasrespuestas/que-es-un-sistema-gestor-de-bases-de-datos-o-sgbd/.

33. Martinez, Rafael. PostgreSQL-es. PostgreSQL-es. [En línea] 02 de 10 de 2010. [Citado el: 4 de 12 de 2014.] http://www.postgresql.org.es/sobre_postgresql.

34. Ecured. Enciclopedia colaborativa en la red cubana. pgAdmin3. [En línea] 2008. [Citado el: 5 de 12 de 2014.] http://www.ecured.cu/index.php/PgAdmin3.

35. pgAdmin. pgAdmin PostgreSQL Tool. [En línea] [Citado el: 8 de 12 de 2014.] http://www.dataprix.com/8-cliente-grafico-pgadmin3.

36. Oracle. Sitio oficial Netbeans. [En línea] 2013. [Citado el: 27 de 11 de 2014.] https://netbeans.org/features/platform/index.html.

37. Massachusetts Institute of Technology: Servidor Apache HTTP. Massachusetts Institute of Technology: Servidor Apache HTTP. [En línea] [Citado el: 14 de 11 de 2014.] http://web.mit.edu/rhel-doc/4/RH-DOCS/rhel-rg-es-4/s1-apache-config.html.

38. git. Getting-Started-About-Version-Control. 2015.

39. Baluart.net. Baluart.net. Introduccion a las pruebas unitarias en php con phpunit. [En línea] [Citado el: 28 de 4 de 2015.] www.baluart.net/articulo/introduccion-a-las-pruebas-unitarias-en-php-con-phpunit.

40. synergix.wordpress.com. synergix.wordpress.com. synergix.wordpress.com. [En línea] 10 de 07 de 2008. [Citado el: 21 de Abril de 2015.]

41. modelocurriculum.net. modelocurriculum.net. modelocurriculum.net. [En línea] 2008. [Citado el: 6 de 3 de 2015.] http://www.modelocurriculum.net/tipos-el-curriculum-combinado-o-mixto.html.

42. infoempleo.com. infoempleo.com. infoempleo.com. [En línea] [Citado el: 6 de 3 de 2015.] http://www.infoempleo.com/Consejos/Cv/CvCronologicoInverso.aspx.

43. Garzás, Javier. javiergarzas.com. javiergarzas.com. [En línea] 12 de 2011. [Citado el: 6 de 3 de 2015.]

44. Gómez, Julián. laboratorioti.com. laboratorioti.com. [En línea] 21 de 02 de 2013. [Citado el: 12 de 3 de 2015.] http://www.laboratorioti.com/2013/02/21/metodo-de-estimacion-agil-puntos-de-historia/.

45. Sommerville, Ian. Requerimientos.Requerimientos del software.Requerimientos funcionles y no funcionales, in Ingeniería se Software. p. 712. Madrid,España. : s.n., 2005.

46. sites.google.com. sites.google.com. sites.google.com. [En línea] [Citado el: 12 de 3 de 2015.] https://sites.google.com/site/xpmetodologia/marco-teorico/funcionamiento.

47. Craig Larman, Luz Maria Hernandez Rodriguez. UML y Patrones. Introducción al análisis y diseño orientado a objetos.

48. Urquizu, Pau. Business Intelligence. Claves Subrogadas. [En línea] 16 de junio de 2009. http://www.businessintelligence.info/serie-dwh/claves-subrogadas.html.

49. Tedeschi, Nicolás. https://msdn.microsoft.com/. https://msdn.microsoft.com/. [En línea] [Citado el: 12 de 3 de 2015.] https://msdn.microsoft.com/es-es/library/bb972240.aspx.

50. Gomez, Mauro. Ingenio Ds. Ingenio Ds. [En línea] 16 de 09 de 2013. [Citado el: 4 de 3 de 2015.] https://ingeniods.wordpress.com/2013/09/16/patrones-arquitectonicos/.

51. Bilbao, Reyvi. desarrolloweb.com. desarrolloweb.com. [En línea] [Citado el: 4 de 3 de 2015.] http://www.desarrolloweb.com/wiki/mvc-modelo-vista-controlador.html#.

52. Carmona, Juan García. http://juan-garcia-carmona.blogspot.com. http://juan-garcia-carmona.blogspot.com. [En línea] 7 de 09 de 2012. [Citado el: 7 de 3 de 2015.] http://juan-garcia-carmona.blogspot.com/2012/09/grasp-creador.html.

53. —. http://juan-garcia-carmona.blogspot.com. http://juan-garcia-carmona.blogspot.com. [En línea] 7 de 09 de 2012. [Citado el: 7 de 3 de 2015.] http://juan-garcia-carmona.blogspot.com/2012/09/grasp-controlador.html.

54. —. juan-garcia-carmona.blogspot.com. juan-garcia-carmona.blogspot.com. [En línea] 09 de 2012. [Citado el: 7 de 3 de 2015.] http://juan-garcia-carmona.blogspot.com/2012/09/grasp-experto-en-informacion.html.

55. Ing. Yosvani Turruelles Tejeda, Lic. Caridad Guzmán Vitón. COMPONENTES SIMULADOS PARA EL SISTEMA DE INFORMACIÓN PARA LA SALUD. La Habana : s.n., 2008.

56. martin fowler. martin fowler. [En línea] 23 de 01 de 2004. [Citado el: 14 de 03 de 2015.] http://martinfowler.com/articles/injection.html.

57. wikiuml. Wiki sobre UML. Diagrama de Componentes. [En línea] 2015. [Citado el: 24 de Abril de 2015.] wikiuml.wikispaces.com/Diagrama+de+Componentes.

58. Ecured. Enciclopedia colaborativa en la red cubana. Diagrama de despliegue. [En línea] www.ecured.cu/index.php/Diagrama_de_despliegue.

59. msdn.microsoft.com/. msdn.microsoft.com/. msdn.microsoft.com/. [En línea] msdn.microsoft.com/.

60. MilkZoft. CodeJobs. www.codejobs.biz. [En línea] Estandares de codificacion en php., 19 de Febrero de 2013. [Citado el: 29 de Abril de 2015.] www.codejobs.biz/es/blog/2013/02/19/estandares-de-codificacion-en-php-psr0-psr1-psr2-y-psr3#sthash.UWM0pfwn.dpbs.

61. Reglas y Prácticas en. Joskowicz, Ing. José.

62. A metrics suite for object oriented design. Shyam R Chidamber, Chris F Kemerer. 6, s.l. : IEEE, 6 de 1994, Software Engineering, IEEE Transactions on, Vol. 20, págs. 476-493.

63. Reifer, Donald J. Software Management. s.l. : John Wiley & Sons, 2006. ISBN-13 978-0471-77562-1.

64. TÉCNICAS DE EVALUACIÓN DE SOFTWARE. Natalia Juristo, Ana M. Moreno, Sira Vegas. 2006, Vol. 12.

65. Joskowicz, Jose. Reglas y Prácticas en eXtreme Programming. 2008.

66. Batz, Alan. Mundo Respuesta. [En línea] 25 de 3 de 2014. [Citado el: 31 de 5 de 2015.] http://www.mundorespuesta.com/como-elaborar-un-curriculum-vitae/.

67. Cepeda, Cecilia. Como Conseguir Trabajos. [En línea] 29 de 5 de 2014. [Citado el: 31 de 5 de 2015.] http://comoconseguirtrabajos.blogspot.com/2014/05/que-es-un-curriculum-vitae_29.html.

68. Domestika. Domestika. [En línea] Domestika, 2012. [Citado el: 31 de 5 de 2015.] http://www.domestika.org/es/projects/52374-diseno-de-mi-curriculum-vitae.

69. Miroglio, Lisa. Cuanto gana. Como Hacer un Curriculum Vitae. [En línea] 3 de 2 de 2015. [Citado el: 12 de 5 de 2015.] http://www.cuanto-gana.com/como-hacer-un-curriculum-vitae/.

BIBLIOGRAFÍA CONSULTADA

1. Curso completo de HTML. 2014.
2. The jQuery Foundation. Sitio Oficial: Jquery. Sitio Oficial: Jquery. [En línea] [Citado el: 10 de 12 de 2014.] http://jquery.com/.
3. Álvarez, Miguel Angel. DesarrolloWeb: Que es PHP. DesarrolloWeb: Que es PHP. [En línea] 09 de 5 de 2001. [Citado el: 7 de 12 de 2014.] http://www.desarrolloweb.com/articulos/392.php.
4. Eguíluz Pérez, Javier. Introducción a CSS. 2010.
5. —. Introducción a XHTML. 2010.
6. González-Vallés Saco, Guillermo. Una introducción a Apache. 2014.
7. Patricio Letelier, Mª Carmen Penadés. Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP). Valencia : Universidad Politécnica de Valencia, 2006.
8. Sánchez Maza, Miguel Ángel. JavaScript. s.l. : IC Editorial, 2012. ISBN: 978-8495733184.
9. W3C. HTML. W3C. [En línea] 2015. http://www.w3c.es/ .
10. http://www.cubadebate.cu/wp-content/uploads/2014/06/codigo-del-trabajo-de-la-republica-de-cuba.pdf.
11. Buenas prácticas para aplicaciones Symfony. Fabien Potencier, Ryan Weaver, Javier Eguiluz.
12. Desarrollo web ágil con Symfony2. Javier Eguiluz.
13. Identifying design patterns in the Symfony framework. Hugo Hamon. May 2nd 2014.
14. Estilos y Patrones en la Estrategia de Arquitectura de Microsoft. Carlos Reynoso – Nicolás Kicillof. Marzo de 2004.
15. Introducción a CSS. Javier Eguiluz.
16. HTML5 y CSS3. Brian P. Hogan.
17. Manual de HTML5 en español. Alejandro Castillo Cantón.
18. Reglas y Prácticas en eXtreme Programming. Ing. José Joskowicz.
19. Técnicas de evaluación de software. Natalia Juristo, Ana M. Moreno, Sira Vegas. 17 de octubre de 2006.
20. Introducción a Extreme Programming. Gerardo Fernández Escribano. 9 de diciembre de 2002.
21. Ingeniería del software. Un enfoque práctico. Séptima Edición. Roger S. Pressman, Ph.D.
22. Easyjob. Ayuda CurriculumFacil.
23. NitroCV. Ayuda NitroCV.

GLOSARIO DE TÉRMINOS

Currículum Vítae: documento en el cual se expone las experiencias laborares y profesionales de un individuo.

Currículo o CV: abreviaturas de Currículum Vítae.

Certificación de evidencias: es reconocer las competencias prácticas, individuales y sociales obtenidas por el trabajador a lo largo de la vida profesional, documentar y señalar las competencias exigidas en la realidad práctica del trabajo.

Framework (Marco de Trabajo): estructura de soporte definida mediante la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Metodología: una metodología de desarrollo de software es un conjunto de procedimientos, técnicas y actividades a seguir para en un fin convertir los requisitos de usuario en un producto de software.

HU: en el contexto de una metodología ágil, popularmente se asocia el concepto “historia de usuario” con el de “requisito funcional”.

CRC: la tarjeta CRC tiene como principal funcionalidad, convertir las clases del sistema en objeto y así cada tarjeta representa una clase.

MVC: Modelo Vista Controlador, es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

GRASP: son Patrones Generales de Software para Asignación de Responsabilidades (General Responsibility Assignment Software Patterns), aunque son considerados que más que patrones propiamente dichos pues constituyen una serie de buenas prácticas recomendables en el diseño de software.

GoF: Gang of Four hace referencia al nombre con el que comúnmente se conoce a los autores del libro Design Patterns. Estos patrones de diseño se clasifican en 3 grandes categorías basadas en su propósito: creacionales, estructurales y de comportamiento.

TOC: Tamaño Operacional de Clases, métrica para medir la calidad del diseño.

RC: Relación entre Clases, métrica para medir la calidad del diseño.