Uso de Ontologías y otras Fuentes de Datos para la Generación de Currículos Vitae.

Kenenias Bethuel Peréz-Betanzos[1], Beatriz Alejandra Olivares-Zepahua[2], Giner Alor-Hernandez[3], Ana María Chávez -Trejo[4]

[1,2,3,4] Instituto Tecnológico de Orizaba
Orizaba, Veracruz, México
[1]keneprz@acm.com, [2]bolivares@ito-depi.edu.mx,
[3]galor@ito-depi.edu.mx, [4]achavez@ito-depi.edu.mx
Paper received on 24/07/12, Accepted on 05/09/12.

Resumen. En el presente trabajo se describe un prototipo para la recuperación de información de diferentes fuentes de información (ontologías y bases de datos) y la generación de Currículos Vitae en documentos PDF. En particular, se presenta el diseño de la arquitectura mostrando su vista lógica, vista de desarrollo y los casos de uso o escenarios; se presenta también la implementación de un prototipo desarrollado en lenguaje Java que usa el marco de trabajo Java ServerFaces (JSF) para la construcción de las interfaces de usuario, la API iTextpara la generación de Currículos Vitae (CV) en documentos PDF, los lenguajes de consulta SQL y SPARQL para la consulta a las fuentes de información, Jena y JDBC para el acceso a las fuentes de información. El propósito general de este trabajo es mostrar los beneficios del uso de las ontologías en los sistemas de recuperación de información.

Palabras Clave: Web Semántica, SPARQL, SQL, Fuentes de Información, Ontologías, Bases de Datos, Java

1 Introducción

La Web Semántica es la evolución de la Web actual en la que el contenido es procesable automáticamente a escala global. Su objetivo principal es permitir, tanto a humanos como a máquinas, encontrar, compartir y combinar información de manera sencilla y automatizada, es decir, tener una Web más útil [1].

El núcleo de la Web Semántica son las ontologías, una ontología es una especificación formal y explícita de una conceptualización compartida [2]. Una ontología está formada por una taxonomía que define clases y sus relaciones, y un conjunto de reglas de inferencia que tienen un conocimiento compartido y común para cualquier dominio entre aplicaciones y personas. Entre los beneficios que se tienen en el uso de las ontologías con respecto a otras fuentes de datos como las bases de datos relacionales son interoperabilidad, búsqueda, reutilización y restructuración [3].

.A. Cortés Fernández, J. H. Sossa Azuela. (Eds.).
Advances in Intelligent and Information Technologies.
Research in Computing Science 60, 2012, pp.227 -237.

En el presente artículo se muestran la arquitectura y un prototipo de una aplicación Web para la generación de formatos de Currículos Vítae haciendo uso de una base de datos y una ontología de productividad de la Maestría en Sistemas Computacionales del Instituto Tecnológico de Orizaba, el prototipo fue desarrollado en el lenguaje de Java y usa un editor de formatos para mostrar, adecuar y guardar cada formato de Currículo Vitae para su exportación a documentos PDF.

2 Problema

Actualmente en la División de Estudios de Postgrado e Investigación (DEPI) del Instituto Tecnológico de Orizaba (ITO), los profesores de la Maestría en Sistemas Computacionales entre sus actividades administrativas necesitan actualizar constantemente sus Currículos Vitae (CV). Estos Currículos Vitae se presentan bajo distintos formatos de acuerdo a la dependencia que lo solicita como PROMEP, CONACYT y DGEST entre otras. Estas actividades de actualizar, modificar y adecuar cada formato de CV conllevan un aumento en la carga de trabajo de los profesores ya que es un distractor respecto a las tareas prioritarias propias de sus cargos.

Los formatos de CV difieren entre sí de acuerdo a la dependencia solicitante y, en ocasiones, a la plaza del investigador (tiempo completo o tiempo parcial). Sin embargo, el CV principal se mantiene en el sistema del Consejo Nacional de Ciencia y Tecnología (CONACYT) y es exportable a un documento PDF pero dicho sistema no incluye consultas sobre la información, de tal forma que recuperar el CV en un formato distinto implica necesariamente la recaptura de información.

3 Propuesta

Para dar solución al problema mencionado anteriormente, se propone aprovechar el uso de las tecnologías Web 2.0 y Web semántica, para desarrollar una aplicación Web basada en una ontología de productividad con la cuál se genera el CV de cada profesor de la Maestría en Sistemas Computacionales del ITO bajo distintos formatos exportables a documentos PDF y Word, los datos de losCV se recuperan de las siguientes fuentes de información:1) la ontología de productividad de la Maestría en Sistemas Computaciones del ITO, 2) la base de datos de Recursos Humanos del ITO y 3) una ontología que mantiene los datos de cada CV que no se encuentran incluidos en las dos fuentes antes mencionadas.

El principal propósito de este trabajo, es mostrar los beneficios del uso de las ontologías con respecto a otras fuentes de datos en los sistemas de recuperación de información, ya que como se mencionó anteriormente los formatos de CV contienen muchos datos similares que solo varían por sinónimos o posición dentro del documento o algunos datos que no son comunes en todos ellos, por lo que el uso de las ontología nos ayuda a tener una clara especificación de los datos, agregando a las propiedades sinónimos o anotaciones teniendo así más reusabilidad y fiabilidad en los datos entre otros beneficios[4].

4 Descripción de la aplicación.

El desarrollo de esta aplicación Web se basa en un modelo de desarrollo de prototipos incrementales [5], este modelo nos ayuda a evaluar el diseño y la implementación, las funcionalidades, así como tener una interacción con la aplicación para mejorar la usabilidad de cada prototipo.

En esta aplicación Web se usa una ontología de productividad de la Maestría en Sistemas Computacionales del ITO, que recupera la productividad de los profesores de la misma respecto a información de proyectos de investigación, artículos publicados y libros entre otros; esta ontología de productividad se usa como fuente de datos primaria para la aplicación, se usa también como fuente de datos la base de datos de Recursos Humanos del ITO, esta base de datos almacena los datos personales de los profesores de la Maestría en Sistemas Computacionales del ITO como nombre, dirección y RFC entre otros; para completar los demás datos faltantes de los formatos de los Currículos Vitae se desarrolló una ontología que mantiene los datos de los profesores que no forman parte ni de la productividad ni de los datos personales, tales como distinciones, estancias de investigación y participación en congresos entre otros; esta última ontología extrae sus datos de los CV de CONACYT que se encuentran en documentos PDF.

A continuación se muestra en la figura 1 una parte de las clases que se crearon para la ontología basada en los datos analizados e identificados de los distintos formatos de CV, dicha ontología se modeló por medio de la herramienta Protégé[6] con el lenguaje OWL [7] (Web Ontology Language, Lenguaje de Ontología Web).

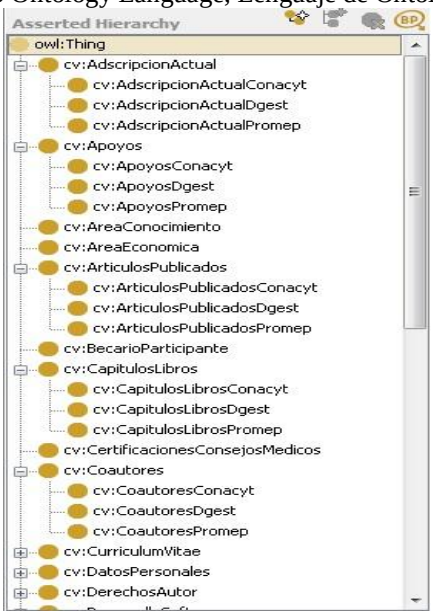

Figura 1. Estructura de las clases de la ontología basada en el CV de CONACYT.

El dominio de esta ontología consiste de los conceptos y sus relaciones con los subconceptos de los datos y secciones identificadas en los CV de CONACYT, DGEST y PROMEP. Las relaciones de los conceptos y sus subconceptos forman un árbol de la taxonomía para CV. Cada una de las clases y subclases de la ontología tienen un conjunto de propiedades para la relación entre los diferentes individuos [8].

Esta aplicación usa el lenguaje Java y el marco de trabajo Java Server Faces (JSF) [9] para construir las interfaces de usuario, la validación de la captura, control de eventos y manejo de estados entre otros.

En la figura 2 se presenta el Diagrama de Componentes, que describe los componentes de software, cada uno de los cuales proporciona una interfaz para la comunicación e interacción ente los mismos. Los componentes mostrados en este diagrama son los que hasta el momento se han identificado, conforme se avance en los prototipos es probable que surjan nuevos componentes. En este Diagrama de Componentes se tienen 8 componentes: 1) Componente CKE ditor que mantiene el control de las funcionalidades del editor de formatos para los CV, 2) Componente Elemento Edición controla el manejo de las plantillas para los formatos de los CV, 3) Componente iTextPDF controla la funcionalidad para la generación de documentos PDF, 4) Componente PrimesFaces controla el manejo y la funcionalidad para AJAX y RIAs, 5) Componente JSF BackingBeans contiene Java Beans que obtienen valores de controles e implementan métodos listener, 6) Componente ModelBeans contiene todas las clases que representan la información de cada CV, 7) Componente API Jena controla y maneja la funcionalidad para consulta, recuperación y modificación de los datos en las ontologías, 8) Componente API JDBC controla y maneja la funcionalidad para la consulta, recuperación y modificación de los datos en la base de datos.

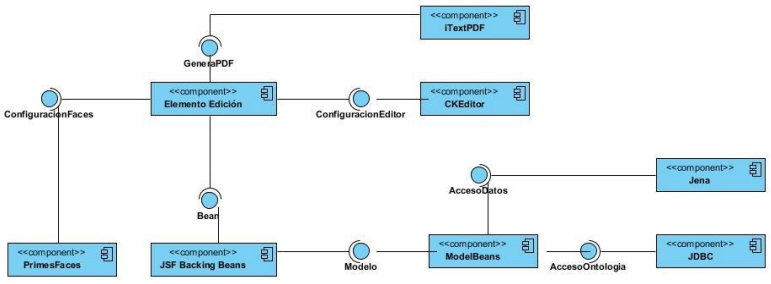

Figura 2. Vista de Desarrollo – Diagrama de Componentes.

5 Arquitectura de la aplicación.

La arquitectura de la aplicación se desarrolló bajo el modelo de 4 +1 vistas [10], este modelo ayuda a tener una relación con el diseño y la implementación de la estructura de la aplicación. En este trabajo se muestran los Casos de Uso o Escenarios que son una abstracción de los requisitos funcionales más importantes de la aplicación, la Vista Lógica que es una abstracción tomada del dominio del problema en la forma de objetos o clases de objetos y la Vista de Desarrollo que está cen-

trada en la organización de los módulos de software a desarrollar; la Vista de Procesos y la Vista Física no se incluyen en este documento debido a los cambios que experimentan con la aplicación del modelo de prototipos incrementales. A continuación se muestran las 3 vistas de la arquitectura mencionadas.

5.1 Casos de Uso o Escenarios

El diagrama de Casos de Usoen UML [11] para la aplicación se realizó con la herramienta Visual Paradigm[12], y se muestra en la figura 3. En este diagrama está incluida la funcionalidad de la aplicación en su conjunto, independientemente de la prioridad de los distintos prototipos. Existen dos actores, el administrador y el profesor, con 16 casos de uso que se describen a continuación:

1. Agregar Datos CV (Ontología) – Agrega datos de los Currículos Vitae a la Ontología, solo para el actor administrador.
2. Agregar Datos CV (BD) – Agrega datos a la base de datos de Recursos Humanos, solo para el actor administrador.
3. Modificar Plantilla CV – Modifica la plantilla de CV y guarda los cambios realizados, solo para actor administrador.
4. Deshabilitar Plantilla CV –Deshabilita la plantilla de CV, solo para el actor administrador.
5. Crear Tipo Formato CV – Crea un nuevo formato de CV, solo para el actor administrador.
6. Crear Plantilla CV – Crea una nueva plantilla de CV, solo para el actor administrador.
7. Mostrar Detalles CV –Muestra los datos del CV seleccionado por el usuario, solo para el actor administrador.
8. Modificar Tipo Formato CV – Modifica un formato de CV, solo para el actor administrador.
9. Eliminar Tipo Formato CV – Elimina un formato de CV, solo para el actor administrador.
10. Registrar Usuario – Agrega un nuevo usuario, solo para el actor administrador.
11. Modificar Usuario – Modifica los datos de un usuario y los guarda, solo para el usuario administrador.
12. Eliminar Usuario – Elimina a un usuario, solo para el actor administrador.
13. Iniciar Sesión – Ingreso a la aplicación, para el actor administrador y profesor.
14. Validar Usuario – Válida el ingreso a la aplicación para los usuarios.
15. Mostrar Detalles Usuario – Muestra los datos de cada usuario, solo para el actor profesor.
16. Generar CV – Genera el CV del usuario ingresado, solo para el actor profesor.

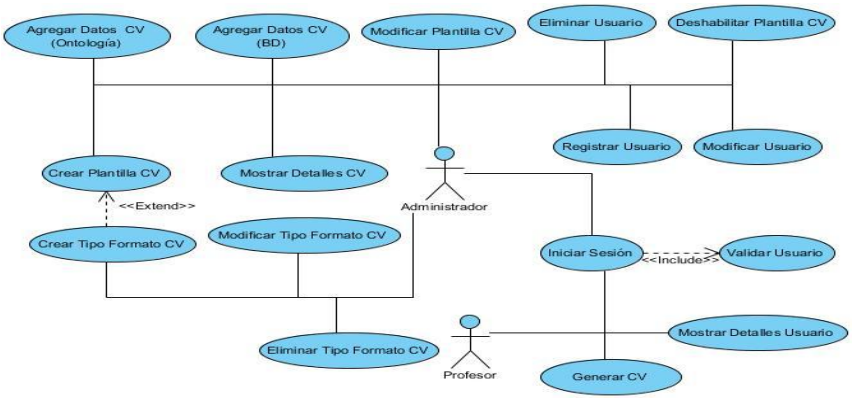

Figura 3. Diagrama de casos de uso

5.2 Vista de desarrollo

En la figura 4 se muestra la vista de desarrollo representada por un Diagrama de Paquetes que utiliza un estilo arquitectónico Modelo-Vista-Controlador [13].

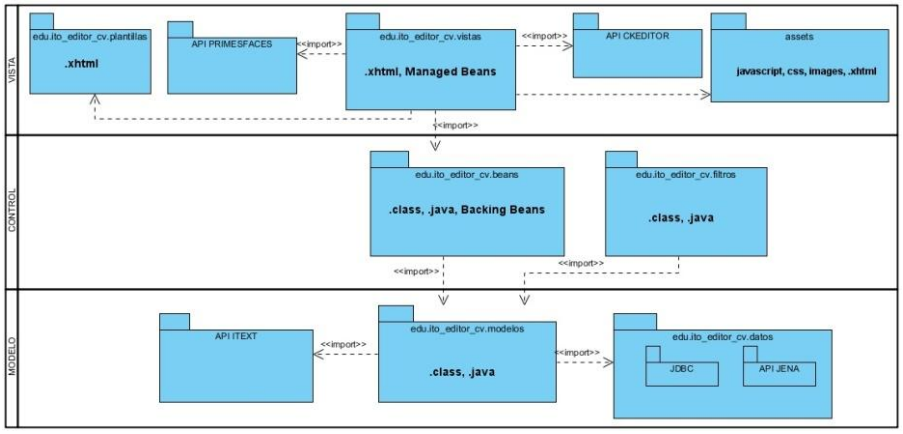

Figura 4. Vista de desarrollo – Diagrama de Paquetes

En la parte de la Vista se tienen 5 paquetes, 1) el paquete Plantillas es el encargado de mantener los formatos genéricos de CV,2) el paquete API PrimesFaces [14] contiene toda la funcionalidad para el manejo de RIAs (Aplicaciones Enriquecidas de Internet) y de AJAX (Asynchronous JavaScript And XML, JavaScript asíncrono y XML), 3) el paquete Vista contiene todas las páginas en XHTML, 4) el paquete API CKEditor [15] contiene la funcionalidad para el manejo del editor de formatos que modificará, adecuará y guardará los cambios de cada uno de los formatos de Currículos Vitae, 5) el paquete Assets almacena todos los archivos JavaScript, hojas de estilo e imágenes que se usan en la aplicación.

En la parte del Control se tienen 2 paquetes,1) el paquete Beans se encarga de la interacción entre las vistas y los modelos de la aplicación, 2) el paquete Filtros contiene las clases que validan el ingreso de los usuarios a la aplicación.

En la parte del Modelo se tiene 3 paquetes,1) el paquete API iText [16] que contiene la funcionalidad para la generación de cada CV en archivos PDF, 2) el paquete Modelos contiene todas las clases que representan la información de negocio, 3) el paquete Datos mantiene el acceso a datos con la base de datos relacional por medio de la APIJDBC [17] y el acceso a ontologías mediante la API Jena [18].

5.3. Vista Lógica

En la figura 5 se muestra la vista lógica que es representada por el Diagrama de Clases, este diagrama contiene las clases que representa la información de los CV, manejo de las plantillas, el ingreso de los usuarios, el acceso a los datos a las ontologías y base de datos, configuración y manejo del editor, la generación de documentos en formatos Word y PDF.

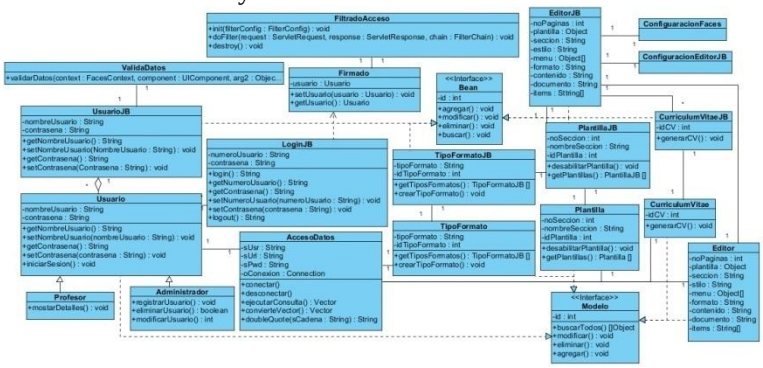

Figura 5. Vista Lógica – Diagrama de Clases

6 Prototipo.

Los prototipos a construir conforme al modelo de desarrollo de prototipos incrementales, son los siguientes: 1) Primer Prototipo – Generación de CV de prueba genérico en documentos PDF desde BD y Ontología, 2) Segundo Prototipo – Generación de CV CONACYT en documentos Word y PDF, 3) Tercer Prototipo - Generación de CV PROMEP y DGEST en documentos Word y PDF, 4) Cuarto Prototipo –Implementación de editor para formatos de los CV, 5) Quinto Prototipo – Generación de las plantillas para CV por medio del editor de formatos.

En la figura 6 se muestra el esquema general de la aplicación que se desarrollará siguiendo el modelo de prototipos incrementales mencionado. La aplicación primeramente recupera la información de las tres fuentes de datos antes mencionadas (ontologías y base de datos), estos datos recuperados son usados por un editor que los distribuye de acuerdo a los formatos de CV, maneja las distintas plantillas

para los CV, guarda los cambios hechos en cada plantilla de CV, y por último exporta el formato de CV previamente llenado hacia documentos PFD y Word.

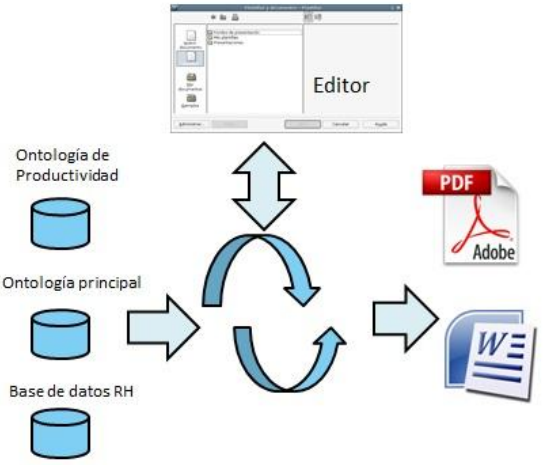

Figura 6. Esquema de la aplicación

El primer prototipo desarrollado genera un formato de CV genérico que no pertenece a los formatos oficiales de CONACYT, PROMEP o DGEST y que funciona como prueba de concepto para verificar la correcta relación entre las distintas tecnologías descritas en la Vista de Desarrollo mencionada en la sección anterior. El prototipo consiste en una pequeña interfaz que solicita el RFC del profesor y con base en él se genera el formato de CV para dicho profesor, esta interfaz se muestra en la figura 7.

Figura 7. Interfaz prototipo para la generación de CV por medio del RFC.

Con el marco de trabajo JSF se construyeron las interfaces de usuario y se manejan los eventos de la aplicación con los JSF BackingBeans, la recuperación de datos se obtiene a partir de las fuentes antes mencionadas a través de JDBC y la API Jena por medio de los lenguajes de consulta SQL[19] y SPARQL [20], los datos recuperados son gestionados por medio de los JB, con los datos recuperados de las fuentes se genera el formato de prueba de un CV genérico, con la ayuda de la API iText que realiza la exportación a documentos PDF, en la figura 8 se muestra una sección del formato de prueba del CV genérico. Actualmente se está trabajando en el desarrollo de los siguientes prototipos para incorporar las demás funcionalidades presentadas en la arquitectura de la aplicación.

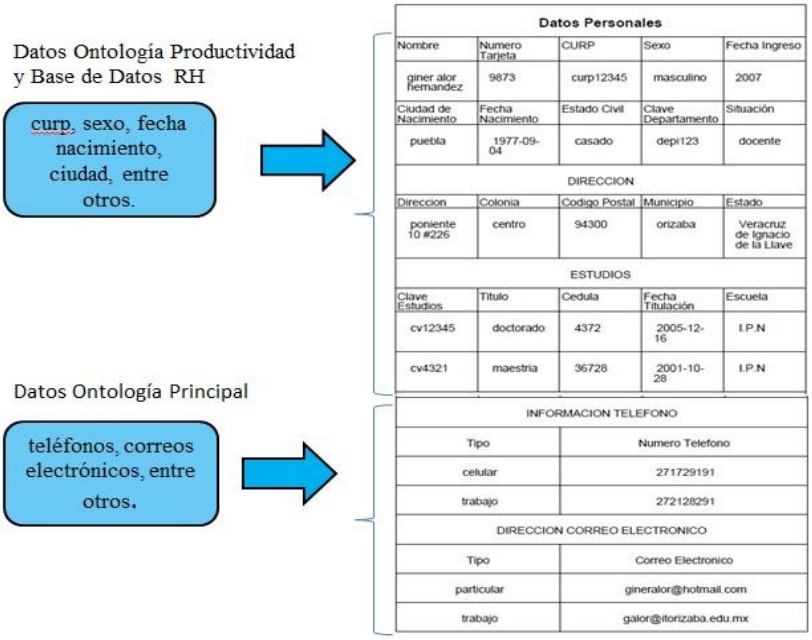

Figura 8. Formato de CV en documento PDF generado con iText

7 Conclusiones.

En este trabajo se presentó una arquitectura para una aplicación Web que genera CVs, esta arquitectura se encuentra basada en el modelo de 4 +1 vistas, cada una de estas vistas manejan los distintos requisitos funcionales y no funcionales separadamente, así como los diferentes intereses (componentes, conectores, contenedores entre otros)de los involucrados en el desarrollo como son: programadores, analistas, diseñadores, entre otros. Esta arquitectura se apoya en un desarrollo iterativo para lograr un mejor diseño, agregar nuevas funcionalidades a medida que se generan los diferentes prototipos incrementales, tener una notación común por medio de UML y manejar la escalabilidad en el desarrollo de la aplicación.

Los resultados obtenidos hasta el momento se reflejan en el desarrollo de una aplicación que está basada en un modelo de prototipos incrementales. El primer prototipo, mostrado en este trabajo, es desarrollado en el lenguaje Java con el marco de trabajo JSF, que maneja la recuperación de datos de cada una de las fuentes de información por medio de JDBC y la API de Jena a través de los lenguajes de consultas SQL y SPARQL respectivamente, los datos recuperados se usan para generar un formato de CV genérico en documentos PDF por medio de la API iText, este prototipo se usó para mostrar la interacción y la compatibilidad entre las diferentes tecnologías antes mencionadas.

8 Trabajos a futuro.

Como trabajos a futuros se tiene el complementar la arquitectura agregando la vista de procesos y la vista física, así como refinar las vistas de desarrollo y lógica con base en los hallazgos que pudieran surgir durante el desarrollo de los prototipos faltantes.

Se continuará con el desarrollo de los prototipos incrementales para agregar las siguientes funcionalidades: generar los CVs de CONACYT, DGEST y PROMEP en documentos PDF y documentos Word, desarrollar las plantillas para el manejo de cada uno de lo CVs, implementar el editor de plantillas para modificar, adecuar y guardar los cambios en cada una de las plantillas de CVs y diseñar todas las interfaces de usuario para la aplicación.

Referencias

1. Guía Breve de la Web Semántica. [En línea]. Disponible: http://www.w3c.es/divulgacion/guiasbreves/websemantica.
2. J. Contreras y J. A. Martínez Comeche.: Tutorial de Ontologías, Universidad Complutense de Madrid, España, 2001.
3. L-van Tang, Hongyan Lu, BaojunQiu, Meimei Li, Jianjun Wang, Lei Wang, Bin Zhou, Dongquing Yang, Shiwei Tang.: WISE: Prototype for Ontology Driven Development of Web Information Systems, Peking University, Beijing, China.
4. Urvi Shah, Tim Finin, Anupam Joshi.: Information Retrieval On The Semantic Web, University of Maryland, Baltimore County, EUA.
5. Jacobson, I., Booch, G., Rumbaugh, J.: El Proceso Unificado de Desarrollo, Addison Wesley, 2000
6. Horridge Matthew, KnublauchHolger, Rector Alan, Stevens Roberts.: A Practical Guide To Building OWL Ontologies Using Protégé and CO-ODE Tools Edition 1.3, The University of Manchester, March 24, 2011, EUA.
7. McGuinness Deborah L. and Harmelen Frank Van.: OWL Web Ontology Language, [En línea]. Disponible: http://www.w3.org/TR/owl-features
8. Maedche and Staab.:Ontology Learning for the Semantic Web, University of Karlsruhe, 2011.
9. "JavaServer Faces Technology - Documentation", [En línea]. Disponible: http://www.oracle.com/technetwork/java/javaee/documentation/index-137726.html
10. KruchtenPhilippe.: Planos Arquitectónicos: El Modelo de 4 + 1 Vistas de la Arquitectura del Software, Marzo 2006.
11. Booch Grady, Rumbaugh James, Jacobson Ivar.: Unified Modeling Language User Guide, 2nd Edition, Edit. Addison Wesley, 2005.
12. Visual ParadigmUser's Guide. [Enlínea]. Disponible: http://www.visual-paradigm.com/support/documents/vpumluserguide.jsp
13. Bass Len, Clements Paul, Ken Bass.: Software Architecture in Practice, Edit. Addison Wesley, December 30, 1997.
14. PrimesFaces – Documentation [En línea]. Disponible: http://primefaces.org/documentation.html
15. CKSource Docs -The Official Documentation Site [En línea]. Disponible: http://docs.cksource.com/
16. API DocumentationiText [En línea]. Disponible: http://api.itextpdf.com/
17. Oracle documentation: Getting Started with Java DB, June 19, 2012.

18. JeongDongwon, Shin Heeyoung, Baik Doo-Kwon and Jeong Young-Sik.: An Efficient Web Ontology Storage Considering Hierarchical Knowledge for Jena-based Applications,Kunsan National University, Korea, March 17, 2009.
19. K. Kevin: SQL in a Nutshell: A Desktop Quick Reference, Edit.O'Really Media
20. H. Steve: SPARQL query processing with conventional relational database systems, University of Southampton, UK.