



“Sitio web de marcadores sociales para la Universidad de las Ciencias Informáticas.”

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

Autores: Ailyn Pérez Aguila.

Yasiel Rodríguez Sánchez.

Tutores: MCs. Orlando Farray Alvarez.

Ing. Yunior Orosa Velázquez.

La Habana, junio de 2015.
“Año 57 de la Revolución”.

Declaración de Autoría

Declaración de Autoría

Declaramos ser los autores del presente trabajo de diploma y otorgamos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

Autor: Ailyn Pérez Aguila

Autor: Yasiel Rodríguez Sánchez

Tutor: Orlando Farray Alvarez

Tutor: Yunior Orosa Velázquez

*“En el pasado eras lo que tenías,
ahora eres lo que compartes.”*



Godfried Bogaard

Les dedicamos esta tesis a nuestros familiares y amigos, en especial a nuestros padres por su apoyo incondicional.

Agradecimientos

Yasiel: Agradezco a todas las personas que de una forma u otra me han ayudado y apoyado en estos cinco años, en especial a mis padres por nunca haber dudado de mí.

Ailyn: Agradezco a mi tutor por su apoyo, a mi hermano que me ha ayudado mucho en estos cinco años de la carrera, a Tania y el Ruso por su preocupación y consejos, a mi padrastro por siempre estar presente y en especial a mi madre por su apoyo incondicional.

Resumen

En la Universidad de las Ciencias Informáticas, existen sitios web que publican noticias de diferentes categorías, permitiendo mantener informada a la comunidad universitaria sobre el acontecer nacional, internacional y en particular de la UCI. Estos sitios web permiten a los usuarios votar y comentar noticias, pero no organizan las noticias por los votos y comentarios de los usuarios, ni permiten guardar o compartir una noticia con otros usuarios lo que en ocasiones le dificulta a los usuarios conocer cuáles son las noticias más votadas o comentadas del sitio. La mayoría de los sitios web existentes en la Universidad utilizan la sindicación de contenidos para resumir el contenido de las noticias que publican. El presente trabajo consiste en el desarrollo de un sitio web de marcadores sociales que contenga las noticias sindicadas por varios sitios a través de RSS y muestre de forma ordenadas las noticias. El desarrollo del software fue guiado por la metodología XP. La implementación se realizó con los lenguajes HTML5, CSS3, JavaScript y PHP, en el IDE NetBeans. UML se utilizó como lenguaje de modelado en la herramienta Visual Paradigm for UML. Fueron empleados para el desarrollo los frameworks jQuery, CodeIgniter, Bootstrap y PHPUnit. En su primera versión, el sitio permite ver las noticias de varios sitios de la UCI, ordenadas por la cantidad de votos y comentarios, autenticar un usuario y permitir que vote, comente, guarde, comparta las noticias. La investigación finaliza con las pruebas de aceptación, demostrando la correcta implementación de las funcionalidades implementadas.

Palabras clave: marcadores sociales, RSS, sindicación de contenidos.

Contenido

Introducción:	1
Capítulo 1: Fundamentación teórica	5
1.1 Web 2.0	5
1.2 Tipología de herramientas Web 2.0.....	5
1.3 Marcadores sociales	6
1.4 Sindicación de contenidos.....	7
1.5 Lectores RSS	8
1.6 Análisis de marcadores sociales	9
1.7 Metodologías de desarrollo de software	13
1.8 HTML	17
1.9 CSS.....	17
1.10 Lenguajes de programación	18
1.10.1 JavaScript.....	18
1.10.1 PHP	18
1.10.3 Java	19
1.10.4 Selección del lenguaje de programación.....	19
1.11 Herramientas de desarrollo.....	19
1.11.1 CMS.....	20
1.11.2 Frameworks.....	20
1.11.3 Selección de la herramienta de desarrollo.....	21
1.12 Librería JQuery	26
1.13 Lenguaje Unificado de Modelado.....	26
1.13.1 Herramienta CASE de modelado con UML	27
1.13.2 Visual Paradigm para UML.....	27
1.14 Sistema gestor de bases de datos.....	28
1.15 Servidor web.....	28
1.16 Balsamiq Mockups	29
1.17 NetBeans	29
1.18 PHPUnit.....	29
Conclusiones parciales.....	30
Capítulo 2: Propuesta de solución	31

2.1 Descripción de la propuesta de solución	31
2.2 Usuarios del sistema	32
2.3 Planificación del proyecto por roles	33
2.4 Modelo de dominio	34
2.5 Aspectos funcionales del sistema	35
2.6 Aspectos no funcionales del sistema.....	37
2.7 Exploración	38
2.7.1 Historias de usuarios.....	38
2.7.2 Historias de usuario de la propuesta de solución.....	40
2.7.3 Prototipo de interfaz de usuario	40
2.8 Planificación.....	41
2.8.1 Plan de Iteraciones.....	41
2.8.2 Plan de duración de las iteraciones.....	41
2.8.3 Plan de entregas.....	42
2.8.4 Plan de entregas de la propuesta de solución.....	43
2.9 Diseño.....	44
2.9.1 Patrón arquitectónico Modelo-Vista-Controlador.....	44
2.9.2 Patrones de diseño	46
2.9.3 Tarjetas CRC.....	49
Conclusiones Parciales	50
Capítulo 3. Iteraciones.....	51
3.1 Iteraciones.....	51
3.1.1 Primera iteración.....	51
3.1.2 Pruebas unitarias de la primera versión	53
3.1.3 Estrategia de codificación. Estándares y estilos a emplear.....	54
3.1.4 Librerías en CodeIgniter	57
3.1.5 Helpers en CodeIgniter.....	58
3.1.6 Pruebas de aceptación de la primera versión.....	58
3.1.7 Resultados de las pruebas de aceptación de la primera versión	60
3.1.8 Segunda iteración.....	60
3.1.9 Pruebas unitarias de la segunda iteración	63
3.1.10 Pruebas de aceptación de la segunda iteración	63
3.1.11 Resultados de las pruebas de aceptación de la segunda versión	63

3.1.12 Tercera iteración.....	63
3.1.13 Pruebas unitarias de la tercera iteración	65
3.1.14 Pruebas de aceptación de la tercera iteración	65
3.1.15 Resultados de las pruebas de aceptación de la tercera versión	65
Conclusiones parciales.....	65
Conclusiones generales:	66
Recomendaciones:	67
Referencias:	68
Anexos:	71
Anexo 1. Resumen de entrevistas con estudiantes de la UCI	71
Anexo 2. Historias de usuario.....	72
Anexo 3. Tarjetas CRC de la propuesta de solución	78
Anexo 4. Tareas de ingeniería, primera iteración	83
Anexo 5. Casos de prueba unitarias de la primera iteración.....	88
Anexo 6. Casos de prueba de aceptación, versión 0.1	93
Anexo 7. Tareas de ingeniería, segunda iteración.....	96
Anexo 8. Casos de prueba unitarias de la segunda iteración	104
Anexo 9. Casos de prueba de aceptación, versión 0.2.....	112
Anexo 10. Tareas de ingeniería, tercera iteración	117
Anexo 11. Casos de prueba unitarias de la tercera iteración.....	125
Anexo 12. Casos de prueba de aceptación, versión 0.3.....	132

Introducción:

El desarrollo constante de las tecnologías de la información y las comunicaciones (TIC) resulta incuestionable en nuestros días. Las TIC han beneficiado todas las esferas de la sociedad, permitiendo un acceso más rápido y un uso más eficiente de las informaciones que se publican en la Red.

La Web también ha sufrido cambios, evidenciándose una evolución considerable desde la aparición de la Web 1.0 hasta la llegada de la Web 2.0. La Web 1.0 se caracteriza principalmente por ser unidireccional y realizada sobre contenidos estáticos, con un carácter principalmente divulgativo, mientras que la Web 2.0 también denominada "Red social", llena Internet de blogs, wikis, foros y finalmente, redes sociales. El objetivo de la Web 2.0 es la compartición del conocimiento, es la Web colaborativa. Con la aparición de la Web 2.0 se revolucionó la forma de brindar y obtener información en Internet, creando así una arquitectura de participación colectiva caracterizada por la libertad de compartir y la comunicación entre los usuarios. (Muños Sánchez, 2007)

En Internet cada vez se tiene acceso a más información. El problema es cómo organizar y guardar las preferencias de los usuarios. Normalmente son utilizados los Favoritos (Internet Explorer) o Marcadores (Firefox). Pero estos sistemas presentan varios problemas:

- Se almacenan en el ordenador desde el que se navega y si se cambia de ordenador no se tiene acceso a ellos.
- Si el número de enlaces que se guarda es pequeño, es manejable, pero cuando la cantidad de éstos crece es impracticable.
- No es posible compartir el marcador con otros usuarios.

Todos estos problemas se resuelven guardando los sitios favoritos (o marcadores) en un servidor web y añadiéndole las componentes de la Web 2.0 (publicación, compartición, comunicación).

En la Universidad de las Ciencias Informáticas (UCI) existen sitios web informativos que publican noticias regularmente. Un usuario para mantenerse actualizado con las noticias de su interés necesita mucho tiempo de lectura si lo hace simplemente navegando, pues

ha de conectarse a un conjunto de sitios web en busca de nuevas informaciones si es que existen.

Además existen otros problemas que dificultan organizar y compartir los contenidos informativos en la Universidad:

- No existe una forma de organizar los contenidos de acuerdo a las preferencias e intereses de los usuarios.
- No existe una forma de guardar los enlaces (*links*) a sitios web que contienen noticias de interés para un usuario específico y que este pueda acceder a los enlaces a través de la Red desde cualquier ordenador conectado a Internet.
- Ninguno de los sitios web que existen actualmente en la UCI permite compartir con otro usuario un enlace a una noticia específica.

La problemática planteada anteriormente dificulta frecuentemente mantener actualizada a la comunidad universitaria pues no organizan las noticias de acuerdo a las preferencias de los usuarios, ni se pueden guardar y compartir enlaces a noticias, estas deficiencias conllevan a formular el siguiente **problema de la investigación**: ¿cómo contribuir a organizar y compartir los contenidos informativos existentes en la Red de la Universidad de las Ciencias Informáticas?

El **objeto de estudio** es el proceso de desarrollo de sitios web.

El **campo de acción** se centra en el proceso de desarrollo de sitios web de marcadores sociales.

Teniendo como **objetivo general** desarrollar un sitio web de marcadores sociales que permita contribuir a organizar y compartir los contenidos informativos existentes en la Red de la Universidad de las Ciencias Informáticas.

Tareas de la investigación:

- Identificación de los recursos usados actualmente en la UCI para organizar y compartir contenidos.
- Análisis de los fundamentos teóricos y conceptuales de los recursos que permiten organizar y compartir contenidos.

- Selección de las tecnologías y herramientas a utilizar para organizar y compartir contenidos informativos.

Para dar cumplimiento a las tareas planteadas se emplearon los siguientes **objetivos específicos**:

- Realizar la fundamentación teórica de la investigación.
- Identificar los requerimientos que debe poseer la propuesta de solución.
- Implementar sitio web de marcadores sociales para la UCI.
- Realizar pruebas de *software* a la propuesta de solución.

Como **hipótesis** se plantea que: si se desarrolla un sitio web de marcadores sociales entonces se podrán organizar y compartir los contenidos informativos existentes en la Red de la Universidad de las Ciencias Informáticas.

Métodos científicos:

Métodos teóricos:

El método analítico - sintético se utilizó para buscar los elementos más importantes que se relacionan con la investigación mediante el análisis de las bibliografías, permitiendo tomar las mejores prácticas de las soluciones similares de sitios de marcadores sociales e identificar las características de la propuesta de solución, además de seleccionar las herramientas y tecnologías que mejor se adaptan para desarrollar la aplicación.

El método histórico - lógico se utilizó para verificar cómo ha ido evolucionado con el paso del tiempo y con el surgimiento de nuevas tecnologías, herramientas y recursos el desarrollo de sitios web informativos y en particular los sitios de marcadores sociales para seleccionar la estructura y funcionalidades de la propuesta de solución.

Métodos empíricos

Observación, usado con el propósito de obtener información sobre las dificultades que se identifican acerca de los sitios web informativos que existen actualmente en la Universidad y además para conocer cuáles son las necesidades existentes, para tenerlas en cuenta al desarrollar el sitio web.

Entrevista, usado para conocer las opiniones de usuarios de la UCI sobre los sitios web, la forma en que organizan y muestran sus artículos y conocer qué se debe mejorar, según sus criterios, para desarrollar un sitio con mayor calidad y que pueda satisfacer las expectativas de la comunidad universitaria. El resumen de la entrevista realizada a los estudiantes de la UCI se encuentra en el [Anexo 1](#).

La presente investigación está conformada por tres capítulos, los mismos quedan definidos de la siguiente manera:

Capítulo 1: Fundamentación teórica

Incluye las tendencias actuales del desarrollo web, principalmente las características de los sitios de marcadores sociales junto con un estudio de sitios similares para determinar la homología con la propuesta de solución. Se analizan herramientas y metodologías, fundamentando la selección para el futuro desarrollo.

Capítulo 2: Propuesta de solución

Describe la propuesta de solución. Abarca las fases de exploración y planificación de la metodología de desarrollo XP. Se generan las Historias de usuario, el plan de iteraciones y el plan de entrega, quedando definida la cantidad de entregas e iteraciones a realizar, cuánto durará cada iteración y cuándo el cliente obtendrá cada entregable del producto. Además describe la arquitectura del sistema, con los patrones aplicados a la solución.

Capítulo 3: Iteraciones

Se generan las tareas de ingeniería en la etapa de iteraciones de la metodología XP y se realiza el proceso de pruebas generando los casos de pruebas necesarios.

Capítulo 1: Fundamentación teórica

Capítulo 1: Fundamentación teórica

La aparición de la Web 2.0 trajo consigo nuevos conceptos o recursos como la sindicación de contenidos y los marcadores sociales (*Bookmarking*) para organizar y compartir las noticias que se publican en diferentes sitios web.

Estudios realizados por el equipo de desarrollo muestran que muchos sitios web utilizan estos recursos como vía para mantener actualizados a los usuarios, proporcionando un espacio que contiene las noticias que se han ido publicando en sitios específicos, tomando en cuenta las preferencias de los usuarios para organizar de forma democrática los contenidos del sitio web. Estos sitios permiten además encontrar noticias en tiempo real, compartir y recomendar cualquier artículo que un usuario considere interesante, votar, dar su opinión, y además gracias a su rápida indexación, también beneficia al sitio en estrategias de posicionamiento web.

1.1 Web 2.0

La Web 2.0 es la evolución de la Web donde los usuarios dejan de ser pasivos para convertirse en usuarios activos, que participan y contribuyen en el contenido de la Red, formando parte de una sociedad que se informa, se comunica y genera conocimiento.

La Web 2.0 es *“un fenómeno social en relación con la creación y distribución de contenidos en Internet, caracterizado por la comunicación abierta, la descentralización de autoridad, la libertad para compartir y usar, dentro de un enfoque que trata a las relaciones humanas y económicas como conversaciones”*. (Ortiz de Zarate Tercero, 2008)

Cuando las páginas ofrecen interacción y se actualizan con los aportes de los usuarios, se habla de Web 2.0. Ésta cambió la relación del usuario con la información pues lo hizo parte de ella, permitiendo compartir y comentar cualquier noticia en la Red.

1.2 Tipología de herramientas Web 2.0

Actualmente existen gran cantidad de recursos en Internet por lo que se ha hecho necesario organizar estos recursos para orientar al usuario en el momento que necesite hacer una selección de las herramientas disponibles.

Capítulo 1: Fundamentación teórica

Con el objetivo de ofrecer una selección de aplicaciones, basadas en las relaciones sociales, que es la característica distintiva de la Web 2.0, en el libro "Planeta Web 2.0. Inteligencia colectiva o medios fast food" de Cristóbal Cobo Romaní y Hugo Pardo Kuklinski, se presenta la siguiente tipología. (Cobo Romaní, et al., 2007)

- *Social Networking*: describe todas aquellas herramientas diseñadas para la creación de espacios que promuevan o faciliten la conformación de comunidades e instancias de intercambio social. Ejemplos: MySpace y Facebook.
- *Contenidos*: hace referencia a aquellas herramientas que favorecen la lectura y escritura en línea, así como su distribución e intercambio. A esta clase pertenecerían los blogs, wikis, procesadores de texto, hojas de cálculo, calendarios en línea y las plataformas para almacenar, publicar, compartir y editar fotografías, vídeos y presentaciones digitales. Ejemplos: wordpress.org y drupal.org
- *Organización social e inteligente de la información*: herramientas y recursos para etiquetar, syndicar e indexar, que facilitan el orden y almacenamiento de la información. En esta categoría se incluirían los buscadores, los lectores de RSS y los marcadores sociales. Ejemplos: del.icio.us, Reddit y Digg.
- *Aplicaciones y servicios*: dentro de esta clasificación se incluyen las herramientas, software, plataformas en línea y cualquier recurso creado para ofrecer servicios de valor añadido al usuario final. Ejemplos: heart.google.com, gmail.google.com y zimbra.com

Para el desarrollo de la propuesta de solución la clasificación de herramientas Web 2.0 que mejor se adapta es la organización social e inteligente de la información pues incluye herramientas que permiten organizar y compartir las informaciones, entre las que se encuentran los marcadores sociales y los lectores de RSS, los cuales permitirán que el sitio web que se pretende desarrollar brinde a los usuarios las opciones de compartir y organizar los artículos que se publican en la UCI favoreciendo el acceso a las noticias más relevantes e interesantes de acuerdo a las preferencias de los usuarios.

1.3 Marcadores sociales

Los marcadores sociales son servicios en línea que permiten almacenar de forma organizada, por categorías o etiquetas, enlaces web y recuperarlos posteriormente.

Capítulo 1: Fundamentación teórica

Similares a la opción de “favoritos” de los navegadores web pero con la ventaja de que la información está accesible desde cualquier ordenador con conexión a Internet. Se pueden compartir con otros usuarios y son recursos interesantes para localizar sitios web sobre un determinado tema marcado por otros usuarios. (Muñoz, 2007,2008)

Los marcadores sociales son administradores de recursos creados para almacenar, etiquetar, organizar y compartir colectivamente el contenido más relevante de la Red. (Muños Sánchez, 2007)

Un sitio web de marcación social permite que el usuario almacene sus marcadores en Internet y los haga accesibles desde cualquier parte. Existen sitios de marcadores sociales que utilizan una estrategia organizacional conocida como *tagging* (etiquetado).

Cuando muchas personas utilizan un sitio de marcadores sociales, sus etiquetas parecen ser repetidas por otras personas que usan los mismos términos claves para describir sus marcadores. Esto permite que los usuarios del sitio encuentren otros sitios web que han sido etiquetados de la misma manera o que el usuario pueda buscar en los marcadores de otra persona con tal de que permanezcan públicos.

Los sitios de marcadores sociales utilizan como herramienta para recopilar y guardar las informaciones que publican otros sitios web, los lectores RSS, los cuales extraen las noticias sindicadas por RSS que genera un sitio web específico y luego muestran los contenidos en su sitio web.

1.4 Sindicación de contenidos

La sindicación de contenidos es el proceso mediante el cual un productor o distribuidor de contenidos proporciona información en formato digital. Permite a un sitio web, principalmente a aquellos que tienen actualizaciones frecuentes, informar las novedades a los interesados, típicamente por medio de la distribución de titulares de noticias y contenidos breves. Cuando el usuario observa un elemento de interés en su canal, puede seguir el enlace para obtener el contenido completo. (Holzner, 2006)

El consumo de sindicación de contenidos es usado para facilitar el acceso a la información web que se actualiza regularmente, permitiendo que los usuarios posean en el momento, las últimas noticias de temas de su interés.

Capítulo 1: Fundamentación teórica

Existen dos diferentes formas de distribuir o syndicar los contenidos, es decir, de generar archivos o *feeds* con las actualizaciones de la página web: RSS y RDF o Atom.

RSS

RSS es una sigla inglesa que significa "*Rich Site Summary*" o Resumen Enriquecido de Sitio Web aunque a veces se define como "*Really Simple Syndication*" o Sindicación Realmente Simple. RSS es un formato XML (*Extensible Markup Language* o Lenguaje de Marcado Extensible) que usa el RDF (*Resource Description Framework* o Marco de descripción de recursos) para representar información sobre recursos de Internet. (D'Souza, 2015)

RDF o Atom

Atom es otro formato basado en la tecnología XML. Es el sublenguaje de XML empleado para distribuir o syndicar los contenidos de Internet, sean noticias o entradas de los *weblogs*. Estos programas se conectan a la Web automáticamente, buscan informaciones nuevas y, si encuentran, notifican de ello al usuario y le presentan una lista de enlaces con los titulares y resúmenes breves de las noticias. (D'Souza, 2015)

Atom surge como una alternativa a RSS que buscaba un nuevo formato que aclarara las ambigüedades de RSS, que consolidara sus múltiples versiones, que aumentara sus capacidades y que estuviera además auspiciado por una organización de estándares. Como diferencias más notables frente a RSS Atom presenta el uso de campos distintos para el resumen y el contenido, una mejor integración con el estándar XML (incluyendo un esquema, un *namespace* y siendo más estricto con la normalización). (Lang, 2013)

1.5 Lectores RSS

Un lector RSS es aquel programa que permite a un usuario suscribirse los RSS de sus páginas web o blogs favoritos para recibir los artículos y contenidos que son de su interés. (Sardá, 2015)

Los lectores RSS reúnen, en un solo lugar, los titulares de las páginas web a las que se ha suscrito el usuario. En función del lector RSS elegido, el programa o lector *online* ofrecerá al usuario la opción de organizar la información por carpetas o por categoría según sus preferencias o intereses.

Capítulo 1: Fundamentación teórica

1.6 Análisis de marcadores sociales

En la actualidad existen diversidad de sitios web informativos, en esta investigación la búsqueda se reduce a los que hagan uso de los marcadores sociales. Los sitios web de marcadores sociales son aquellos sitios que consumen los RSS de otros sitios, muestran las noticias en un sitio central y permiten a los usuarios guardar noticias como marcadores y compartir las noticias marcadas. Se analizaron varios sistemas en el ámbito nacional e internacional para determinar características y funcionalidades de la propuesta a desarrollar.

Marcadores sociales

Nacionales:

En la UCI se han desarrollado varios sitios que consumen servicios de RSS, ejemplo de ellos son: PrimaNews, Actualízate y 4x4.

PrimaNews, sitio de consumo de noticias que brinda sindicación y suscripción a canales RSS, realiza el consumo solo de los sitios de la UCI y separa las noticias por canales RSS.

Actualízate, servicio de consumo RSS que brinda la posibilidad de mantener actualizado a la comunidad universitaria de las noticias de interés y permite al usuario configurar una página personal, votar y comentar las noticias. Brinda al usuario la posibilidad de agregar sus canales RSS, marcar las fuentes UCI y de otros sitios, de las que desee ver las noticias contenidas en ellas y visualizar esta información en las vistas Mis RSS, UCI y Otros.

4x4, sitio de consumo RSS que muestra las noticias de varios sitios de la UCI, permitiendo votar una noticia, muestra las noticias similares, la cantidad de lecturas y votos y un vínculo al sitio web de donde se consume cada noticia.

Ninguno de los sitios web antes mencionados ordena las noticias de acuerdo a la cantidad de votos y comentarios, ni permiten guardar y compartir noticias. Además todos estos sitios se encuentran actualmente fuera de servicio.

Capítulo 1: Fundamentación teórica

Internacionales:

Del.icio.us.com: Delicious es un servicio de gestión de marcadores sociales que consume RSS de varios sitios internacionales. Permite agregar los marcadores que clásicamente se guardaban en los navegadores y categorizarlos con un sistema de etiquetado denominado folcsonomías. No sólo puede almacenar sitios web, sino que también permite compartirlos con otros usuarios de del.icio.us y determinar cuántos tienen un determinado enlace guardado en sus marcadores. (<https://delicious.com/>)

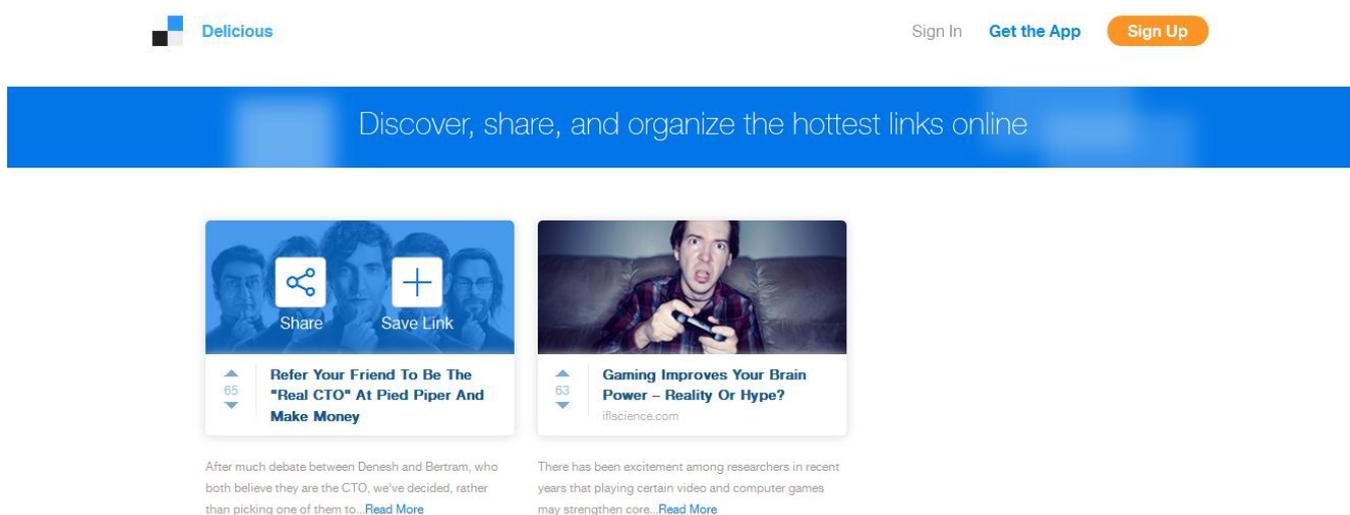
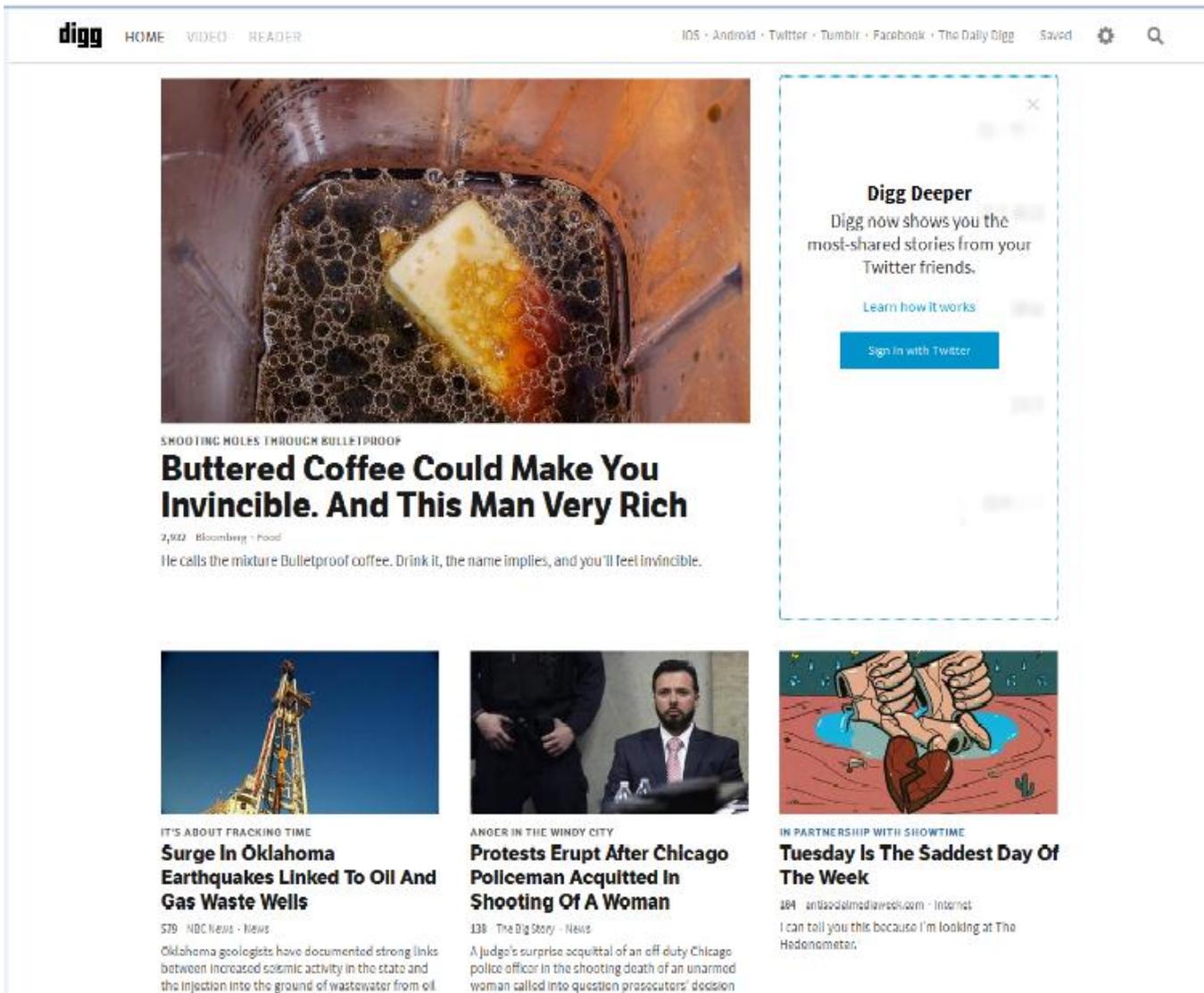


Ilustración 1: Delicious.com

Digg.com: Digg es un sitio web principalmente sobre noticias de ciencia y tecnología. Combina marcadores sociales y sindicación de RSS de otros sitios web internacionales con una organización sin jerarquías, con control editorial democrático, lo cual permite que se publiquen artículos sobre una gran variedad de géneros. Los usuarios envían relatos de noticias y recomendaciones de páginas web y los ponen a disposición de la comunidad, quien las juzga, se caracteriza por organizar las noticias según la opinión de los usuarios. Permite suscribirse a un boletín diario de noticias, brinda la opción de votar por la noticia, salvar el enlace y compartirlo en Facebook o Twitter. (<http://digg.com/>)

Capítulo 1: Fundamentación teórica



The screenshot shows the Digg.com homepage. At the top, there is a navigation bar with the 'digg' logo, 'HOME', 'VIDEO', and 'READER' links. On the right, there are links for 'iOS', 'Android', 'Twitter', 'Tumblr', 'Facebook', 'The Daily Digg', and 'Saved', along with a search icon. The main content area features a large article titled 'Buttered Coffee Could Make You Invincible. And This Man Very Rich' with a sub-headline 'SHOOTING HOLES THROUGH BULLETPROOF'. Below the main article are three smaller article thumbnails: 'Surge In Oklahoma Earthquakes Linked To Oil And Gas Waste Wells', 'Protests Erupt After Chicago Policeman Acquitted In Shooting Of A Woman', and 'Tuesday Is The Saddest Day Of The Week'. A sidebar on the right contains a 'Digg Deeper' section with a 'Sign in with Twitter' button.

Ilustración 2: Digg.com

Reddit.com: Reddit es un sitio web de marcadores sociales y agregador de noticias que consume RSS de varios sitios web internacionales y donde los usuarios pueden dejar enlaces a contenidos web. Otros usuarios pueden votar a favor o en contra de los enlaces, haciendo que aparezcan más o menos destacados. Permite enviar un nuevo post de texto o un nuevo enlace para que el sitio tome sus noticias e informaciones, brinda la opción de suscribirse o desuscribirse de las categorías o etiquetas en las que divide sus noticias para que solo se muestren las noticias de las categorías que desee el usuario. Divide las noticias en activos, nuevos, popular, subiendo, polémico, con gold, wiki, patrocinados. (<http://www.reddit.com/>)

Capítulo 1: Fundamentación teórica



The image is a screenshot of the Reddit.com homepage. At the top, there is a navigation bar with various subreddits like 'PORTADA', 'TODOS', 'ALEATORIO', etc. Below this is the Reddit logo and a search bar. The main content area displays a list of posts. The first post is titled 'Alcoholics aren't just screwing their family, but their liver too...' and has 4 comments. The second post is titled 'Scientists bred extremely sexually attractive male mosquitoes whose offspring are unable to breed...' and has 1389 comments. The third post is titled 'This is the first time I have ever had someone tattoo one of my illustrations on them...' and has 373 comments. On the right side, there is a login form with fields for 'nombre de usuario' and 'contraseña', and buttons for 'recordarme', 'reseteo contraseña', and 'iniciar sesión'. Below the login form are two buttons: 'Enviar un nuevo enlace' and 'Enviar un nuevo post de texto'. At the bottom right, there is a red octagonal 'STOP' sign.

Ilustración 3: Reddit.com

Resultado del análisis de los marcadores sociales

Luego de analizar las tendencias de desarrollo en los sitios web estudiados, que utilizan los marcadores sociales para guardar y compartir enlaces, ninguno gestiona información de las noticias e informaciones de Cuba y en específico, centrándose en las informaciones y noticias que son necesarias que sean conocidas, valoradas y compartidas por la comunidad universitaria. Estos sitios web de marcadores sociales aportan una serie de elementos significativos para la futura propuesta de solución. Los sistemas analizados concuerdan con el uso de lectores RSS para guardar y mostrar las noticias e informaciones de sitios específicos. Una característica importante que comparten todos estos sitios es la forma de organizar las noticias en dependencia de la cantidad de votos positivos de los usuarios. Para garantizar el intercambio entre usuarios y aprovechar la inteligencia colectiva se habilita la opción de comentar las noticias y tener en cuenta la cantidad de comentarios recibidos por una noticia para destacarla sobre las demás. Se concluye que la vinculación de las características mencionadas, con las noticias e informaciones que se encuentran distribuidas en los sitios existentes en la UCI, brindará un espacio común de encuentro donde el usuario podrá interactuar con el contenido, contribuirá a la organización del sitio con sus votos y comentarios, además de que podrá guardar y compartir enlaces de noticias de interés

Capítulo 1: Fundamentación teórica

para toda la comunidad universitaria.

1.7 Metodologías de desarrollo de software

Existen numerosas propuestas metodológicas que inciden en distintas dimensiones del proceso de desarrollo de software. Por una parte están las propuestas tradicionales que se centran especialmente en el control del proceso, estableciendo rigurosamente las actividades involucradas, los artefactos que se deben producir, y las herramientas y notaciones que se usarán. Por otro lado se encuentran las propuestas ágiles, las cuales se centran en el factor humano y el producto software. Esta es la filosofía de las metodologías ágiles, las cuales dan mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas. Este enfoque se adapta a proyectos con requisitos muy cambiantes y cuando se exige reducir drásticamente los tiempos de desarrollo pero manteniendo una alta calidad. (Condori Palomeque, et al., 2015)

Entre las metodologías robustas se encuentran: MSF (por sus siglas en inglés, *Microsoft Solution Framework*), RUP (por sus siglas en inglés, *Rational Unified Process*), entre otras. Estas metodologías incluyen largas fases para la especificación de requisitos, análisis y diseño, haciendo la corrección de errores muy costosa por ser metodologías poco flexibles a los cambios, debido a que el cliente y el equipo de desarrollo no mantienen una relación directa, no siendo el caso de la presente investigación. (Condori Palomeque, et al., 2015)

Las metodologías ágiles están definidas para el desarrollo de proyectos con requisitos poco definidos o cambiantes. Son aplicables para equipos de trabajo pequeños resolviendo problemas concretos, dividen el trabajo en iteraciones minimizando los fallos y los costes. Se puede hacer mención dentro de las metodologías ágiles a: XP (*Extreme Programming*), Scrum, *Crystal Methodologies*, CMM (*Capability Maturity Model*), ASD (*Adaptive Software Development*), DSDM (*Dynamic Systems Development Method*), FDD (*Feature Driven Development*), LD (*Lean Development*) entre otras.

Tabla 1: Comparación entre metodologías ágiles y tradicionales. (Cortés, 2015)

Metodología ágil	Metodología tradicional
Pocos artefactos. El modelado es prescindible, modelos desechables	Más artefactos. El modelado es esencial, mantenimiento de modelos

Capítulo 1: Fundamentación teórica

Pocos roles, mas genéricos y flexibles	Mas roles, más específicos
No existe un contrato tradicional, debe ser bastante flexible	Existe un contrato prefijado
Cliente es parte del equipo de desarrollo	El cliente interactúa con el equipo de desarrollo mediante reuniones
Orientada a proyectos pequeños Corta duración o entregas frecuentes, equipos pequeños(< 10 integrantes) y trabajando en el mismo sitio	Aplicables a proyectos de cualquier tamaño, pero suelen ser especialmente efectivas/usadas en proyectos grandes y con equipos posiblemente dispersos
La arquitectura se va definiendo y mejorando a lo largo del proyecto	Se promueve que la arquitectura se defina tempranamente en el proyecto
Énfasis en los aspectos humanos: el individuo y el trabajo en equipo	Énfasis en la definición del proceso: roles, actividades y artefactos
Basadas en heurísticas provenientes de prácticas de producción de código	Basadas en normas proveniente de estándares seguidos por el entorno de desarrollo
Se esperan cambios durante el proyecto	Se esperan que no ocurran cambios de gran impacto durante el proyecto

Después de realizar un análisis de la tabla 1 y por las razones antes mencionadas se decide desechar las metodologías tradicionales pues no se corresponden con las características del software a desarrollar.

Tabla 2: Ranking de agilidad. (Cortés, 2015)

Factor	XP	Scrum	Crystal	CMM	ASD	DSDM	FDD	LD
Sistema cambiante	5	5	4	1	5	3	3	4
Colaboración	5	5	5	2	5	4	4	4
Resultados	5	5	5	2	5	4	4	4
Simplicidad	5	5	4	1	4	3	5	3
Adaptabilidad	3	4	5	2	5	3	3	4
Excelencia técnica	4	3	3	4	3	4	4	4

Capítulo 1: Fundamentación teórica

Prácticas de colaboración	5	4	5	2	5	4	3	3
---------------------------	---	---	---	---	---	---	---	---

Nota: Los valores más altos representan una mayor “agilidad”, con grado de satisfacción de 5 puntos.

Para el desarrollo de esta investigación se analizaron las metodologías SCRUM y XP por ser las más ágiles (Tabla 1), adaptarse a las características del equipo de desarrollo y estar indicadas especialmente a proyectos con requisitos cambiantes.

SCRUM

Define un marco para la gestión de proyectos, que se ha utilizado con éxito durante los últimos años. Está especialmente indicada para proyectos con un rápido cambio de requisitos. Sus principales características se pueden resumir en dos. El desarrollo de software se realiza mediante iteraciones, denominadas *sprints*, con una duración de 30 días. El resultado de cada *sprint* es un incremento ejecutable que se muestra al cliente. La segunda característica importante son las reuniones a lo largo proyecto, entre ellas destaca la reunión diaria de 15 minutos del equipo de desarrollo para coordinación e integración. (Canós, et al., 2015)

XP (eXtremming Programming)

XP es una metodología ágil, centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo del software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en la realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico. (Canós, et al., 2015)

Fases de XP:

- Exploración.
- Planificación.
- Iteraciones.

Capítulo 1: Fundamentación teórica

- Producción.
- Mantenimiento.
- Muerte del proyecto.

Prácticas XP:

- Entregas pequeñas: la idea es producir rápidamente versiones del sistema que sean operativas, aunque obviamente no cuenten con toda la funcionalidad pretendida para el sistema. Una entrega no debería tardar más 3 meses.
- Diseño simple: se debe diseñar la solución más simple que pueda funcionar y ser implementada en un momento determinado del proyecto. La complejidad innecesaria y el código extra debe ser removido inmediatamente.
- Pruebas: la producción de código está dirigida por las pruebas unitarias. Las pruebas unitarias son establecidas antes de escribir el código y son ejecutadas constantemente ante cada modificación del sistema. Los clientes escriben las pruebas funcionales para cada historia de usuario que deba validarse.
- Programación en parejas: toda la producción de código debe realizarse con trabajo en parejas de programadores. Las principales ventajas de introducir este estilo de programación son: muchos errores son detectados conforme son introducidos en el código (inspecciones de código continuas), por consiguiente la tasa de errores del producto final es más baja, los diseños son mejores y el tamaño del código menor (continua discusión de ideas de los programadores), los problemas de programación se resuelven más rápido, se posibilita la transferencia de conocimientos de programación entre los miembros del equipo, varias personas entienden las diferentes partes sistema, los programadores conversan mejorando así el flujo de información y la dinámica del equipo.
- Propiedad colectiva del código: cualquier programador puede cambiar cualquier parte del código en cualquier momento. Esta práctica motiva a todos a contribuir con nuevas ideas en todos los segmentos del sistema, evitando a la vez que algún programador sea imprescindible para realizar cambios en alguna porción de código.
- Cliente *in-situ*: el cliente tiene que estar presente y disponible todo el tiempo para el equipo. Gran parte del éxito del proyecto XP se debe a que es el cliente quien conduce constantemente el trabajo hacia lo que aportará mayor valor de negocio

Capítulo 1: Fundamentación teórica

y los programadores pueden resolver de manera inmediata cualquier duda asociada.

1.8 HTML

HTML (*Hyper Text Markup Language*) es un lenguaje que sirve para escribir hipertexto, es decir, documentos de texto, presentado de forma estructurada, con enlaces (*links*) que conducen a otros documentos o a otras fuentes de información, que pueden estar en la máquina del usuario o en máquinas remotas de la Red. (Gauchat, 2012)

El HTML5 es la quinta revisión del lenguaje de programación “básico” de la *World Wide Web*, el HTML. Esta nueva versión pretende remplazar al actual (X)HTML, corrigiendo problemas con los que los desarrolladores web se encuentran, así como rediseñar el código actualizándolo a nuevas necesidades que demanda la Web de hoy en día. HTML5 permite crear nuevas etiquetas, atributos y eliminar aquellas marcas que están en desuso o se utilizan inadecuadamente. (Castillo Cantón, 2015)

1.9 CSS

Hojas de Estilo en Cascada (*Cascading Style Sheets*), es un mecanismo simple que describe cómo se va a mostrar un documento en la pantalla o cómo se va a imprimir. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos.

CSS se utiliza para dar estilo a documentos HTML y XML, separando el contenido de la presentación. Los estilos CSS definen la forma de mostrar los elementos HTML y XML. Permite a los desarrolladores web controlar el estilo y el formato de múltiples páginas web al mismo tiempo. Cualquier cambio en el estilo marcado para un elemento en CSS afectará a todas las páginas vinculadas a esa CSS en las que aparezca ese elemento. (Gauchat, 2012)

La última versión del lenguaje CSS es la versión 3 (CSS3), la cual ha incorporado nuevas características (Gauchat, 2012)

- Bordes redondeados y decorados.
- Presentación de un texto en varias columnas.
- Sombra aplicadas al texto y al resto de elementos.

Capítulo 1: Fundamentación teórica

- Los elementos gráficos como los filtros, las máscaras, el efecto espejo o a fusión de imágenes.

1.10 Lenguajes de programación

Un lenguaje de programación es un conjunto de reglas, notaciones, símbolos y/o caracteres que permiten a un programador poder expresar el procesamiento de datos y sus estructuras en la computadora. Cada lenguaje posee sus propias sintaxis. Existen varios lenguajes de programación entre los que se encuentran JavaScript, PHP y Java. (Instituto Tecnológico de Celaya, 2014)

1.10.1 JavaScript

JavaScript (a veces abreviado como JS) es un lenguaje ligero, orientado a objetos, más conocido como el lenguaje de script para páginas web. Es un lenguaje basado en prototipos, dinámico y soporta estilos de programación orientada a objetos. (Gauchat, 2012)

JavaScript es lo que se conoce como lenguaje script, es decir, se trata de código de programación que se inserta dentro de un documento. Fue desarrollado por la empresa Netscape con la idea de potenciar la creación de páginas web dinámicas para su navegador *Navigator*.

JavaScript (en contra de lo que se podría suponer) es totalmente distinto de Java. Java crea programas totalmente independientes y operativos; mientras que JavaScript es más sencillo porque lo único que permite es insertar código especial dentro del HTML. La ventaja fundamental de JavaScript es que su aprendizaje y uso son muy sencillos. (Gauchat, 2012)

1.10.1 PHP

PHP (*Hypertext Preprocessor*) es un lenguaje de código abierto especialmente adecuado para el desarrollo web. Lo que distingue a PHP es que el código es ejecutado en el servidor, generando HTML y enviándolo al cliente. Es simple para el principiante pero también ofrece muchas características avanzadas para los programadores profesionales. (Maraboli Rosselott, 2003)

Capítulo 1: Fundamentación teórica

PHP puede emplearse en todos los sistemas operativos principales, admite la mayoría de servidores web de hoy en día, por lo que con PHP, se tiene la libertad de elegir el sistema operativo y el servidor web. Además, se tiene la posibilidad de utilizar programación orientada a objetos (POO). (Maraboli Rosselott, 2003)

PHP es un lenguaje de programación de estilo clásico, es decir que es un lenguaje de programación con variables, sentencias condicionales, bucles, funciones. No es un lenguaje de etiquetas como podría ser HTML, XML o WML. (Maraboli Rosselott, 2003)

1.10.3 Java

Java es un lenguaje compilado e interpretado. Todo programa en Java ha de compilarse y el código que se genera es interpretado por una máquina virtual. De este modo se consigue la independencia de la máquina. Java es un lenguaje orientado a objetos de propósito general, se puede utilizar para construir cualquier tipo de proyecto. En el diseño de Java se prestó especial atención a la seguridad. (Belmonte Fernández, 2005)

Es un lenguaje totalmente independiente de la plataforma y orientado a objetos. Además es fácil para crear aplicaciones para redes TCP/IP. El código Java no se compila, sino que se precompila, de tal forma que se crea un código intermedio que no es ejecutable. Para ejecutarle hace falta pasarle por un intérprete que va ejecutando cada línea. Ese intérprete suele ser la máquina virtual de Java. (Sánchez, Manual de Java, 2004)

1.10.4 Selección del lenguaje de programación

Después de realizar un análisis de los lenguajes de desarrollo descritos anteriormente se decide escoger JavaScript y PHP para desarrollar la propuesta de solución pues son los lenguajes que mejor domina el equipo de desarrollo, siendo la experiencia de los programadores una de las bases de la metodología seleccionada: XP.

La versión del lenguaje PHP a utilizar para el desarrollo de la propuesta de solución es la 5.

1.11 Herramientas de desarrollo

Para desarrollar aplicaciones web existen dos herramientas que constituyen opciones viables para desarrollar un proyecto: CMS y Frameworks.

Capítulo 1: Fundamentación teórica

1.11.1 CMS

Un CMS o Sistema Gestor de Contenidos (*Counter Management Systems*), es una herramienta muy flexible que está centrada especialmente en la gestión de contenidos mediante la web. Los gestores de contenido permiten crear contenidos HTML que facilitan la creación de los contenidos de forma visual. No es necesario tener conocimientos de ningún tipo de lenguaje de programación para administrarlos. Los gestores de contenidos permiten personalizar las fechas de publicación de contenidos y de caducidad. Disponen de una herramienta de roles y perfiles que permite definir las acciones que puede realizar un usuario. Los perfiles generales son Administrador, y usuario anónimo. Pero se pueden crear más perfiles. Separan la presentación del contenido. (Copyrigh, 2012)

La principal ventaja de estas estructuras, es que sin muchos conocimientos de programación ni maquetación, se puede administrar los contenidos del portal, poniendo como ejemplo, la fácil administración de un blog desarrollado en Wordpress.

En sus grandes comunidades, de desarrolladores y de soporte, se puede consultar y encontrar todo tipo de módulos y componentes.

Como desventaja resalta que al instalar el CMS vienen incorporados muchos componentes y módulos que ralentizaran la carga, teniendo que optimizar el portal para que funcione de manera eficiente y no resulte muy pesada su carga en navegadores.

1.11.2 Frameworks

Un framework es un producto que sirve como base para la programación avanzada de aplicaciones, que aporta una serie de funciones o códigos para realizar tareas habituales. Son librerías de código que contienen procesos o rutinas ya listos para usar. Son utilizados para no tener que desarrollar tareas más básicas, pues en el framework hay implementaciones que están probadas, funcionan y no se necesitan volver a programar. (Alvaréz, Manual de jquery, 2013)

La mayor ventaja que poseen es su adaptabilidad a todo tipo de proyecto, desde los más simples hasta los más complejos y, como se parte desde cero en cuanto a estructura, se puede ir adaptando a las necesidades que el proyecto requiera y afinar su efectividad, haciendo que éste quede optimizado.

Capítulo 1: Fundamentación teórica

También poseen una gran comunidad que facilita la implementación y desarrollo de funcionalidades que el proyecto requiera.

En cuanto a desventajas, dependiendo del framework que elijamos nos llevará una línea de tiempo en el aprendizaje mayor o menor, hasta que aprendamos a cómo implementar todas las APIs (del inglés *Application Programming Interface*, y en castellano Interfaz de programación de aplicaciones) y funciones.

Además, se tienen que poseer conocimientos de programación de antemano, ya que serán necesarios a la hora de implementar toda la infraestructura de usuarios y artículos.

1.11.3 Selección de la herramienta de desarrollo

Para apoyar la selección de la herramienta de desarrollo se muestra la siguiente tabla comparativa:

Tabla 3: Comparación entre CMS y Framework (*Forma-Pro, 2001-2013*)

CMS	Framework
Provee buenas funcionalidades básicas.	Provee buenas herramientas de desarrollo.
Buenos para un largo número de sitios web con funcionalidades típicas.	Bueno para un sitio con funcionalidades únicas.
No necesita reinventar funcionalidades comunes.	Funcionalidades comúnmente usadas pueden ser fácilmente añadidas.
Provee un conjunto de plugins listos para usar.	Provee un conjunto de librerías listas para usar.
Restringe lo que se puede hacer.	No tiene restricciones.
Personalización lenta para las funcionalidades únicas.	Personalización rápida.

Capítulo 1: Fundamentación teórica

Hacer cambios en el código puede resultar difícil.	Hacer cambios en el código es fácil.
--	--------------------------------------

Después de analizadas las características principales de los CMS y los *frameworks* con sus ventajas y desventajas, se decide escoger un *framework* pues se adapta mejor a las características del sitio web a desarrollar por ser un sitio con funcionalidades únicas que requiere un tiempo de consulta bastante rápido y los CMS para funcionalidades únicas la personalización es lenta además de que contiene varios módulos que no son necesarios para el desarrollo de la propuesta de solución y ralentizarían la carga de la aplicación además de que el equipo de desarrollo posee conocimientos sobre *frameworks* PHP lo que facilita el desarrollo.

Comparativa entre 5 frameworks PHP

Zfort Group realizó una investigación continua de las tendencias, votación de los desarrolladores, foros, comentarios y valoraciones de usuarios y se recopiló información única e interesante para la selección de un framework PHP para el desarrollo de una aplicación web. (Bizina, 2013)

Capítulo 1: Fundamentación teórica

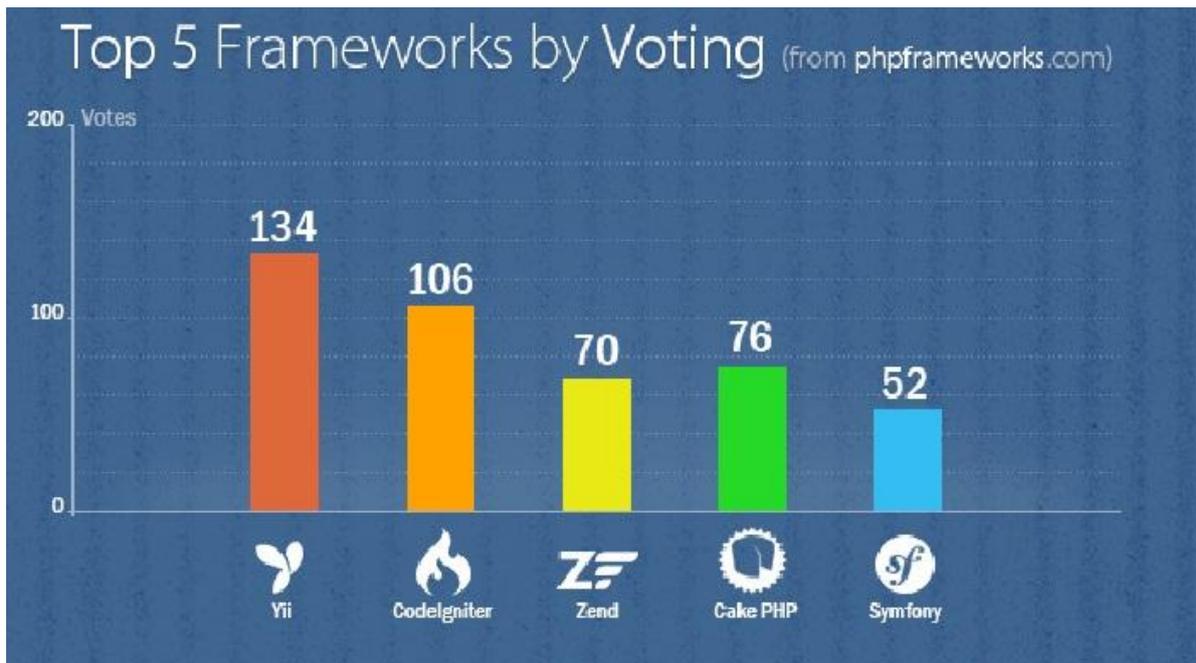


Ilustración 4: Comparación entre frameworks por votos de usuarios

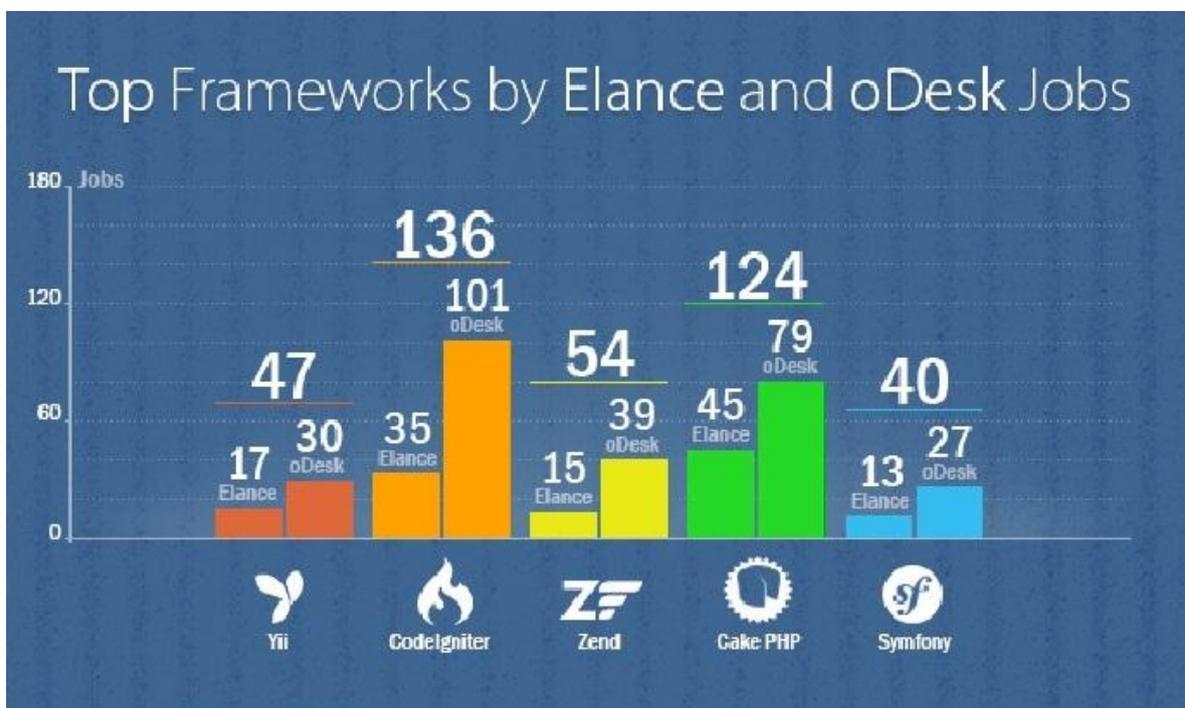


Ilustración 5: Comparación entre frameworks por sus características

Selección del framework PHP

Capítulo 1: Fundamentación teórica

Después de haber realizado un análisis de los frameworks PHP basados en las comparaciones anteriores se selecciona CodeIgniter en su versión 1.8, pues según los votos de los usuarios de las comunidades PHP ocupa el 2do lugar y según sus características ocupa el 1er lugar, además de ser un framework que se caracteriza por la velocidad de las consultas a la base de datos.

CodeIgniter es un framework para el desarrollo de aplicaciones que utiliza lenguaje PHP. Permite desarrollar proyectos de forma rápida y concentrarte en el proyecto minimizando la cantidad de código necesaria para una tarea determinada. Es distribuido bajo licencia de código abierto. (Alvaréz, 2010)

CodeIgniter contiene una serie de librerías que sirven para el desarrollo de aplicaciones web y además propone una manera de desarrollarlas. Marca una manera específica de codificar las páginas web y clasificar sus diferentes scripts, que sirve para que el código esté organizado y sea más fácil de crear y mantener. CodeIgniter implementa el proceso de desarrollo llamado *Model View Controller* (MVC), que es un estándar de programación de aplicaciones, utilizado para desarrollar sitios web.

Características generales de CodeIgniter: (Alvaréz, 2010)

- **Compatibilidad:** es compatible con la versión PHP 4 y PHP 5, lo que hace que se pueda utilizar en cualquier servidor, incluso en algunos antiguos.
- **Facilidad de instalación.**
- **Flexibilidad:** define una manera de trabajar específica, pero en muchos de los casos se puede seguirla o no. Algunos módulos como el uso de plantillas son totalmente opcionales.
- **Ligereza:** el núcleo de CodeIgniter es bastante ligero, lo que permite que el servidor no se sobrecargue interpretando o ejecutando grandes porciones de código. la mayoría de los módulos o clases que ofrece se pueden cargar de manera opcional, sólo cuando se van a utilizar realmente.
- **Documentación tutorializada:** La documentación de CodeIgniter es fácil de seguir y de asimilar, porque está escrita en modo de tutorial.

Frameworks CSS

Existen muchos frameworks CSS entre los que se encuentran: Cascade Framework, Twitter Bootstrap, Foundation, Skeleton, entre otros.

Capítulo 1: Fundamentación teórica

Según el fundador de Web Bizarro (<http://www.webbizarro.com/>) los frameworks más populares del 2014 son (en el orden en que aparecen): Bootstrap, Foundation, Semantic UI, PURE, Ulkit. (Rios, 2014)

Se decide comparar Bootstrap y Foundation por ser los frameworks más populares, para realizar la selección del framework CSS a utilizar en el desarrollo del sitio web.

Bootstrap y Foundation son frameworks similares y para determinar cuál es mejor Natalia Ventre desarrolló una página web con cada uno y comparó los resultados. Para prototipos rápidos obtuvo mejores resultados con Bootstrap. No tuvo inconvenientes y en aproximadamente media hora terminó el prototipo y la documentación de Bootstrap le resultó muy clara. (Ventre, 2013)

Basándonos en los resultados obtenidos por Natalia Ventre se escoge el framework Bootstrap para realizar el desarrollo del sitio web de marcadores sociales pues principalmente el equipo de desarrollo requiere de rapidez y agilidad en el desarrollo de la aplicación.

Bootstrap es un framework de CSS, es un conjunto de archivos CSS que incluye en las páginas para maquetarlas. (Otto, et al., 2015)

Bootstrap es un framework diseñado para simplificar el proceso de creación de diseños web. Para ello nos ofrece una serie de plantillas CSS y de ficheros JavaScript, los cuales nos permiten conseguir: (Otto, et al., 2015)

- Interfaces que funcionen bien en los navegadores actuales, y correcta en los no tan actuales.
- Un diseño que pueda ser visualizado de forma correcta en distintos dispositivos.
- Una mejor integración con las librerías que se usan habitualmente, como por ejemplo jQuery.
- Un diseño sólido basado en estándares como CSS3/HTML5.

La versión de este framework a utilizar para el desarrollo de la propuesta de solución es Bootstrap 3.0.

Capítulo 1: Fundamentación teórica

1.12 Librería JQuery

JQuery es una biblioteca JavaScript rápida, pequeña y brinda gran cantidad de funciones. Su objetivo principal es simplificar las tareas de creación de páginas web a la hora de interactuar con los documentos HTML, las cuales funcionan en todos los navegadores modernos. (Alvaréz, 2013)

Ventajas de jQuery: (Alvaréz, 2013)

- Implementa una serie de clases (de programación orientada a objetos) que nos permiten programar sin preocuparnos del navegador con el que nos está visitando el usuario, ya que funcionan de forma exacta en las plataformas más habituales.
- Ofrece ayuda en la creación de interfaces de usuario y efectos dinámicos.
- Gratuito, ya que la librería tiene licencia para uso en cualquier tipo de plataforma, personal o comercial.

Al igual que otras bibliotecas, JQuery ofrece una serie de funcionalidades basadas en JavaScript, que de otra manera requerirían de mucho más código. Con las funciones propias de esta biblioteca se pueden obtener buenos resultados en poco tiempo y espacio.

1.13 Lenguaje Unificado de Modelado

El lenguaje UML es un estándar OMG (*Object Management Group*) diseñado para visualizar, especificar, construir y documentar software orientado a objetos. (Jacobson, et al., 2015)

UML son las siglas de *Unified Modeling Language* (Lenguaje Unificado de Construcción de Modelos), notación (esquemática en su mayor parte) con que se construyen sistemas por medio de conceptos orientados a objetos. (Larman, 1998)

El Lenguaje Unificado de Modelado contiene un conjunto de notaciones y diagramas estándar para modelar sistemas orientados a objetos, y describe la semántica esencial de lo que estos diagramas y símbolos significan.

UML se puede usar para modelar distintos tipos de sistemas: sistemas de software, sistemas de hardware, y organizaciones del mundo real.

Capítulo 1: Fundamentación teórica

1.13.1 Herramienta CASE de modelado con UML

Las herramientas CASE (*Computer Aided Software Engineering*, Ingeniería de Software asistida por Computadora) son un conjunto de herramientas y métodos asociados que proporcionan asistencia automatizada en el proceso de desarrollo del software a lo largo de su ciclo de vida. (copyright SOFT112)

Fueron desarrolladas para automatizar esos procesos y facilitar las tareas de coordinación de los eventos que necesitan ser mejorados en el ciclo de desarrollo de software. Su objetivo es aumentar la productividad del desarrollo y mantenimiento de los sistemas informáticos, mejorar la calidad del software, reducir tiempos y costos de desarrollo y mantenimiento del software.

Además permiten automatizar el desarrollo del software, la documentación, generación del código, el chequeo de errores y la gestión del proyecto

1.13.2 Visual Paradigm para UML

La herramienta CASE para el desarrollo de aplicaciones utilizando modelado UML, es muy útil para los ingenieros de software, analistas y arquitectos que están interesados en la construcción de sistemas a gran escala y necesitan confiabilidad y estabilidad en el desarrollo orientado a objetos. (copyright SOFT112)

Visual Paradigm permite la navegación entre la escritura del código y su visualización, genera informes en formato PDF/HTML, proporciona un ambiente visualmente superior de modelado y permite la sincronización de código fuente en tiempo real.

Es una plataforma de modelado diseñada para apoyar a los arquitectos de sistemas, desarrolladores y diseñadores de UML en el proceso de análisis y diseño de aplicaciones complejas.

Para el modelado del sistema, se decide utilizar Visual Paradigm para UML en su versión 8.0, por ser una herramienta que soporta el ciclo de vida completo en el desarrollo de software: exploración, planificación, iteraciones, producción, mantenimiento y muerte del proyecto.

La versión de Visual Paradigm a utilizar para el desarrollo de la propuesta de solución es la 8.0.

Capítulo 1: Fundamentación teórica

1.14 Sistema gestor de bases de datos

PostgreSQL es un Sistema de Gestión de Bases de Datos Objeto-Relacional (*Object Relational Database Management System* por sus siglas en inglés ORDBMS) basado en el proyecto POSTGRES, de la Universidad de Berkeley. PostgreSQL es una derivación libre (*Open Source*) de este proyecto. Es un sistema objeto-relacional, ya que incluye características de la orientación a objetos, como son la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional.

Características de PostgreSQL: (Copyright, 1996-2014)

- Soporta distintos tipos de datos: además del soporte para los tipos básicos, también soporta datos de tipo fecha, monetarios, elementos gráficos y cadenas de bits. También permite la creación de tipos propios.
- Incorpora una estructura de datos *array*.
- Incorpora funciones de diversa índole: manejo de fechas y geométricas.
- Permite la declaración de funciones propias, así como la definición de disparadores.
- Incluye herencia entre tablas (aunque no entre objetos, ya que no existen), por lo que a este gestor de bases de datos se le incluye entre los gestores objeto-relacionales.

La versión a utilizar para el desarrollo de la propuesta de solución es PostgreSQL 9.2.

1.15 Servidor web

Apache es un servidor HTTP libre y ejecutable en varios sistemas operativos como son Unix, Linux, Solaris y Microsoft Windows. Permite múltiples lenguajes de script: PHP, Perl y Python.

La versión de Apache a utilizar para el desarrollo del sitio web es la 2.4 la cual presenta las siguientes mejoras con respecto a la versión 1.3: (Copyright, 2013)

- Soporte multiprotocolo: la nueva versión tiene la infraestructura necesaria para servir distintos protocolos.

Capítulo 1: Fundamentación teórica

- Soporte mejorado para las plataformas que no son tipo Unix: la versión 2.0 de Apache es más rápida y más estable que la versión antigua, en sistemas que no son tipo Unix, tales como Windows.
- Mensajes de error en diferentes idiomas: los mensajes de error que se envían a los navegadores están ahora disponibles en diferentes idiomas.

1.16 Balsamiq Mockups

Balsamiq Mockups es un programa producido por *Balsamiq Studios* que permite crear diagramas funcionales como dibujos a mano pero en forma digital. *Balsamiq* facilita edición, modificación y el movimiento de componentes para que los usuarios puedan realizar el mejor diseño posible antes de desarrollar el código real de la aplicación. (Alsheikh, 2011)

1.17 NetBeans

NetBeans es un entorno integrado de desarrollo o IDE (*Integrated Development Environment*) que permite realizar tareas asociadas a la programación como son: (Gimeno, et al., 2010-2011)

- Editar el código.
- Compilarlo.
- Ejecutarlo.
- Depurarlo.

NetBeans 7.3 permite el uso de tecnologías de desarrollo tanto para escritorio, como aplicaciones web, o para dispositivos móviles. Da soporte a las tecnologías, Java, PHP, C/C++ y HTML5. (Gimeno, et al., 2010-2011)

1.18 PHPUnit

PHPUnit es una herramienta de prueba para PHP que permite realizar las pruebas pertinentes al código, verificando que el funcionamiento de las aplicaciones PHP sea el deseado y encuentra bugs y errores que una vez solucionados mejorarán la calidad del desarrollo de la aplicación web. (Martínez Guaita, 2010)

Es un framework orientado a la programación para probar código PHP. Es una instancia de la arquitectura xUnit para la realización de pruebas unitarias. (Bergmann, 2001-2015)

Capítulo 1: Fundamentación teórica

Conclusiones parciales

Con el análisis de soluciones similares a la propuesta de solución, se pudo comprobar que actualmente existen sitios web de marcadores sociales que permiten organizar, guardar y compartir noticias, sin embargo, estos sitios solo contienen noticias internacionales y no sindicadas RSS de sitios nacionales, en especial noticias de interés para la comunidad universitaria. Como resultado de este análisis se decidió desarrollar un sitio web de marcadores sociales, que apoye de esta forma la socialización del conocimiento en la UCI. La implementación se realizará empleando el IDE NetBeans, los lenguajes, JavaScript, HTML5, CSS3 y PHP y los frameworks jQuery, Bootstrap, CodeIgniter y PHPUnit, asegurando de esta forma velocidad en el desarrollo por presentar las características propicias para la solución propuesta, el cual será dirigido y controlado por la metodología XP.

Capítulo 2: Propuesta de solución

Capítulo 2: Propuesta de solución

El ciclo de vida de XP consiste básicamente de seis fases, siendo las dos primeras exploración y planificación. En la fase de exploración los clientes crean las historias de usuario. El equipo de desarrollo se familiariza con las herramientas, la tecnología y las prácticas a ser utilizadas durante el proyecto. El objetivo de la fase de planificación es fijar la prioridad de cada una de las historias de usuario. Los programadores estiman cuanto esfuerzo requiere cada historia de usuario. (Abrahamsson, et al., 2002)

2.1 Descripción de la propuesta de solución

El sitio web de marcadores sociales va dirigido a todos los usuarios de la UCI, con el objetivo de que se tengan en cuenta las opiniones y preferencias de los usuarios para organizar los contenidos en un sitio web único que contenga las informaciones que se encuentran distribuidas en los sitios existentes en la Universidad. Además el sitio web permitirá guardar las noticias favoritas de un usuario específico para que este pueda consultarla desde cualquier ordenador con conexión a la Red en el momento en que lo desee, así como compartir con otros usuarios artículos y noticias.

Cuando el usuario accede a la aplicación se muestra una página principal con las noticias ordenadas por la cantidad de votos y comentarios. Las noticias que se muestran fueron extraídas previamente de varias fuentes insertadas de forma manual en la base de datos de la aplicación. Las fuentes que contiene el sitio son las siguientes: Blender++, Comunidad de Investigadores (Academia), Drupaleros, Firefoxmanía, HumanOS, Internos, Las Cubanas, Periódico Mella, Portal de Eventos Científicos Estudiantiles, Portal de seguridad, Señal 3 – Internacionales, Señal 3 – Nacionales, Señal 3 – UCI, Soluciones Java y Vigitec.

El usuario podrá acceder a la noticia completa, con sus comentarios, al dar un clic en el título de la misma, también podrá hacer búsquedas de noticias filtrando información, sugerir una fuente de información, conocer las estadísticas, tendencias y autenticarse en el sitio.

Una vez autenticado, el usuario además de poder acceder a todas las opciones anteriores también puede comentar y votar una noticia, insertar y eliminar un contacto, crear un marcador social y compartirlo o eliminarlo si lo desea.

Capítulo 2: Propuesta de solución

Dentro del sitio web los usuarios podrán desempeñar los siguientes roles: Anónimo, Autenticado y Administrador.

El usuario con rol Anónimo podrá acceder a las noticias, sugerir fuente, ver las estadísticas, tendencias y realizar búsqueda de noticias filtrando información.

El usuario con rol Autenticado hereda las opciones del usuario Anónimo, además de tener las opciones de seleccionar las fuentes de las que desea ver noticias, etiquetar noticias que no tengan categoría, gestionar sus marcadores sociales y contactos.

El usuario con rol Administrador será el encargado de gestionar las fuentes de información, y las fuentes sugeridas por los usuarios.

La actualización de las noticias de las fuentes se realizará de forma automática.

2.2 Usuarios del sistema

El sitio web que se desarrollará contará con una serie de funcionalidades para cumplir con los objetivos propuestos. Su implementación se realizará a través de módulos donde se establecerán un grupo de roles para asignar los diferentes permisos para su acceso. Tanto los roles como los permisos determinan el nivel de acceso a las funciones del sistema de cualquier usuario que interactúe con la aplicación.

Tabla 4: Roles de usuario.

Usuario	Descripción
Anónimo	Es la persona que navega por el sistema sin haberse registrado aún, interactúa con éste sin privilegios y tiene la posibilidad de visualizar las diferentes opciones que brinda el mismo.
Autenticado	Todos los usuarios deben autenticarse para acceder a cualquier opción que así lo requiera. Puede acceder a todas las opciones, menos a las de administración.

Capítulo 2: Propuesta de solución

Administrador	Son aquellos usuarios que tienen permiso para administrar distintas funcionalidades del Portal. Gestionan los contenidos y los servicios que se brindan.
----------------------	--

2.3 Planificación del proyecto por roles

De los roles propuestos por Kent Beck se utilizarán los siguientes.

- **Programador:** el programador escribe las pruebas unitarias y produce el código del sistema. Debe existir una comunicación y coordinación adecuada entre los programadores y otros miembros del equipo.
- **Cliente:** el cliente escribe las historias de usuario y las pruebas funcionales para validar su implementación. Además, asigna la prioridad a las historias de usuario y decide cuáles se implementan en cada iteración, centrándose en aportar mayor valor al negocio. El cliente es sólo uno dentro del proyecto pero puede corresponder a un interlocutor que está representando a varias personas que se verán afectadas por el sistema.
- **Encargado de pruebas (*Tester*):** el encargado de pruebas ayuda al cliente a escribir las pruebas funcionales. Ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para pruebas.
- **Encargado de seguimiento (*Tracker*):** el encargado de seguimiento proporciona realimentación al equipo en el proceso XP. Su responsabilidad es verificar el grado de acierto entre las estimaciones realizadas y el tiempo real dedicado, comunicando los resultados para mejorar futuras estimaciones. También realiza el seguimiento del progreso de cada iteración y evalúa si los objetivos son alcanzables con las restricciones de tiempo y recursos presentes. Determina cuándo es necesario realizar algún cambio para lograr los objetivos de cada iteración.

Tabla 5: Asignación de roles.

Rol	Nombre
Cliente	Ing Yunior Orosa Velázquez Ing Yordanis Rodríguez Rodríguez
Programador	Ailyn Perez Aguila

Capítulo 2: Propuesta de solución

- Usuario: representa a cualquier usuario que acceda al sitio web.
- Anónimo: usuario que no se ha autenticado en el sitio
- Autenticado: usuario que ha sido autenticado en el sitio.
- Administrador: usuario encargado de administrar el sitio web.
- Noticia: representa las noticias extraídas de las fuentes.
- Marcador social: es una noticia que ha sido guardada por un usuario autenticado.
- Categoría: representa la categoría que puede ser insertada a una noticia que hasta el momento no tenga categoría.
- Preferencias: permite al usuario marcar o desmarcar fuentes de noticias.
- Comentario: representa los comentarios de las noticias.
- Voto: representa los votos de las noticias.
- Fuente: representa las fuentes de RSS del sitio.
- Contacto: representa los contactos de un usuario autenticado.

2.5 Aspectos funcionales del sistema

Los aspectos funcionales del sistema son requisitos funcionales que debe cumplir la aplicación para satisfacer las necesidades del cliente. Es necesario redactar un listado de requerimientos bien detallados para que posteriormente puedan ser probados. A continuación se muestran los requisitos funcionales del producto.

R1 Extraer RSS

R1.1 Extraer RSS de las fuentes.

R2 Mostrar noticias similares

R2.1 Mostrar similares: Encargado de mostrar las noticias similares a una noticia específica.

R3 Mostrar noticias

R3.1 Mostrar noticias (ordenadas por cantidad de votos y comentarios).

R4 Administrar fuentes

Capítulo 2: Propuesta de solución

R4.1 Activar fuente.

R4.2 Desactivar fuente.

R4.3 Insertar fuente.

R4.4 Sugerir fuente.

R4.5 Mostrar fuente.

R5 Mostrar tendencias

R5.1 Mostrar noticias más votadas.

R5.2 Mostrar noticias más comentadas

R5.3 Mostrar usuarios que más comentan.

R5.4 Mostrar sitios que más noticias publican.

R6 Mostrar estadísticas

R6.1 Mostrar cantidad de noticias.

R6.2 Mostrar cantidad de fuentes activas.

R6.3 Mostrar cantidad de categorías de noticias.

R7 Configurar preferencias

R7.1 Marcar fuente (Seleccionar las fuentes de las que el usuario desea ver noticias)

R7.2 Desmarcar fuente (Seleccionar las fuentes de las que el usuario no desea ver noticias)

R8 Gestionar marcador social

R8.1 Insertar marcador social.

Capítulo 2: Propuesta de solución

R8.2 Eliminar marcador social.

R8.3 Compartir marcador social.

R8.4 Mostrar marcador social.

R9 Autenticar usuario por LDAP

R9.1 Autenticar usuario por LDAP.

R10 Votar noticias

R10.1 Votar positivamente.

R10.2 Votar negativamente.

R11 Comentar noticias

R11.1 Comentar noticias.

R12 Insertar categoría

R12.1 Insertar categoría.

R13 Buscar noticias

R13.1 Buscar noticias.

R14 Gestionar contactos

R14.1 Insertar contacto.

R14.2 Eliminar contacto.

R14.3 Mostrar contacto.

2.6 Aspectos no funcionales del sistema

Los aspectos no funcionales del sistema son propiedades o cualidades que el producto

Capítulo 2: Propuesta de solución

debe tener. Son requisitos que imponen restricciones en el diseño o la implementación.

Fiabilidad:

- El sistema tiene que estar disponible en todo momento.

Restricciones de diseño:

- Se utilizan herramientas y tecnología libres.

Eficiencia:

- El tiempo de respuesta a las peticiones se dará en el orden de milisegundos.

Requisitos para la documentación.

- El trabajo debe entregarse con el código con comentarios.

2.7 Exploración

En esta fase los clientes realizan las historias de usuario que desean que estén para la primera entrega. Cada historia de usuario describe una de las funcionalidades que el programa tendrá. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas, la tecnología y las prácticas a ser utilizadas durante el proyecto. La duración de esta fase puede extenderse desde unas pocas semanas a varios meses dependiendo de la adaptación del equipo de desarrollo. (Beck, et al., 1999)

2.7.1 Historias de usuarios

Las historias de usuario son la técnica utilizada en XP para especificar los requisitos del software. Se trata de tarjetas de papel en las cuales el cliente describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas. Las historias de usuario son descompuestas en tareas de programación y asignadas a los programadores para ser implementadas durante una iteración. (Canós, et al., 2015)

Capítulo 2: Propuesta de solución

Las historias de usuario sirven para registrar los requerimientos de los clientes y son utilizadas para poder realizar la estimación de cada una de las iteraciones durante la fase de planificación.

Son escritas por los clientes en base a lo que se estima es necesario para el sistema. Están escritas en un formato de oraciones en la terminología del cliente, sin necesidad de sintaxis técnicas.

También son utilizadas para poder crear las pruebas de aceptación. Por lo general se necesitan una o más pruebas de aceptación para verificar que la historia de usuario ha sido implementada correctamente.

Cada historia de usuario puede llevar entre 1 y 3 semanas en ser desarrollada en un desarrollo ideal. Este desarrollo ideal es cuánto tiempo se tarda en implementar una historia de usuario si no hay distracciones, ni otras asignaciones y se sabe exactamente qué hacer. Más de tres semanas significa que la historia de usuario debe ser dividida para ser implementada. Si toma menos de una semana se pueden combinar con otras historias de usuario. (Beck, et al., 1999)

Tabla 6: Historia de usuario.

Historia de usuario	
No: número consecutivo a partir del 1.	Nombre: identifica la HU.
Usuario: ¿Quién ejecuta la HU?	
Prioridad de negocio: define la relevancia e impacto de la historia de usuario para el negocio de acuerdo con las necesidades del usuario.	Puntos estimados: permite estimar la duración de la implementación, representando con 1, una semana de trabajo.
Nivel de complejidad: define la dificultad técnica que supone desarrollar la historia de usuario desde el punto de vista del programador.	Iteración asignada: precisa la iteración en la que será desarrollada la HU.

Capítulo 2: Propuesta de solución

Descripción: explica en qué consiste la HU, teniendo en cuenta las acciones realizadas por el usuario y la respuesta brindada por el sistema.

Observaciones: brinda información extra que se estime agregar para hacer más comprensible la HU. Por ejemplo: conceptos, post-condiciones y relación con otros requisitos.

2.7.2 Historias de usuario de la propuesta de solución

Las historias de usuario de la propuesta de solución se muestran en el [Anexo 2](#).

2.7.3 Prototipo de interfaz de usuario

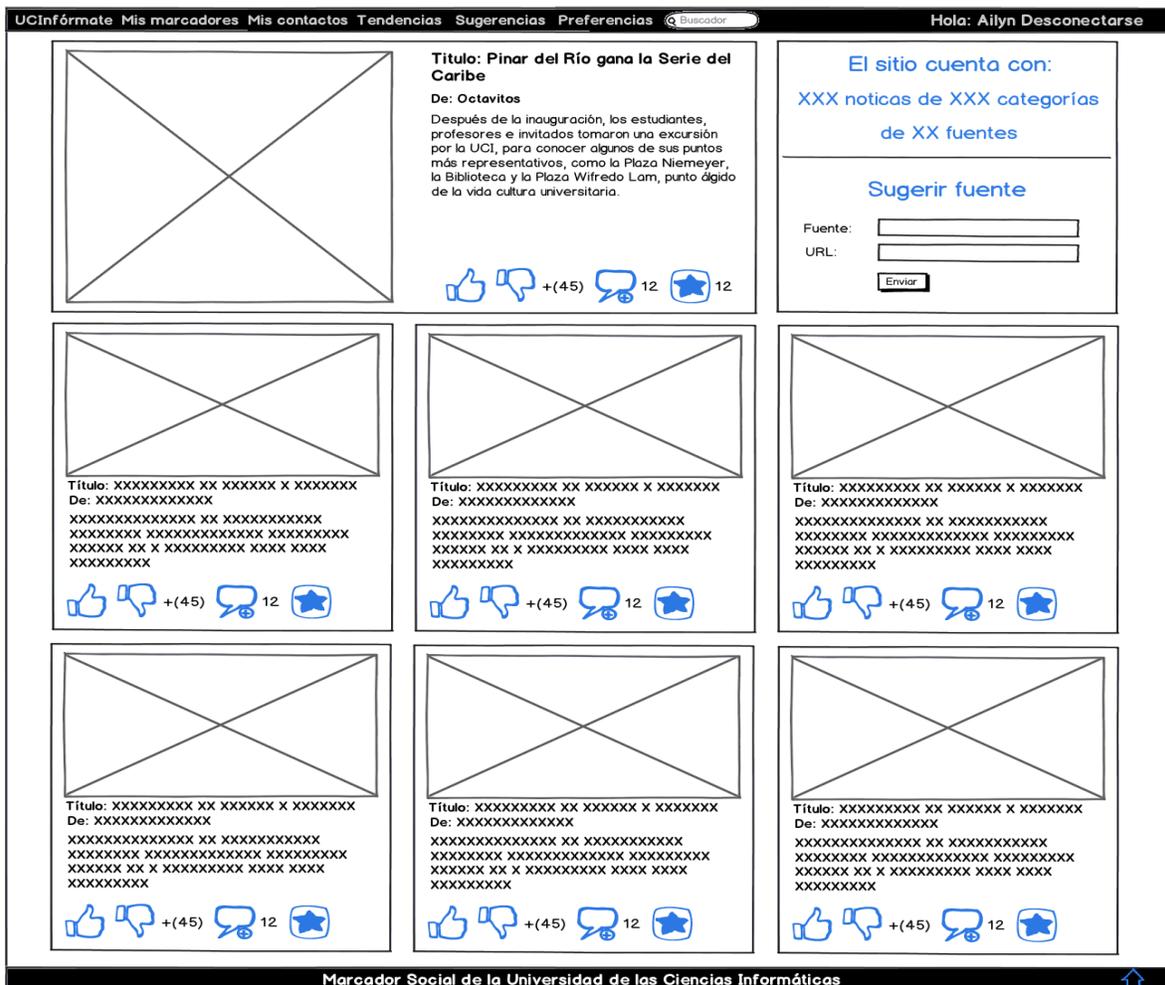


Ilustración 7: Prototipo de interfaz de usuario.

Capítulo 2: Propuesta de solución

2.8 Planificación

El objetivo de esta fase es fijar la prioridad de cada una de las historias de usuario y establecer cuál va a ser el contenido de la primera entrega (*release*). Los programadores estiman cuánto esfuerzo requiere cada historia de usuario y se establece el cronograma. La duración del calendario para la primera entrega no suele superar los dos meses.

Se debe llegar a un acuerdo entre los clientes y los programadores en cuáles serán las historias de usuario a ser implementadas durante cada iteración. Si se hace una buena preparación durante la fase de exploración esta actividad no suele llevar más de un día o dos. (Beck, et al., 1999)

2.8.1 Plan de Iteraciones

En el plan de iteraciones el cliente decide las historias seleccionadas para cada iteración.

Tabla 7: Planificación de las iteraciones.

Iteraciones	Orden de las HU a implementar	Cantidad de tiempo de trabajo
Número de iteración	Historias de usuarios en el orden en que van a ser implementadas.	Tiempo estimado en semanas en que se demorara la iteración

Tabla 8: Planificación de las iteraciones de la propuesta de solución.

Iteraciones	Orden de las HU a implementar	Cantidad de tiempo de trabajo
Iteración 1	HU1,HU2,HU3,HU4,HU5, HU6	6 semanas
Iteración 2	HU7,HU8, HU9,HU10	4 semanas
Iteración 3	HU11,HU12,HU13, HU14	4 semanas

2.8.2 Plan de duración de las iteraciones

Tabla 9: Plan de duración de las iteraciones.

Capítulo 2: Propuesta de solución

Iteración	Orden de las HU	Duración total
Número de iteración	Historias de usuario en el orden en que serán implementadas	Duración de la iteración en semanas

Tabla 10: Plan de duración de las iteraciones de la propuesta de solución.

Iteración	Orden de las HU	Duración total
1	<ul style="list-style-type: none">✓ Extraer RSS✓ Mostrar noticias similares✓ Mostrar noticias✓ Autenticar usuario✓ Votar noticia✓ Comentar noticia	6 semanas
2	<ul style="list-style-type: none">✓ Gestionar contactos✓ Gestionar marcador social✓ Administrar fuente✓ Insertar categoría	4 semanas
3	<ul style="list-style-type: none">✓ Mostrar tendencias✓ Buscar noticias✓ Mostrar estadísticas✓ Configurar preferencias	4 semanas

2.8.3 Plan de entregas

En el plan de entregas, el cliente elige aquellas historias de usuario que crea son más importantes para implementar. Cada entrega debe tener una duración máxima de dos meses y se estima la duración suponiendo que no hay retrasos o imprevistos. En el plan de iteraciones, el cliente selecciona las historias de usuario a ser implementadas, y se detallan las pruebas de aceptación para cada historia de usuario. (Beck, et al., 1999)

Capítulo 2: Propuesta de solución

2.8.4 Plan de entregas de la propuesta de solución

Tabla 11: Plan de entregas.

Entregable	1ra entrega	2da entrega	3ra entrega
Nombre de la aplicación a entregar	Nombre de la primera versión a entregar	Nombre de la segunda versión a entregar	Nombre de la tercera versión a entregar

Tabla 12: Plan de entregas de la propuesta de solución.

Entregable	1ra entrega (4ta semana de Marzo)	2da entrega (2da semana de Abril)	3ra entrega (2da semana de Mayo)
Sitio web de marcadores sociales para la Universidad de las Ciencias Informáticas	Versión 0.1	Versión 0.2	Versión 0.3

Tabla 13: Funcionalidades disponibles por entrega del producto de la propuesta de solución.

Historia de usuario	1ra entrega	2da entrega	3ra entrega
HU1. Extraer RSS	X		
HU2. Mostrar noticias similares	X		
HU3. Mostrar noticias	X		
HU4. Autenticar usuario por LDAP	X		
HU5. Votar noticia	X		
HU6. Comentar noticia	X		
HU7. Gestionar contactos		X	
HU8. Gestionar marcador social		X	
HU9. Administrar fuente		X	
HU10. Insertar categoría		X	

Capítulo 2: Propuesta de solución

HU11. Mostrar tendencias			X
HU12. Buscar noticias			X
HU13. Mostrar estadísticas			X
HU14. Configurar preferencias			X

2.9 Diseño

El diseño en XP se realiza en toda la vida del proyecto. Deben realizarse diseños simples, si existe una parte del sistema que requiere un desarrollo complejo, se divide, para lograr una mayor rapidez en la implementación. La recodificación es un proceso de mejora continua del diseño que lleva a cabo XP para perfeccionar y modificar la estructura y codificación del código sin cambiar su funcionalidad. Además en el diseño se generan como artefactos las Tarjetas CRC.

2.9.1 Patrón arquitectónico Modelo-Vista-Controlador

El framework de desarrollo seleccionado, CodeIgniter, tiene un estilo de programación que se debe seguir cuando se desarrolla aplicaciones web: el Modelo - Vista - Controlador. El Modelo - Vista - Controlador (en inglés *Model - View -Controller*) es un patrón de desarrollo o un estilo de arquitectura de software que separa el código fuente de las aplicaciones en tres grupos:

Modelo

Contiene todo el código que requiere acceso a base de datos. En el modelo se mantiene encapsulada la complejidad de la base de datos y simplemente se crean funciones para recibir, insertar, actualizar o borrar información de las tablas. Al mantenerse todas las llamadas a la base de datos en un mismo código, desde otras partes del programa se puede invocar las funciones que se necesiten del modelo y éste se encargará de procesarlas.

Vista

Capítulo 2: Propuesta de solución

La vista codifica y mantiene la presentación final al usuario de la aplicación. En la vista se coloca todo el código HTML, CSS, JavaScript que se tiene que generar para producir la página tal cual se desea que la vea el usuario.

Controlador

El controlador permite el enlace entre el modelo, la vista y cualquier otro recurso que se tenga que procesar en el servidor para generar la página web. En el controlador se guarda la lógica de las páginas y se realizan todas las acciones que sean necesarias para generarlas, ayudados del modelo o la vista.

Aplicación del patrón arquitectónico Modelo-Vista-Controlador

Controladores:

- *Readrss*: recoge todas las noticias existentes en las fuentes y las inserta en la base de datos.
- *Test*: muestra la vista principal del sitio con las noticias guardadas en la base de datos.
- *Categoría*: muestra las categorías de noticias guardadas en la base de datos.

Modelos:

- *Categoria_dao*: permite insertar y seleccionar las categorías, obtener el id de las noticias con una categoría específica y obtener las cantidad de categorías existentes en la base de datos.
- *Categoria_model*: permite crear una nueva categoría.
- *Dao*: clase genérica para cargar la base de datos y realizar consultas a la misma.
- *Fuente_dao*: permite obtener todas las fuentes de noticias y el total de fuentes existentes en la base de datos.
- *Fuente_model*: permite crear una nueva fuente.
- *Noticia_categoria_dao*: permite insertar y seleccionar la categoría de las noticias.
- *Noticia_categoria_model*: permite asociar una noticia con una categoría.

Capítulo 2: Propuesta de solución

- *Noticia_dao*: permite insertar y seleccionar las noticias.
- *Noticia_fuente_dao*: permite obtener las noticias por su fuente y obtener el url de las fuentes.
- *Noticia_fuente_model*: permite asociar una noticia con una fuente.
- *Noticia_model*: permite crear una nueva noticia.
- *Similar_dao*: permite insertar y seleccionar las noticias similares.
- *Similar_model*: permite asociar una noticia con otras noticias similares.

Vistas:

- *footer_view*: muestra el final de la vista.
- *head_view*: muestra el menú principal de la vista.
- *view_test*: muestra las noticias recogidas de los sitios de la UCI.

2.9.2 Patrones de diseño

Los patrones GRASP (*General Responsibility Assignment Software Patterns*) son patrones generales de software para la asignación de responsabilidades, que describen los principios fundamentales del diseño de objetos y la asignación de responsabilidades. (Larman)

Para lograr una mayor calidad en el diseño del sitio web a desarrollar se tuvieron en cuenta los siguientes patrones:

- Experto: asignar la responsabilidad a la clase que contiene toda la información necesaria para cumplirla. Se utilizó este patrón pues todas las clases se especializan en sus responsabilidades.

Capítulo 2: Propuesta de solución

```
<?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');
class Noticia_categoria_dao extends Model
{
    function Noticia_categoria_dao()
    {
        parent::Model();
        $this->load->database();
        $this->load->model('noticia_categoria_model');
    }
    //obtiene las noticias por categorias
    function obtenerNoticiaCategoria()
    {
        $consulta = $this->db->get('ci_noticia_categoria');
        $arreglo = array();
        foreach($q->result() as $row)
        $arreglo[] = new Noticia_categoria_model($row->noticiaid, $row->categoria);
        return $arreglo;
    }
    //inserta la noticia con su categoria
    function insertarNoticiaCategoria($id, $categoria_id)
    {
        $this->db->set('noticiaid', $id);
        $this->db->set('categoria', $categoria_id);
        $this->db->insert('ci_noticia_categoria');
    }
}
```

Ilustración 8: Aplicación del patrón Experto en la clase Noticia_categoria_dao

- Bajo acoplamiento: las clases se comunican con el menor número de clases posible para evitar mucha dependencia entre clase.
- Alta cohesión: las clases tienen sus funcionalidades bien definidas y todas las funcionalidades están relacionadas.
- Creador: asignar a una clase la responsabilidad de crear una instancia de otra clase. Se utilizó este patrón pues en las clases controladoras se cargan los modelos y las vistas además de que muchos modelos crean objetos de otros.

Capítulo 2: Propuesta de solución

```
<?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');
class Similar_dao extends Model
{
    function Similar_dao()
    {
        parent::Model();
        $this->load->database();
        $this->load->model('similar_model');
    }
    //obtiene un arreglo con las noticias similares a una noticia
    function getSimilares()
    {
        $consulta = $this->db->get('ci_similar');

        $arreglo = array();
        foreach($consulta->result() as $row)
            $arreglo[] = new Similar_model($row->noticiaid, $row->nsimilar);

        return $arreglo;
    }
}
```

Ilustración 9: Aplicación del patrón Creador en la clase Similar_dao

- Controlador: asignar la responsabilidad de controlar el flujo de eventos del sistema, a clases específicas. Este patrón se ve claramente debido a que se utiliza el framework CodeIgniter que está basado en el patrón MVC y las clases controladoras son las encargadas de controlar toda la información.

Capítulo 2: Propuesta de solución

```
<?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');
class Test extends Controller
{
    function Test()
    {
        parent::Controller();
    }

    function index()
    {
        $datos= $this->obtenerNoticias();

        // Cargamos las vistas de la aplicación.

        $tendencias = $this->obtener_Estadisticas();

        $this->load->view('plantilla/head_B_view', $tendencias);
        $this->load->view('view_test_no_auten', $data);
        $this->load->view('plantilla/footer_up');
    }
}
```

Ilustración 10: Aplicación del patrón Controlador en la clase Test

2.9.3 Tarjetas CRC

Las tarjetas CRC (clase, responsabilidad, colaboración) representan explícitamente varios objetos simultáneamente. Sin embargo, en lugar de simplemente rastrear los detalles de una colaboración en forma de envío de mensajes, las tarjetas CRC ponen el foco del diseñador en la motivación para la colaboración mediante la representación (potencialmente) de muchos mensajes como una frase. (Beck, 1989)

Tabla 14: Tarjeta CRC.

Nombre de la clase	
Responsabilidades de la clase	Colaboradores o clases colaboradoras

Capítulo 2: Propuesta de solución

Las tarjetas CRC de la propuesta de solución se muestran en el [Anexo 3](#)

Conclusiones Parciales

El desarrollo de las historias de usuario dejaron especificados los requisitos de la propuesta de solución. Las iteraciones se planificaron de tal manera que se vayan implementando las HU de mayor valor para el negocio según el cliente. Se aplicó el patrón arquitectónico Modelo-Vista-Controlador lo que permitió separar la parte lógica del acceso a datos y de la presentación. El desarrollo de las tarjetas CRC permitió comprender mejor el envío de datos entre las clases, sirviendo como base para la posterior implementación de las iteraciones.

Capítulo 3. Iteraciones

En este capítulo se toma como base el proceso de planificación descrito en el capítulo anterior dando paso al desarrollo de entregas funcionales al cliente. El centro de este capítulo está en la fase iteraciones de la metodología XP.

En la fase de iteración se realizan las pruebas unitarias automáticas al código antes de comenzar la implementación. Son implementadas las HU seleccionadas por el usuario para cada iteración. Además, se realizan las pruebas de aceptación, creadas por el cliente, al final de cada iteración.

Al finalizar la última iteración el proyecto está listo para poner en producción.(Abrahamsson, et al., 2002)

3.1 Iteraciones

La primera iteración debe ser un esqueleto de funcionamiento del sistema como un todo. Se escogen una serie de HU de alta prioridad para el negocio, que obliguen a crear la arquitectura completa. Luego se deben implementar las HU de la forma más simple que puedan funcionar. (Beck, et al., 1999)

Se debe realizar las pruebas unitarias antes de comenzar la iteración .Al final de cada iteración el cliente debe analizar que todas las HU estén implementadas y realizar las pruebas de aceptación y que estas resulten ser exitosas. (Abrahamsson, et al., 2002)

3.1.1 Primera iteración

El objetivo de esta iteración es implementar la HU1, HU2, HU3, HU4, HU5 y HU6. Estas son algunas de las HU con mayor prioridad del negocio para el cliente, y sirven de base para las próximas iteraciones. Para el desarrollo de la iteración se trazaron las tareas que se enuncian a continuación.

HU 1 Extraer RSS

Tarea No.1: Establecer el estándar de código y de comentarios.

Tarea No.2: Implementar las clases principales del núcleo de la aplicación.

Tarea No.3: Diseñar página principal.

Capítulo 3: Iteraciones

Tarea No.4: Extraer RSS.

HU 2 Mostrar noticias similares

Tarea No.5: Realizar prueba unitaria: Test_mostrar_similares.

Tarea No.6: Mostrar noticias similares.

HU 3 Mostrar noticias

Tarea No.7: Realizar prueba unitaria: Test_mostrar_noticias.

Tarea No.8: Mostrar noticias ordenadas por la cantidad de votos y comentarios.

HU 4 Autenticar usuario por LDAP

Tarea No.9: Realizar prueba unitaria: Test_autenticar_LDAP.

Tarea No.10: Autenticar usuario por LDAP.

HU 5 Votar noticias

Tarea No.11: Realizar prueba unitaria: Test_votarP.

Tarea No.12: Votar positivamente.

Tarea No.13: Realizar prueba unitaria: Test_votarN.

Tarea No.14: Votar negativamente.

HU 6 Comentar noticias

Tarea No.15: Realizar prueba unitaria: Test_comentar.

Tarea No.16: Comentar noticia.

Tabla 15. Representación de las tareas de ingeniería

Número de la tarea	Número de la historia de usuario
Nombre de tarea:	

Capítulo 3: Iteraciones

Tipo de tarea: de configuración o de implementación	Estimación: tiempo de demora en días en desarrollar la tarea.
Fecha de inicio:	Fecha de fin:
Programador responsable:	
Descripción:	

Las tablas que representan las tareas de ingeniería se encuentran descritas en el [Anexo 4](#).

3.1.2 Pruebas unitarias de la primera versión

Las pruebas unitarias son desarrolladas por los programadores y son las encargadas de verificar el código de forma automática, aseguran que un único componente de la aplicación produce una salida correcta para una determinada entrada. Este tipo de pruebas validan la forma en la que las funciones y métodos trabajan en cada caso particular. XP aconseja que la realización de las pruebas unitarias al sistema se realice de forma automatizada y en etapas tempranas del desarrollo, permitiendo disminuir la ocurrencia de defectos y aprovechar las ventajas de la retroalimentación que se produce en el proceso. (Beck, et al., 2000)

Entre las ventajas que ofrecen este tipo de pruebas se encuentran:

- Al escribir primero los casos de prueba, se define de manera formal los requisitos que se espera que cumpla la aplicación.
- Al escribir una prueba de unidad, se piensa en la forma correcta de utilizar un módulo que aún no existe.
- Los casos de prueba permiten que los desarrolladores no sientan inseguridad a la hora de realizar modificaciones en el código.

Para realizar las pruebas unitarias a la propuesta de solución se utilizó el framework PHPUnit en su versión 4.6, el cual permitió encontrar y solucionar errores en el código desarrollado y de esta forma lograr un código con más calidad. A continuación se representa un caso de prueba unitaria. Los casos de prueba unitarias de la primera iteración se muestran en el [Anexo 5](#).

Tabla 16: Caso de prueba unitaria.

Número del caso de prueba unitaria
Iteración: Número de la iteración

Capítulo 3: Iteraciones

Descripción: Se prueba la obtención de una noticia específica					
Responsable:					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones

3.1.3 Estrategia de codificación. Estándares y estilos a emplear

Los estándares de codificación son pautas de programación que no están enfocadas a la lógica del programa, sino a su estructura y apariencia física para facilitar la lectura, comprensión y mantenimiento del código.

La guía de usuario del framework CodeIgniter proporciona un estándar de codificación el cual fue utilizado en el desarrollo del código de la aplicación. (Copyright, 2006)

Nomenclatura de clases y métodos

Los nombres de clases siempre deberían comenzar con una letra mayúscula. Todos los otros métodos de clase deberían comenzar con minúsculas. Ejemplo de clase: `Noticia_categoria_dao`. Ejemplo de método: `insertarNoticiaCategoria ()`.

Capítulo 3: Iteraciones

```
<?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');
class Noticia_categoria_dao extends Model
{
    function Noticia_categoria_dao()
    {
        parent::Model();
        $this->load->database();
        $this->load->model('noticia_categoria_model');
    }
    //obtiene las noticias por categorias
    function obtenerNoticiaCategoria()
    {
        $consulta = $this->db->get('ci_noticia_categoria');
        $arreglo = array();
        foreach($q->result() as $row)
        $arreglo[] = new Noticia_categoria_model($row->noticiaid, $row->categoria);
        return $arreglo;
    }
    //inserta la noticia con su categoria
    function insertarNoticiaCategoria($id, $categoria_id)
    {
        $this->db->set('noticiaid', $id);
        $this->db->set('categoria', $categoria_id);
        $this->db->insert('ci_noticia_categoria');
    }
}
```

Ilustración 11: Ejemplo de la aplicación de la nomenclatura de clases y métodos.

Nombres de variables

La directriz para el nombramiento de variables es muy similar al usado para los métodos de clase. Concretamente, las variables deberían contener solamente letras minúsculas, usar guiones de subrayado como separadores y tener un nombre que razonablemente indique su propósito y contenido. Ejemplo de variable: \$value.

Capítulo 3: Iteraciones

```
<?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');
class Similar_model extends Model
{
    var $noticia;
    var $nsimilar;
    //constructor
    function Similar_model($pnoticia = NULL, $pnsimilar = NULL)
    {
        parent::Model();
        $this->noticia = $pnoticia;
        $this->nsimilar = $pnsimilar;
    }
    //obtiene el nombre de una variable
    function __get($name)
    {
        return $this->$name;
    }
    //modifica el nombre de una variable
    function __set($name, $value)
    {
        $this->$name = $value;
    }
}
```

Ilustración 12: Ejemplo de nombres de variable en la clase `Similar_model`

Comentarios

En general, el código debe ser comentado, no sólo ayuda a describir el flujo y la intención del código para los programadores con menos experiencia, sino que puede resultar muy valioso, al regresar a su propio código meses después.

```
<?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');
class Similar_dao extends Model
{
    function Similar_dao()
    {
        parent::Model();
        $this->load->database();
        $this->load->model('similar_model');
    }
    //obtiene un arreglo con las noticias similares a una noticia
    function getSimilares()
    {
        $consulta = $this->db->get('ci_similar');

        $arreglo = array();
        foreach($consulta->result() as $row)
            $arreglo[] = new Similar_model($row->noticiaid, $row->nsimilar);

        return $arreglo;
    }
}
```

Ilustración 13: Ejemplo de comentarios en la clase *Similar_dao*

3.1.4 Librerías en CodeIgniter

Las librerías en CodeIgniter son clases de programación orientadas a objetos (POO) preparadas para realizar tareas típicas en el desarrollo de páginas web. Son las clases que este framework pone a disposición de los desarrolladores para construir aplicaciones web. Estas librerías contemplan muchos de los casos habituales que se pueden necesitar resolver en páginas web, como accesos a base de datos, trabajo con calendarios, con URLs, *cookies*, entre otros. (Alvaréz, 2010)

Librerías utilizadas en el desarrollo del sitio web

- *Simplepie*: librería PHP basada en RSS y Atom feed que permite leer las noticias sindicadas por otros sitios web y mostrarlas en un sitio web. Recoge informaciones de las noticias como son: título, url, descripción, contenido, autor, categoría y fecha.
- *Benchmark*: permite marcar puntos y calcular la diferencia de tiempo que demora en ejecutar un código específico. Permite mostrar además el consumo de memoria.
- *Calendar*: permite la creación de calendarios.
- *Controller*: clase controladora de CodeIgniter.
- *Form_validation*: permite validar formularios.

- *Model*: clase modelo de CodeIgniter.
- *URI*: determina las URLs y el enrutamiento.
- *Service_lib*: permite autenticar usuario por LDAP.

3.1.5 Helpers en CodeIgniter

Los *helpers* son bibliotecas de funciones que CodeIgniter proporciona. Son juegos de funciones, ordenados por temática en diferentes paquetes, que ayudan en la realización de tareas habituales en las aplicaciones web. Existen *helpers* para muchas cosas distintas, como para trabajo con *arrays*, fechas, *cookies*, *emails*, URLs, formularios, entre otros. Los *helpers* no dependen de programación orientada a objetos, sino que son funciones (de programación funcional o procedimental, más tradicional que la POO) que se pueden utilizar para solucionar temas concretos y cuyo uso es independiente. (Alvaréz, 2010)

Helpers utilizados en el desarrollo del sitio web

- *array_helper*: biblioteca para trabajar con arreglos.
- *date_helper*: biblioteca para trabajar con fechas.
- *form_helper*: biblioteca para trabajar con formularios.
- *text_helper*: biblioteca para trabajar con texto.
- *url_helper*: biblioteca para trabajar con URLs.

La implementación en XP comienza en la fase de iteración y continúa en las fases de producción y mantenimiento hasta llegar a la fase de muerte del proyecto. En la fase de producción es donde el producto se pone en producción. En la fase de mantenimiento se debe mantener el sistema en ejecución y se hacen las recodificaciones (*refactoring*) del código que no se pudieron realizar anteriormente. Finalmente se llega a la fase de muerte del proyecto cuando el cliente esté satisfecho con el sistema y no tenga ninguna otra funcionalidad que agregar.

3.1.6 Pruebas de aceptación de la primera versión

Las pruebas de aceptación están destinadas a evaluar si al final de una iteración se obtuvo la funcionalidad requerida, además de comprobar que dicha funcionalidad sea la esperada por el cliente. (Beck, 2000)

Capítulo 3: Iteraciones

Las pruebas de aceptación se describen mediante una tabla llamada “Caso de prueba de aceptación” y son destinadas a evaluar si al final de una iteración se obtuvo la funcionalidad requerida, creada sobre la base de una HU. En cada ciclo de la iteración del desarrollo, el cliente debe especificar uno o diversos escenarios para comprobar que las HU han sido correctamente implementadas. (Joskowicz, 2008)

Tabla 17: Caso de prueba de aceptación.

Caso de prueba de aceptación	
Código: Código con que se identifica el caso de prueba de aceptación.	Historia de usuario: Nombre de la HU.
Nombre: Nombre de la funcionalidad que se prueba.	
Descripción: descripción de la funcionalidad.	
Condición de ejecución: condición que tiene que cumplirse para que se pueda acceder a la funcionalidad.	
Entrada/Pasos de ejecución: Pasos a seguir para acceder a la funcionalidad.	
Resultado esperado: Resultado que se espera al acceder a la funcionalidad.	

Casos de pruebas de aceptación para la versión 0.1

A continuación se muestran los casos de prueba de aceptación de la primera versión. Para esta entrega se realizaron nueve casos de prueba de aceptación, identificando tres no conformidades. Debido a la complejidad mínima de las no conformidades, fueron todas resueltas y no quedaron casos de pruebas pendientes para la siguiente versión.

A continuación se muestra el caso de prueba de aceptación de la HU Extraer RSS, los demás casos de prueba de esta versión se encuentran en el [Anexo 6](#).

Tabla 18: Caso de prueba de aceptación 1 HU Extraer RSS

Caso de prueba de aceptación	
Código: PA1HU1	Historia de usuario: Extraer RSS
Nombre: Extraer RSS.	
Descripción: Se extraen los RSS de las fuentes existentes en la base de datos.	
Condición de ejecución:	

Capítulo 3: Iteraciones

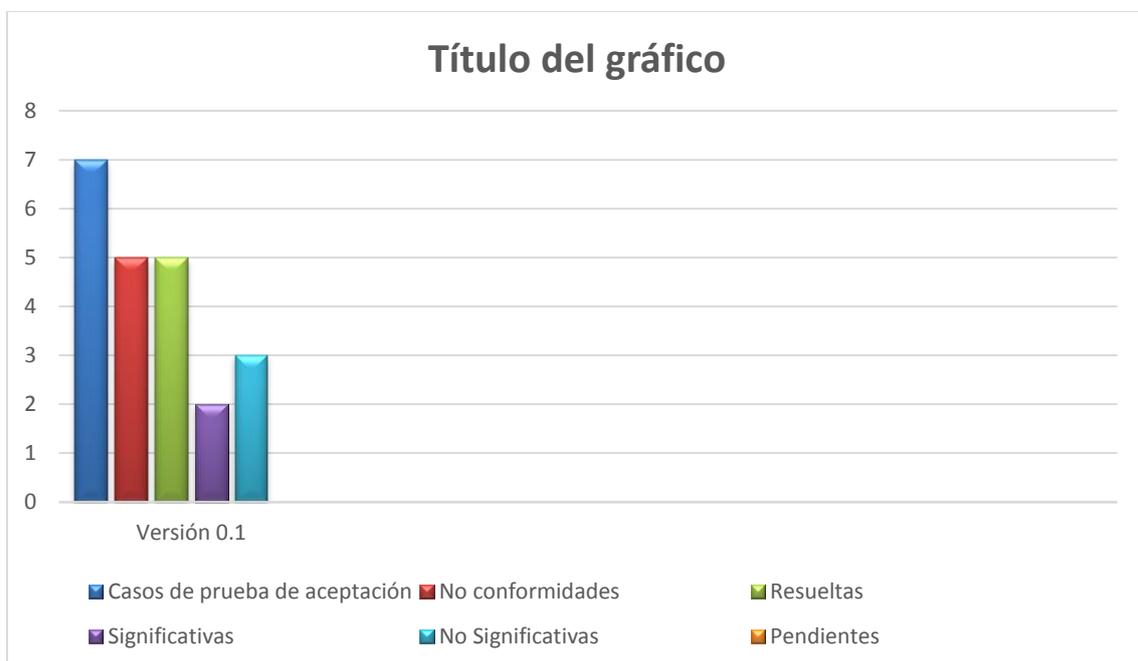
Entrada/Pasos de ejecución:

El sitio web extrae los RSS de las fuentes automáticamente cada 1 hora.

Resultado esperado: Se actualizan las noticias en la base de datos.

3.1.7 Resultados de las pruebas de aceptación de la primera versión

Para la entrega de la primera versión de la aplicación se realizaron un total de siete casos de prueba de aceptación, se detectaron 5 no conformidades, las cuales fueron resueltas.



3.1.8 Segunda iteración

El objetivo de esta iteración es implementar las HU6, HU7, HU8, HU9. Las tablas que representan estas tareas se encuentran descritas en el [Anexo 7](#).

HU 6 Gestionar contactos

Tarea No.17: Realizar prueba unitaria: Test_ insertar_contacto.

Tarea No.18: Insertar contacto.

Tarea No.19: Realizar prueba unitaria: Test_ eliminar_contacto.

Capítulo 3: Iteraciones

Tarea No.20: Eliminar contacto.

Tarea No.21: Realizar prueba unitaria: Test_ mostrar_contacto.

Tarea No.22: Mostrar contacto.

HU 7 Gestionar marcador social

Tarea No.23: Realizar prueba unitaria: Test_ insertar_marcador.

Tarea No.24: Insertar marcador social.

Tarea No.25: Realizar prueba unitaria: Test_ compartir_marcador.

Tarea No.26: Compartir marcador social.

Tarea No.27: Realizar prueba unitaria: Test_ eliminar_marcador.

Tarea No.28: Eliminar marcador social.

Tarea No.29: Realizar prueba unitaria: Test_ mostrar_marcador.

Tarea No.30: Mostrar marcador social.

HU 9 Administrar fuente

Tarea No.31: Realizar prueba unitaria: Test_ activar_fuente.

Tarea No.32: Activar fuentes.

Tarea No.33: Realizar prueba unitaria: Test_ desactivar_fuente.

Tarea No.34: Desactivar fuentes.

Tarea No.35: Realizar prueba unitaria: Test_ sugerir_fuente.

Tarea No.36: Sugerir fuente.

Tarea No.37: Realizar prueba unitaria: Test_ insertar_fuente.

Tarea No.38: Insertar fuente.

Capítulo 3: Iteraciones

Tarea No.39: Realizar prueba unitaria: Test_ mostrar_fuente.

Tarea No.40: Mostrar fuente.

HU 10 Insertar categoría

Tarea No.41: Realizar prueba unitaria: Test_ insertar_categoria.

Tarea No.42: Insertar categoría.

Capítulo 3: Iteraciones

3.1.9 Pruebas unitarias de la segunda iteración

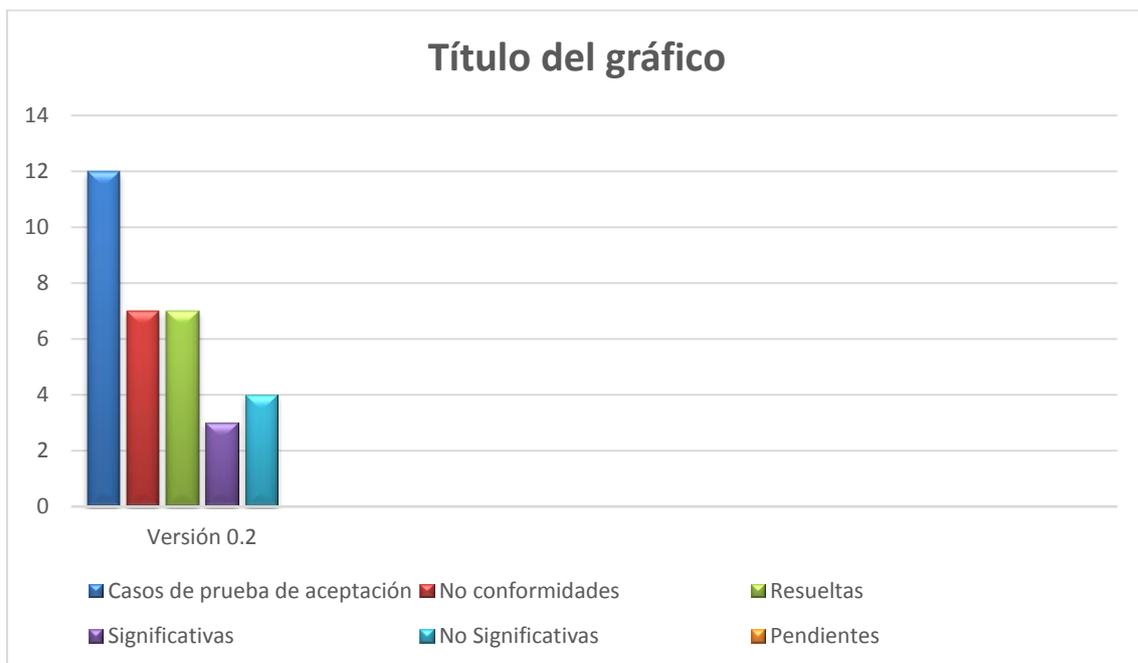
Los casos de prueba unitarias de la segunda iteración se muestran en el [Anexo 8](#).

3.1.10 Pruebas de aceptación de la segunda iteración

Las tablas que representan los casos de prueba de aceptación para esta versión se encuentran descritas en el [Anexo 9](#).

3.1.11 Resultados de las pruebas de aceptación de la segunda versión

Para la entrega de la segunda versión de la aplicación se realizaron un total de doce casos de prueba de aceptación, se detectaron siete no conformidades, tres significativas y cuatro no significativas, todas fueron resueltas.



3.1.12 Tercera iteración

La tercera y última iteración tiene como objetivo implementar las HU10, HU11, HU12 y HU13. Al final de esta iteración se obtiene una versión del producto donde se satisfacen todos los requisitos del cliente. Las tablas que representan las tareas de ingeniería de esta iteración se encuentran descritas en el [Anexo 10](#).

HU 10 Mostrar tendencias

Capítulo 3: Iteraciones

Tarea No.43: Realizar prueba unitaria: Test_ mostrar_votadas.

Tarea No.44: Mostrar noticias más votadas.

Tarea No.45: Realizar prueba unitaria: Test_ mostrar_comentadas.

Tarea No.46: Mostrar noticias más comentadas.

Tarea No.47: Realizar prueba unitaria: Test_ mostrar_usuarios.

Tarea No.48: Mostrar usuarios que más comentan.

Tarea No.49: Realizar prueba unitaria: Test_ mostrar_sitios.

Tarea No.50: Mostrar sitios que más noticias publican.

HU 11: Buscar noticias

Tarea No.51: Realizar prueba unitaria: Test_ buscar.

Tarea No.52: Buscar noticias.

HU 12: Mostrar estadísticas

Tarea No.53: Realizar prueba unitaria: Test_ mostrar_cantN.

Tarea No.54: Mostrar cantidad de noticias.

Tarea No.55: Realizar prueba unitaria: Test_ mostrar_cantF.

Tarea No.56: Mostrar cantidad de fuentes activas.

Tarea No.57: Realizar prueba unitaria: Test_ mostrar_cantC.

Tarea No.58: Mostrar cantidad de categorías de noticias.

HU 13: Configurar preferencias

Tarea No.59: Realizar prueba unitaria: Test_ marcar_fuente.

Tarea No.60: Marcar fuentes.

Capítulo 3: Iteraciones

Tarea No.61: Realizar prueba unitaria: Test_ desmarcar_fuente.

Tarea No.62: Desmarcar fuentes.

3.1.13 Pruebas unitarias de la tercera iteración

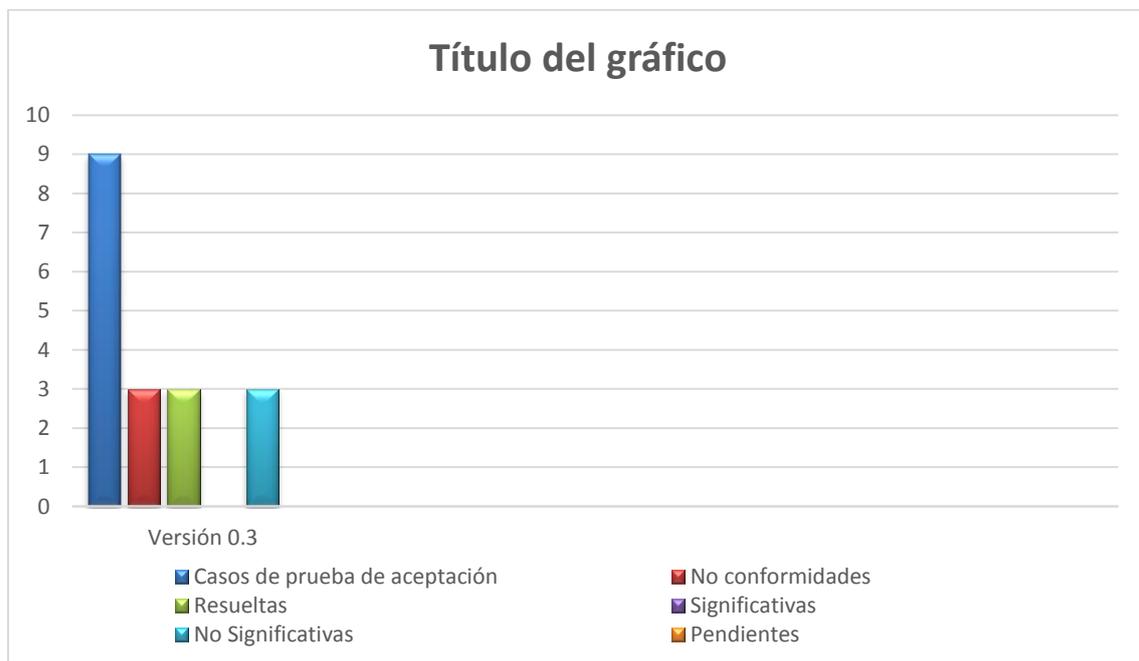
Los casos de prueba unitaria de la tercera iteración en el [Anexo 11](#).

3.1.14 Pruebas de aceptación de la tercera iteración

Las tablas que representan los casos de prueba de aceptación para esta versión se encuentran descritas en el [Anexo 12](#).

3.1.15 Resultados de las pruebas de aceptación de la tercera versión

Para la entrega de la tercera versión de la aplicación se realizaron un total de nueve casos de prueba de aceptación, se detectaron 3 no conformidades no significativas, las cuales fueron resueltas.



Conclusiones parciales

La realización de las pruebas unitarias permitió disminuir la ocurrencia de errores demostró la eficacia de un proceso desarrollo dirigido por pruebas. Se satisficieron las necesidades del cliente. Las pruebas de aceptación realizadas comprobaron la correcta implementación de las funcionalidades cumpliendo con las expectativas del cliente.

Conclusiones generales

Finalizada la implementación del sitio web de marcadores sociales para la Universidad de las Ciencias Informáticas, se arribando a las siguientes conclusiones:

- El estudio realizado permitió profundizar sobre el desarrollo de sitios web de marcadores sociales, distinguiendo los principales servicios que brindan, adecuándolo a las características del sitio web, además se determinan las herramientas que se utilizan en el desarrollo, teniendo en cuenta las tendencias de las tecnologías web y el uso de herramientas libres.
- El estudio realizado permitió escoger el framework CodeIgniter como el más adecuado para el desarrollo, seleccionando como metodología para la guía del proceso a XP, generando los artefactos fundamentales para lograr un desarrollo con calidad y poder cumplir con las expectativas del cliente.
- Las herramientas y tecnologías seleccionadas permitieron desarrollar un sitio web para satisfacer las necesidades definidas por el cliente, identificando las funcionalidades y características para eliminar los problemas existentes y convertirlo en una herramienta de comunicación entre los usuarios de la UCI.
- Las pruebas de aceptación realizadas permitieron verificar el correcto funcionamiento de las funcionalidades definidas demostrando la eficacia del proceso de desarrollo.

Después de la entrega de la propuesta de solución al cliente, se pudo constatar que cumple con todas las funcionalidades identificadas, satisfaciendo las necesidades existentes para organización y distribución de los contenidos informativos.

Recomendaciones

Después de analizados los resultados obtenidos por la presente investigación se recomienda:

- Usar el sitio web de marcadores sociales en todas las áreas de la Universidad.
- Extender la sindicación de contenidos a todos los sitios web de la Universidad.
- Estandarizar la forma en que se sindicán los contenidos de los sitios web de la Universidad.

Referencias

- Abrahamsson, Pekka, y otros. 2002.** *Agile software development methods. Review and analysis.* 2002.
- Alsheikh, Tamara. 2011.** *Balsamiq manual V5.* 2011.
- Alvaréz, Miguel Angel. 2010.** Crear tus propias librerías en CodeIgniter. [En línea] 17 de Junio de 2010. [Citado el: 23 de Abril de 2015.] <http://www.desarrolloweb.com/articulos/crear-librerias-codeigniter.html>.
- . **2010.** *Manual de CodeIgniter.* 2010.
- . **2013.** *Manual de jquery.* 2013.
- Becerril-Isidro, Javier y Vallejo-Lass, Adriana Paola. 2012.** *La web 2.0: un análisis de su impacto en lo social, político, cultural y económico.* México : s.n., 2012.
- Beck, Kent. 1989.** *A Laboratory For Teaching Object-Oriented Thinking.* New Orleans, Louisiana : s.n., 1989.
- Beck, Kent y Andres, Cynthia. 1999.** *Extreme Programming Explained: Embrace Change.* 1999.
- Beck, Kent y Fowler, Martin. 2000.** *PlanningExtremeProgramming.* s.l. : Addison wesley, 2000. 0-201-71091-9.
- Belmonte Fernández, Oscar. 2005.** *Introducción al lenguaje de programación Java.: Una guía básica.* 2005.
- Bergmann, Sebastian. 2001-2015.** Welcome to PHPUnit! [En línea] 2001-2015. [Citado el: 28 de Abril de 2015.] <https://phpunit.de/>.
- Bizina, Elena. 2013.** Top 5 PHP Frameworks Infographic | Zfort Group Blog. [En línea] 5 de Abril de 2013. [Citado el: 17 de Abril de 2015.] <http://www.zfort.com/blog/top-5-php-frameworks-infographic..>
- Calabria, Luis y Píriz , Pablo . 2003.** *Metodología XP.* 2003.
- Canós, José H., Letelier , Patricio y Penad, M^a Carmen. 2015.** *Metodologías ágiles en el desarrollo de software.* Valencia : s.n., 2015.
- Castillo Cantón, Alejandro. 2015.** *Manual de HTML5 en español.* 2015.
- Cobo Romaní, Cristóbal y Pardo Kuklinski, Hugo. 2007.** *Planeta Web 2.0 o medios fast food.* México : Grup de Recerca d'Interaccions Digitals, Universitat de Vic, 2007. 978-84-934995-8-7 .
- Condori Palomeque, Raquel y Ticona Condori, Shirley Fabiola. 2015.** *Comparación de metodologías.* 2015.

- Copyright. 1996-2014 .** *PostgreSQL 9.3.5 Documentation.* 1996-2014 .
- copyright SOFT112.** Visual Paradigm for UML Standard 8.0- Free Download. [En línea] [Citado el: 2 de Mayo de 2015.] <http://visual-paradigm-for-uml-standard.soft112.com/>.
- Copyrigh. 2006-2012.** *Guía de usuario de CodeIgniter.* s.l. : EllisLab, Inc., 2006-2012.
- 2001-2013.** Job for . [En línea] Forma-Pro, 2001-2013. [Citado el: 6 de 6 de 2015.] <http://forma-pro.com/cms-vs-framework>.
- . **2013.** *Visión general de las nuevas funcionalidades de Apache 2.0 - Servidor HTTP Apache.* 2013.
- Cortés, Carlos. 2015.** *Metodologías ágiles en el desarrollo de software.* 2015.
- DECSAI. 2006.** *Programación J2ME con.* 2006.
- Dixon, Angela y College Swansea, Gower.** *Marcadores sociales.*
- D'Souza, Quentin. 2015.** *Web 2.0 Ideas for Educators: A Guide to RSS and More.* 2015.
- Fraganillo, Jorge y Catalán, Marcos Antonio. 2005.** *Bitácoras y sindicación de contenidos: dos herramientas para.* Barcelona : s.n., 2005. 1575-5886.
- Gauchat, Juan Diego. 2012.** *El gran libro de HTML5, CSS3 y javascript.* Barcelona : Marcombo, 2012. SE-7867-2011.
- Gimeno, Juan Manuel y González, José Luis. 2010-2011.** *Introducción a Netbeans.* 2010-2011.
- Guardo García, María Elena. 2009.** *LOS COMPONENTES DEL DISEÑO TEÓRICO DE LA INVESTIGACIÓN CIENTÍFICA. UNA REFLEXIÓN PRAXIOLÓGICA.* 2009.
- Hernández Orallo, Enrique .** *El Lenguaje Unificado de Modelado (UML).*
- Holzner, Steven. 2006.** *Secrets of RSS.* 2006.
- Instituto Tecnológico de Celaya. 2010.** *Programación.* 2010.
- Jacobson, Ivar , y otros. 2015.** *El Proceso Unificado de Desarrollo de Softwar.* 2015.
- Joskowicz, José. 2008.** *Reglas y Prácticas en eXtreme Programming.* 2008.
- Kabir, Mohammed J.** *La biblia del Servidor Apache.*
- Lang, Pablo. 2013.** Sindicación de contenidos (RSS y Atom). [En línea] 2013. [Citado el: 13 de Abril de 2015.] <http://pablolandweb.blogspot.com/2013/03/sindicacion-de-contenidos-rss-y-atom.html>.
- Larman, Craig. 1998.** *Applying-uml-and-patterns.* 1998.

Letelier, Patricio y Penadés, M^a Carmen. *Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP)*. Valencia : s.n. 46022 .

López, Patricia . *INGENIERÍA DEL SOFTWARE I. Práctica 1.Herramienta CASE Visual Paradigm*.

Maraboli Rosselott, Marcelo. 2003. *Manual de programación en PHP*. s.l. : Valparaíso, 2003.

Martínez Guaita, Alvaro. 2010. Desarrolloweb.con. *PHPUnit: herramienta de testeo PHP*. [En línea] 16 de Noviembre de 2010. [Citado el: 18 de Abril de 2015.]

http://www.desarrolloweb.com/de_interes/phpunit-herramienta-testeo-php-4436.html.

Muños Sánchez, Pablo. 2007. *La Web 2.0 en el aula de traducción*. España : Panace@, 2007.

Ortiz de Zarate Tercero, Alberto. 2008. *Manual de uso del blog en la empresa. Cómo prosperar en la sociedad de la conversación*. Barcelona : edicions cantalaia, 2008. 978-84-612-1535-5.

Otto, Mark y Thornton, Jacob. 2015. *Bootstrap 3, el manual oficial*. 2015.

Pavón Mestras, Juan. 2012-2013. *Servidores Web – Apache*. 2012-2013.

Peña Ruiz, Inmaculada. 2011. *Manual de Edición del Recurso Lector RSS, Alkacon*. 2011.

Sæther Bakken, Stig , y otros. 1997, 1998, 1999, 2000, 2001. *Manual de PHP*. 1997, 1998, 1999, 2000, 2001.

Sánchez, Jorge. 2003. *JavaScript. Manual de referencia*. 2003.

Sardá, Javier. 2015. Lector RSS. [En línea] Invespromo, 2015. [Citado el: 2 de mayo de 2015.] <http://www.rssnom.es/lector-rss>.

Vázquez Mariño, Carlos. 2008. *Programación en PHP5. Nivel Básico*. 2008.

Ventre, Natalia. 2013. *Bootstrap vs Foundation Parte 1: Prototipo rápido*. 2013.

2013. zfort. [En línea] 5 de abril de 2013. [Citado el: 115 de Abril de 2015.]

<http://www.zfort.com/blog/top-5-php-frameworks-infographic/>.

Anexos:

Anexo 1. Resumen de entrevistas con estudiantes de la UCI

Preguntas realizadas en la entrevista con estudiantes de varias facultades de la UCI:

¿Le gusta mantenerse actualizado del acontecer en la UCI, Cuba y el resto del mundo?

¿Qué sitios web de la UCI visita más y por qué?

¿Crees que se puede conocer cuáles son las noticias más relevantes de los sitios de la UCI?

¿Qué tiempo promedio se demora en leer las noticias de interés en los sitios existentes en la UCI?

¿Usas los marcadores del navegador para guardar enlaces a páginas de interés? ¿Qué crees de esta forma de guardar noticias?

¿Algún sitio de la UCI permite compartir una noticia con otros usuarios? ¿Crees que sería útil compartir noticias con otros usuarios?

Resumen de las respuestas

A la mayoría de los estudiantes entrevistados les interesa mantenerse actualizado. El 70% de los entrevistados visitan los portales de cada facultad, los sitios de las diferentes comunidades además de la Intranet e Internos pues en estos sitios se publican noticias variadas que son de interés y utilidad para los estudiantes de la UCI. Actualmente para saber si una noticia es relevante se deben revisar casi todos los sitios de la Universidad. El tiempo promedio en que se demora en leer las noticias de interés para un usuario, según la mayoría de los entrevistados es de 30 minutos o más. El 60% de los estudiantes entrevistados guardan marcadores en su navegador para acceder a las páginas que contienen noticias de su interés, un 50% opina que los marcadores son una herramienta útil pero que estas solo se pueden acceder desde tu ordenador. Todos los entrevistados concuerdan en que actualmente ningún sitio de la UCI permite compartir noticias, y el 65% de los entrevistados plantean que sería de gran utilidad para la comunidad universitaria, pues ahorraría tiempo de búsqueda de las noticias.

Anexo 2. Historias de usuario

Tabla 19: HU1 Extraer RSS.

Historia de usuario	
No: 1	Nombre: Extraer RSS
Usuario: Administrador	
Prioridad de negocio: alto	Puntos estimados: 1
Nivel de complejidad: alto	Iteración asignada: 1ra iteración
Descripción: el administrador se autentica en el sitio web y se muestra la página principal de administración de las fuentes del sitio. Cada una hora, mientras el administrador este autenticado, el sitio web extrae los RSS de las fuentes que existen en la base de datos y actualiza las noticias en la página principal del sitio.	
Observaciones: el usuario tiene que estar autenticado como administrador y mantenerse conectado en el sitio web por al menos una hora.	

Tabla 20: HU2 Mostrar noticias similares.

Historia de usuario	
No: 2	Nombre: Mostrar noticias similares
Usuario: Anónimo y Autenticado	
Prioridad de negocio: alto	Puntos estimados: 1
Nivel de complejidad: media	Iteración asignada: 1ra iteración
Descripción: el usuario selecciona el título de una noticia y se muestra la noticia con todo su contenido, luego se muestra: "Noticias relacionadas" y debajo los enlaces a las noticias similares a la noticia seleccionada en al menos un 60%.	
Observaciones:	

Tabla 21: HU3 Mostrar noticias.

Historia de usuario	
No: 3	Nombre: Mostrar noticias
Usuario: Anónimo y Autenticado	
Prioridad de negocio: alto	Puntos estimados: 1

Nivel de complejidad: media	Iteración asignada: 1ra iteración
Descripción: cuando un usuario accede al sitio se muestra en la página principal las noticias ordenadas por cantidad de votos y comentarios.	
Observaciones:	

Tabla 22: HU4 Autenticar usuario por LDAP.

Historia de usuario	
No: 4	Nombre: Autenticar usuario por LDAP
Usuario: Anónimo	
Prioridad de negocio: alto	Puntos estimados: 1
Nivel de complejidad: alto	Iteración asignada: 1ra iteración
Descripción: al acceder a la aplicación, el usuario dispondrá de la opción “Autenticarse”, al acceder a esta opción se muestra una ventana modal con los campos usuario y contraseña, el usuario introduce el usuario y la contraseña del domino si los datos son correctos, se muestra la página principal del sitio.	
Observaciones: el ordenador del usuario debe estar conectado a la Red.	

Tabla 23: HU5 Votar noticia.

Historia de usuario	
No: 5	Nombre: Votar noticia
Usuario: Autenticado	
Prioridad de negocio: alto	Puntos estimados: 1
Nivel de complejidad: medio	Iteración asignada: 1ra iteración
Descripción: al acceder a la página principal aparecen debajo de la descripción de la noticia los botones para votar positivamente (mano con pulgar hacia arriba) o negativamente (mano con pulgar hacia abajo) por las noticias. Al dar clic sobre el botón para votar positivamente se incrementa en 1 la cantidad de votos, si se da clic sobre el botón para votar negativamente se resta 1 a la cantidad de votos de la noticia.	
Observaciones: el usuario tiene que estar autenticado	

Tabla 24: HU6 Comentar noticia.

Historia de usuario	
No: 6	Nombre: Comentar noticia
Usuario: Autenticado	
Prioridad de negocio: alto	Puntos estimados: 1
Nivel de complejidad: medio	Iteración asignada: 1ra iteración
Descripción: al acceder a la página principal aparece debajo de la descripción de la noticia el botón para comentar por las noticias. Al dar clic sobre el botón se muestran, de existir, los comentarios de la noticia y el campo para comentar, el usuario introduce el comentario y presiona la opción “Guardar”, se muestra el comentario y se incrementa la cantidad de comentarios de la noticia.	
Observaciones: el usuario tiene que estar autenticado	

Tabla 25: HU7 Gestionar contactos.

Historia de usuario	
No: 7	Nombre: Gestionar contactos
Usuario: Autenticado	
Prioridad de negocio: alto	Puntos estimados: 1
Nivel de complejidad: bajo	Iteración asignada: 2da iteración
Descripción: al acceder a la opción “Mis contactos” se muestran los contactos del usuario que hasta autenticado en el sitio, el usuario puede eliminar un contacto dando clic en la opción “Eliminar” y puede insertar un contacto nuevo al dar clic sobre la opción “Insertar contacto”, al realizar esta operación se muestra el campo para introducir el contacto y al presionar la opción “Guardar” se muestra el nuevo contacto insertado en la lista de contactos del usuario.	
Observaciones: el usuario tiene que estar autenticado.	

Tabla 26: HU8 Gestionar marcador social.

Historia de usuario	
No: 8	Nombre: Gestionar marcador social
Usuario: Autenticado	
Prioridad de negocio: alto	Puntos estimados: 1

Nivel de complejidad: bajo	Iteración asignada: 2da iteración
Descripción: al acceder a la opción “Mis marcadores” se muestran los marcadores del usuario que hasta autenticado en el sitio, el usuario puede eliminar un marcador dando clic en la opción “Eliminar”. Para insertar un nuevo marcador, en la página principal debajo de la descripción de las noticias se muestra el botón para guardar marcador (estrella), al dar clic sobre el botón, se guarda el marcador y luego la acceder a la opción “Mis marcadores” aparece el nuevo marcador guardado.	
Observaciones: El usuario tiene que estar autenticado.	

Tabla 27: HU9 Administrar fuente.

Historia de usuario	
No: 9	Nombre: Administrar fuente
Usuario: Todos	
Prioridad de negocio: medio	Puntos estimados: 1
Nivel de complejidad: bajo	Iteración asignada: 2da iteración.
Descripción: al acceder a la página principal se muestra “Sugerir fuente” y debajo los campos para introducir el nombre de la fuente y la URL, se introducen los datos y presiona la opción “Enviar”. EL administrador al acceder a la opción “Fuentes sugeridas”, visualiza las fuentes sugeridas por los usuarios y puede eliminar una fuente dando clic en la opción “Eliminar” o insertarla dando clic en la opción “Insertar”	
Observaciones: el usuario tiene que estar autenticado como administrador para insertar, activar o desactivar fuentes. Los administradores pueden insertar o no la fuente de noticia sugerida por el usuario.	

Tabla 28: HU5 Insertar categoría.

Historia de usuario	
No: 5	Nombre: Insertar categoría
Usuario: Anónimo y Autenticado	
Prioridad de negocio: alto	Puntos estimados: 1
Nivel de complejidad: medio	Iteración asignada: 2da iteración
Descripción: al dar clic sobre el título de una noticia se muestra la categoría de la noticia, de no tener ninguna categoría se muestra el botón: Insertar categoría, al dar clic sobre este, aparece	

un campo para introducir la categoría que el usuario desea que tenga la noticia, presiona la opción guardar y se muestra la noticia con la categoría insertada.

Observaciones:

Tabla 29: HU11 Mostrar tendencias.

Historia de usuario	
No: 11	Nombre: Mostrar tendencias
Usuario: Anónimo y Autenticado	
Prioridad de Negocio: bajo	Puntos estimados: 1
Nivel de Complejidad: medio	Iteración asignada: 3ra iteración
Descripción: al acceder a la opción “Tendencias” se muestran las fuentes que más noticias publican, las noticias más comentadas , las más votadas y los usuarios que más comentan	
Observaciones:	

Tabla 30: HU12 Buscar noticias.

Historia de usuario	
No: 12	Nombre: Buscar noticias
Usuario: Anónimo y Autenticado	
Prioridad de negocio: alto	Puntos estimados: 1
Nivel de complejidad: alto	Iteración asignada: 3ra iteración
Descripción: El usuario puede buscar noticias introduciendo una palabra en el campo que se muestra en el menú principal del sitio y presionar “Enter”, se muestran todas las noticias que cumplan con el criterio de búsqueda especificado por el usuario.	
Observaciones:	

Tabla 31: HU13 Mostrar estadísticas.

Historia de usuario	
No: 13	Nombre: Mostrar estadísticas
Usuario: Todos	
Prioridad de negocio: alto	Puntos estimados: 1
Nivel de complejidad: alto	Iteración asignada: 3ra iteración

Descripción: Al acceder a la página principal se muestran la cantidad de noticias, fuentes y categorías que existen en el sitio hasta ese momento.
Observaciones:

Tabla 32: HU14 Configurar preferencias.

Historia de usuario	
No: 14	Nombre: Configurar preferencias
Usuario: Autenticado	
Prioridad de negocio: alto	Puntos estimados: 1
Nivel de complejidad: alto	Iteración asignada: 3ra iteración
Descripción: el usuario accede a la opción "Preferencias", se muestran todas las fuentes activas, si el usuario desea ver las noticias que publica un sitio da clic en la opción "Si", sino da clic en la opción "No".	
Observaciones: el usuario tiene que estar autenticado.	

Anexo 3. Tarjetas CRC de la propuesta de solución

Tabla 33: Tarjeta CRC Readrss.

Readrss	
Responsabilidades	Colaboradores
Obtener fuentes Insertar noticias Insertar noticias similares Mostrar tiempo en que demora extraer RSS Método(index)	Benchmark Date Simplepie Dao Fuente_dao Noticia_dao Similar_dao

Tabla 34: Tarjeta CRC Test_autenticado.

Test_autenticado	
Responsabilidades	Colaboradores
Cargar perfil de usuario: (index) viewNoticia getNoticias getNoticiasFuente getFuente updateNoticiaVotos getNoticia1 noticiasXUsuario actualizarEstado	Noticia_dao Service_lib Perfil_dao Fuente_dao Noticia_dao Noticia_fuente_dao Usuario_dao Comentario_dao

Tabla 35: Tarjeta CRC Autenticar por LDAP.

Autenticar	
Responsabilidades	Colaboradores
AutenticarLDAP	Autenticar_usuario_model Autenticar_usuario_view Test_autenticado

Tabla 36: Tarjeta CRC Admin_fuentes_sugeridas.

Fuente	
Responsabilidades	Colaboradores
mostrar_Fuentes_Sugeridas insertar_Fuentes_Sugeridas eliminar_Fuentes_Sugeridas sugerir_Fuente	Fuente_dao Fuente_sugerida_dao Fuente_model

Tabla 37: Tarjeta CRC Marcador.

Marcador social	
Responsabilidades	Colaboradores
EliminarMarcador AgregarMarcador MostrarMarcador	Marcador_social_dao Marcador_social_model

Tabla 40: Caso de prueba unitaria No.1.

Número del caso de prueba unitaria					
Iteración: 1					
Descripción: Se prueba la obtención de noticias					
Responsable:					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
getNoticia ()	getNoticias ()	New Noticia _Model (new Noticia _model 1, titulo1, NULL, 1, 1, 1, 0, 2015- 05-05 20:31:0 0, NULL, 0, Drupal eros, NULL, "http:// drupale ros.uci. cu/?q=	Se obtiene una noticia	Tests: 1, Assertions: 1, Failures 0	Método de la clase Test_autentica do

		node/560");)			
--	--	--------------	--	--	--

Tabla 41: Caso de prueba unitaria No.2.

Caso de prueba unitaria No.2					
Iteración: 1					
Descripción: Se prueba la votar positivamente					
Responsable: Ailyn Pérez Aguila					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
updateNoticia Votos()	updateNoticia Votos()	1,1	Se actualizan los votos de la noticia con id=1	Tests: 1, Assertions: 1, Failures 0	Método de la clase Test_authentication

Tabla 42: Caso de prueba unitaria No.3.

Caso de prueba unitaria No.3					
Iteración: 1					
Descripción: Se prueba la votar negativamente					
Responsable: Ailyn Pérez Aguila					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones

updateNoticia Votos()	updateNoticia a Votos()	1,-1	Se actualizan los votos de la noticia con id=1	Tests: 1, Assertions : 1, Failures 0	Método de la clase Test_autenticad o
--------------------------	-------------------------------	------	---	---	---

Anexo 4. Tareas de ingeniería, primera iteración

Tabla 38. Tarea 1: Establecer estándar de código y de comentarios.

Tarea No.1		HU1
Nombre de tarea: Establecer estándar de código y de comentarios.		
Tipo de tarea: Configuración	Estimación: 1 día	
Fecha de inicio: 9 de febrero del 2015	Fecha de fin: 9 de febrero del 2015	
Programador responsable: Ailyn Pérez Aguila		
Descripción: redactar el estándar de código a emplear siguiendo los utilizados internacionalmente en proyectos de PHP. Consultar documentación en la Red par a la definición del mismo.		

Tabla 39. Tarea 2: Implementar las clases principales del núcleo de la aplicación.

Tarea No.2		HU1
Nombre de tarea: Implementar las clases principales del núcleo de la aplicación.		
Tipo de tarea: Implementación	Estimación: 1 días	
Fecha de inicio: 10 de febrero del 2015	Fecha de fin: 10 de febrero del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: implementar las clases Test.php y Test_autenticado.php las cuales serán empleadas para la posterior implementación de las demás funcionalidades.		

Tabla 40. Tarea 3: Diseñar página principal.

Tarea No.3		HU1
Nombre de tarea: Diseñar interfaz de usuario.		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 11 de febrero del 2015	Fecha de fin: 11 de febrero del 2015	
Programador responsable: Ailyn Pérez Aguila		
Descripción: diseñar página de principal en el head_view y footer_view siguiendo las tendencias actuales.		

Tabla 41. Tarea 4: Extraer RSS.

Tarea No.4		HU1
Nombre de tarea: Extraer RSS		

Tipo de tarea: Implementación	Estimación: 2 día
Fecha de inicio: 12 de febrero del 2015	Fecha de fin: 13 de febrero del 2015
Programador responsable: Ailyn Pérez Aguila	
Descripción: : implementar la clase Readrss.php que permitirá mostrar las noticias de las fuentes insertadas en la base de datos	

Tabla 42. Tarea 5: Realizar prueba unitaria: Test_mostrar_similares.

Tarea No.5		HU2
Nombre de tarea: Realizar prueba unitaria: Test_mostrar_similares.		
Tipo de tarea: Implementación	Estimación: 2 día	
Fecha de inicio: 16 de febrero del 2015	Fecha de fin: 17 de febrero del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: implementar función Test_mostrar_similares en la clase Test_Readrss.php para probar la obtención de noticias similares.		

Tabla 43. Tarea 6: Buscar noticias similares.

Tarea No.6		HU2
Nombre de tarea: Buscar noticias similares.		
Tipo de tarea: Implementación	Estimación: 3 día	
Fecha de inicio: 18 de febrero del 2015	Fecha de fin: 20 de febrero del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: adicionar procedimientos necesarios a la función mostrar_similares en la clase Readrss.php para realizar una búsqueda de noticias similares		

Tabla 44. Tarea No.7: Realizar prueba unitaria: Test_mostrar_noticias.

Tarea No.7		HU3
Nombre de tarea: Realizar prueba unitaria: Test_mostrar_noticias.		
Tipo de tarea: Implementación	Estimación: 2 días	
Fecha de inicio: 23 de febrero del 2015	Fecha de fin: 24 de febrero del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		

Descripción: implementar función Test_mostrar_noticias en la clase Test_autenticado.php para probar la obtención de noticias.

Tabla 45. Tarea No.8: Mostrar noticias ordenadas por la cantidad de votos y comentarios.

Tarea No.8		HU3
Nombre de tarea: Mostrar noticias ordenadas por la cantidad de votos y comentarios.		
Tipo de tarea: Implementación	Estimación: 3 días	
Fecha de inicio: 25 de febrero del 2015	Fecha de fin: 27 de febrero del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: adicionar procedimientos necesarios para mostrar las noticias ordenadas por la cantidad de votos y comentarios en la clase Test_autenticado.php. Además se crea View_test.php para mostrar las noticias.		

Tabla 46. Tarea No.9: Realizar prueba unitaria: Test_ autenticar_LDAP.

Tarea No.9		HU4
Nombre de tarea: Realizar prueba unitaria: Test_ autenticar_LDAP.		
Tipo de tarea: Implementación	Estimación: 2 días	
Fecha de inicio: 2 de marzo del 2015	Fecha de fin: 3 de marzo del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: implementar función Test_ autenticar_LDAP en la clase Test_autenticado.php para probar autenticar un usuario.		

Tabla 47. Tarea No.10: Autenticar usuario por LDAP.

Tarea No.10		HU4
Nombre de tarea: Autenticar usuario por LDAP.		
Tipo de tarea: Implementación	Estimación: 3 días	
Fecha de inicio: 4 de marzo del 2015	Fecha de fin: 6 de marzo del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: adicionar procedimientos necesarios a la función autenticar en la clase Test_autenticado.php para autenticar un usuario. Además se crea en el		

Autenticar_view. Php con el uso del framework Bootstrap el formulario formlogin, para mostrar la información.

Tabla 48. Tarea No.11: Realizar prueba unitaria: Test_ votarP.

Tarea No.11		HU5
Nombre de tarea: Realizar prueba unitaria: Test_ votarP.		
Tipo de tarea: Implementación	Estimación: 1 días	
Fecha de inicio: 9 de marzo del 2015	Fecha de fin: 9 de marzo del 2015	
Programador responsable: Ailyn Pérez Aguila		
Descripción: implementar función Test_ votarP en la clase Test_autenticado.php para probar votar por una noticia.		

Tabla 49. Tarea No.12: Votar positivamente.

Tarea No.12		HU5
Nombre de tarea: Votar positivamente		
Tipo de tarea: Implementación	Estimación: 1 días	
Fecha de inicio: 10 de marzo del 2015	Fecha de fin: 10 de abril del 2015	
Programador responsable: Ailyn Pérez Aguila		
Descripción: adicionar procedimientos necesarios a la función updateNoticiaVotos en la clase Test_autenticado.php para votar una noticia. Además se actualiza en el View_test.php la cantidad de votos.		

Tabla 50. Tarea No.13: Realizar prueba unitaria: Test_ votarN.

Tarea No.13		HU5
Nombre de tarea: Realizar prueba unitaria: Test_ votarN.		
Tipo de tarea: Implementación	Estimación: 1 días	
Fecha de inicio: 11 de marzo del 2015	Fecha de fin: 11 de marzo del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: implementar función Test_ votarN en la clase Test_autenticado.php para probar votar una noticia.		

Tabla 51. Tarea No.14: Votar negativamente.

Tarea No.14		HU5
Nombre de tarea: Votar negativamente		
Tipo de tarea: Implementación	Estimación: 2 días	
Fecha de inicio: 12 de marzo del 2015	Fecha de fin: 13 de marzo del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción Adicionar procedimientos necesarios a la función updateNoticiaVotos en la clase Test_autenticado.php para votar una noticia. Además se actualiza en el View_test.php la cantidad de votos.		

Tabla 52. Tarea No.15: Realizar prueba unitaria: Test_comentar.

Tarea No.15		HU6
Nombre de tarea: Realizar prueba unitaria: Test_comentar.		
Tipo de tarea: Implementación	Estimación: 2 día	
Fecha de inicio: 16 de marzo del 2015	Fecha de fin: 17 de marzo del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: implementar función Test_comentar en la clase Test_Fuente.php para probar comentar una noticia.		

Tabla 53. Tarea No.16: Comentar noticia.

Tarea No.16		HU6
Nombre de tarea: Comentar noticia		
Tipo de tarea: Implementación	Estimación: 3 día	
Fecha de inicio: 18 de marzo del 2015	Fecha de fin: 20 de marzo del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: adicionar procedimientos necesarios a la función comentar_noticia en la clase Fuente.php comentar una noticia. Además se muestra en comentarios_view.php los comentarios que tiene la noticia y en la vista insertar_comentarios con el uso del framework Bootstrap el formulario formcomentar, para mostrar el textarea para que el usuario introduzca su comentario y el botón Enviar comentario.		

Anexo 5. Casos de prueba unitarias de la primera iteración

Tabla 54: Caso de prueba unitaria No.1.

Caso de prueba unitaria No.1					
Iteración: 1					
Descripción: Se prueba mostrar noticias similares					
Responsable:					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
mostrar_similares()	Test_mostrar_similares()	New Noticia_Model (1, titulo1, NULL, 1, 1, 1, 0, 2015-05-05 20:31:00, NULL, 0, Drupal eros, NULL, "http://drupaleros.uci.cu/?q=	Se obtiene una noticia	Tests: 1, Assertions: 1, Failures 0	Método de la clase Test_authenticado

		node/560")			
--	--	------------	--	--	--

Tabla 55: Caso de prueba unitaria No.2.

Caso de prueba unitaria No.2					
Iteración: 1					
Descripción: Se prueba mostrar noticias					
Responsable:					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
mostrar_noticias()	Test_mostrar_noticias()	New Noticia_Model (1, titulo1, NULL, 1, 1, 1, 0, 2015-05-05 20:31:00, NULL, 0, Drupal eros, NULL, "http://drupaleros.uci.cu/?q=	Se muestra una noticia	Tests: 1, Assertions: 1, Failures 0	Método de la clase Test_authenticado

		node/5 60")			
--	--	----------------	--	--	--

Tabla 56: Caso de prueba unitaria No.3.

Caso de prueba unitaria No.3					
Iteración: 1					
Descripción: Se prueba la autenticar un usuario por LDAP					
Responsable: Ailyn Pérez Aguila					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
autenticar_LDAP()	Test_autenticar_LDAP()	Aguila,123456789	Se autentica el usuario aaguila en el sitio.	Tests: 1, Assertions: 1, Failures: 0	Método de la clase Test_autenticado

Tabla 57: Caso de prueba unitaria No.4.

Caso de prueba unitaria No.4					
Iteración: 1					
Descripción: Se prueba votar positivamente					
Responsable: Ailyn Pérez Aguila					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones

votarP()	Test_votarP()	new Noticia_model(1, titulo1, NULL, 1, 1, 1, 0, 2015-05-05 20:31:00, NULL, 0, Drupaleros, NULL, "http://drupaleros.uci.cu/?q=node/560"),1	Se incrementan en 1, los votos de la noticia.	Tests: 1, Assertions: 1, Failures 0	Método de la clase Test_authenticado
----------	---------------	---	---	-------------------------------------	--------------------------------------

Tabla 58: Caso de prueba unitaria No.5.

Caso de prueba unitaria No.5					
Iteración: 1					
Descripción: Se prueba la votar negativamente					
Responsable: Ailyn Pérez Aguila					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
votarN()	Test_votarN()	new Noticia_model(1, titulo1, NULL, 1, 1, 1, 0, 2015-05-05 20:31:00, NULL, 0, Drupaleros, NULL, "http://drupaleros.uci.cu/?q=node/560"),1,-1	Se resta 1 a los votos de la noticia	Tests: 1, Assertions: 1, Failures 0	Método de la clase Test_authenticado

Tabla 59: Caso de prueba unitaria No.6.

Caso de prueba unitaria No.6					
Iteración: 1					
Descripción: Se prueba la comentar					
Responsable: Ailyn Pérez Aguila					

Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
Comentar()	Test_comentar()	new Noticia_model(1, titulo1, NULL, 1, 1, 1, 0, 2015-05-05 20:31:00, NULL, 0, Drupaleros, NULL, "http://drupaleros.uci.cu/?q=node/560"),comentario	Se inserta el comentario a la noticia	Tests: 1, Assertions: 1, Failures 0	Método de la clase Comentario

Anexo 6. Casos de prueba de aceptación, versión 0.1

Tabla 60: Caso de prueba de aceptación 2 HU Mostrar noticias similares

Caso de prueba de aceptación	
Código: PA2HU2	Historia de usuario: Mostrar noticias similares
Nombre: Mostrar noticias similares.	
Descripción: Se buscan las noticias similares y se guardan en la base de datos.	
Condición de ejecución:	
Entrada/Pasos de ejecución: Cuando un usuario marca una noticia, se muestran las otras noticias similares a ella si existen.	
Resultado esperado: Se muestran las noticias similares.	

Tabla 61: Caso de prueba de aceptación 3 HU Mostrar noticias

Caso de prueba de aceptación	
Código: PA3HU3	Historia de usuario: Mostrar noticias
Nombre: Mostrar noticias.	
Descripción: Se muestran las noticias ordenadas por la cantidad de votos y comentarios.	
Condición de ejecución:	
Entrada/Pasos de ejecución: Cuando un usuario accede al sitio, se muestran las noticias ordenadas por votos y comentarios	
Resultado esperado: Se muestran las noticias ordenadas por cantidad de votos y comentarios.	

Tabla 62: Caso de prueba de aceptación 4 HU Autenticar usuario por LDAP.

Caso de prueba de aceptación	
Código: PA4HU4	Historia de usuario: Autenticar usuario por LDAP
Nombre: Autenticar usuario por LDAP	

Descripción: Se autentica un usuario
Condición de ejecución: El usuario y la contraseña tiene que ser correctos.
Entrada/Pasos de ejecución: El usuario entra su nombre de usuario y contraseña Da clic en la opción Aceptar.
Resultado esperado: Se autentica al usuario permitiendo que entre a su perfil.

Tabla 63: Caso de prueba de aceptación 5 HU Votar noticia.

Caso de prueba de aceptación	
Código: PA5HU5	Historia de usuario: Votar noticia
Nombre: Votar positivamente	
Descripción: Se vota positivamente por una noticia	
Condición de ejecución: El usuario debe estar autenticado. El usuario solo puede votar una sola vez.	
Entrada/Pasos de ejecución: El usuario selecciona la opción votar noticia positivamente (Mano con dedo anular apuntando hacia arriba).	
Resultado esperado: Se vota positivamente por la noticia.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 64: Caso de prueba de aceptación 6 HU Votar noticia.

Caso de prueba de aceptación	
Código: PA6HU5	Historia de usuario: Votar noticia
Nombre: Votar negativamente	
Descripción: Se vota negativamente por una noticia	
Condición de ejecución: El usuario debe estar autenticado. El usuario solo puede votar una sola vez.	
Entrada/Pasos de ejecución: El usuario selecciona la opción votar noticia negativamente (Mano con dedo anular apuntando hacia abajo).	
Resultado esperado: Se vota negativamente por la noticia.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 65: Caso de prueba de aceptación 7 HU Comentar noticia.

Caso de prueba de aceptación	
Código: PA7HU6	Historia de usuario: Comentar noticia
Nombre: Comentar noticia.	
Descripción: Se comenta una noticia	
Condición de ejecución: el usuario debe estar autenticado	
Entrada/Pasos de ejecución: El usuario introduce su comentario. Da clic en la opción Enviar comentario.	
Resultado esperado: Se comenta la noticia.	
Evaluación de la prueba: Prueba satisfactoria.	

Anexo 7. Tareas de ingeniería, segunda iteración

Tabla 66. Tarea No.17: Realizar prueba unitaria: Test_ insertar_contacto.

Tarea No.17		HU 7
Nombre de tarea: Realizar prueba unitaria: Test_ insertar_contacto.		
Tipo de tarea: Implementación	Estimación: 1 días	
Fecha de inicio: 23 de marzo del 2015	Fecha de fin: 23 de marzo del 2015	
Programador responsable: Ailyn Pérez Aguila		
Descripción: implementar función Test_ insertar_contacto en la clase Test_Insertar_contacto.php para probar insertar un contacto en la lista de contactos de un usuario específico.		

Tabla 67. Tarea No.18: Insertar contacto.

Tarea No.18		HU 7
Nombre de tarea: Insertar contacto		
Tipo de tarea: Implementación	Estimación: 1 días	
Fecha de inicio: 23 de marzo del 2015	Fecha de fin: 23 de marzo del 2015	
Programador responsable: Ailyn Pérez Aguila		
Descripción: adicionar procedimientos necesarios a la función insertar_contacto en la clase Insertar_contacto.php para insertar un contacto en la lista de contactos de un usuario específico. Además se crea la vista contactos_view.php para mostrar los contactos del usuario.		

Tabla 68. Tarea No.19: Realizar prueba unitaria: Test_ eliminar_contacto.

Tarea No.19		HU 7
Nombre de tarea: Realizar prueba unitaria: Test_ eliminar_contacto.		
Tipo de tarea: Implementación	Estimación: 1 días	
Fecha de inicio: 24 de marzo del 2015	Fecha de fin: 24 de marzo del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: implementar función Test_eliminar_contacto en la clase Test_Contacto.php probar para eliminar un contacto de la lista de contactos de un usuario específico.		

Tabla 69. Tarea No.20: Eliminar contacto.

Tarea No.20		HU 7
Nombre de tarea: Eliminar contacto		
Tipo de tarea: Implementación	Estimación: 1 días	
Fecha de inicio: 25 de marzo del 2015	Fecha de fin: 25 de marzo del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: adicionar procedimientos necesarios a la función eliminar_contacto en la clase Contacto.php para eliminar un contacto de la lista de contactos de un usuario específico. Además se crea el botón Eliminar con el uso del framework Bootstrap.		

Tabla 70. Tarea No.21: Realizar prueba unitaria: Test_ mostrar_contacto.

Tarea No.21		HU 7
Nombre de tarea: Realizar prueba unitaria: Test_ mostrar_contacto.		
Tipo de tarea: Implementación	Estimación: 1 días	
Fecha de inicio: 26 de marzo del 2015	Fecha de fin: 26 de marzo del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: implementar función Test_mostrar_contacto en la clase Test_Contacto.php para probar obtener un contacto.		

Tabla 71. Tarea No.22: Mostrar contacto.

Tarea No.22		HU 7
Nombre de tarea: Mostrar contacto		
Tipo de tarea: Implementación	Estimación: 1 días	
Fecha de inicio: 27 de marzo del 2015	Fecha de fin: 27 de marzo del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: adicionar procedimientos necesarios a la función mostrar_contacto en la clase Contacto.php para mostrar un contacto en la lista de contactos de un usuario específico.		

Tabla 72. Tarea No.23: Realizar prueba unitaria: Test_ insertar_marcador.

Tarea No.23		HU 8
Nombre de tarea: Realizar prueba unitaria: Test_ insertar_marcador.		

Tipo de tarea: Implementación	Estimación: 1 día
Fecha de inicio: 30 de marzo del 2015	Fecha de fin: 30 de marzo del 2015
Programador responsable: Ailyn Pérez Aguila	
Descripción: implementar función insertar_marcador en la clase Test_Marcador.php para probar insertar un marcador social.	

Tabla 73. Tarea No.24: Insertar marcador social.

Tarea No.24		HU 8
Nombre de tarea: Insertar marcador social		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 30 de marzo del 2015	Fecha de fin: 30 de marzo del 2015	
Programador responsable: Ailyn Pérez Aguila		
Descripción: adicionar procedimientos necesarios a la función insertar_marcador en la clase Marcador.php para insertar un marcador social en la lista de marcadores sociales de un usuario específico. Además se crea marcadores_view.php para mostrar los marcadores del usuario.		

Tabla 74. Tarea No.25: Realizar prueba unitaria: Test_compartir_marcador.

Tarea No.25		HU 8
Nombre de tarea: Realizar prueba unitaria: Test_compartir_marcador.		
Tipo de tarea: Implementación	Estimación: 1 días	
Fecha de inicio: 31 de marzo del 2015	Fecha de fin: 31 de marzo del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: implementar función Test_compartir_marcador en la clase Test_Marcador.php para probar compartir marcador social con uno o más usuarios.		

Tabla 75. Tarea No.26: Compartir marcador social.

Tarea No.26		HU 8
Nombre de tarea: Compartir marcador social		
Tipo de tarea: Implementación	Estimación: 1 días	
Fecha de inicio: 31 de marzo del 2015	Fecha de fin: 31 de marzo del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		

Descripción: adicionar procedimientos necesarios a la función compartir_marcador en la clase Marcador.php para compartir marcador social con uno o más usuarios. Además se crea marca_comp_view.php para mostrar el marcador a compartir y los contactos con los que se puede compartir el marcador.

Tabla 76. Tarea No.27: Realizar prueba unitaria: Test_ eliminar_marcador.

Tarea No.27		HU 8
Nombre de tarea: Realizar prueba unitaria: Test_ eliminar_marcador.		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 1 de abril del 2015	Fecha de fin: 1 de abril del 2015	
Programador responsable: Ailyn Pérez Aguila		
Descripción: implementar función Test_eliminar_marcador en la clase Test_Marcador.php para probar eliminar un marcador social.		

Tabla 77. Tarea No.28: Eliminar marcador social.

Tarea No.28		HU 8
Nombre de tarea: Eliminar marcador social		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 1 de abril del 2015	Fecha de fin: 1 de abril del 2015	
Programador responsable: Ailyn Pérez Aguila		
Descripción: Adicionar procedimientos necesarios a la función eliminar_marcador en la clase Marcador.php para eliminar un marcador social de la lista de marcadores sociales de un usuario específico. Además se crea boton Eliminar con el uso del framework Bootstrap.		

Tabla 78. Tarea No.29: Realizar prueba unitaria: Test_ mostrar_marcador.

Tarea No.29		HU 8
Nombre de tarea: Realizar prueba unitaria: Test_ mostrar_marcador.		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 2 de abril del 2015	Fecha de fin: 2 de abril del 2015	
Programador responsable: Ailyn Pérez Aguila		

Descripción: implementar función Test_mostrar_marcador en la clase Test_Marcador.php para mostrar un marcador social en la lista de marcadores sociales de un usuario específico.

Tabla 79. Tarea No.30: Mostrar marcador social.

Tarea No.30		HU 8
Nombre de tarea: Mostrar marcador social		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 3 de abril del 2015	Fecha de fin: 3 de abril del 2015	
Programador responsable: Ailyn Pérez Aguila		
Descripción: adicionar procedimientos necesarios a la función mostrar_marcador en la clase Marcador.php para mostrar un marcador social en la lista de marcadores sociales de un usuario específico.		

Tabla 80. Tarea No.31: Realizar prueba unitaria: Test_ activar_fuente.

Tarea No.31		HU 9
Nombre de tarea: Realizar prueba unitaria: Test_ activar_fuente.		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 6 de abril del 2015	Fecha de fin: 6 de abril del 2015	
Programador responsable: Ailyn Pérez Aguila		
Descripción: implementar función Test_ activar_fuente en la clase Test_Fuente.php para probar activar fuentes de RSS del sitio web.		

Tabla 81. Tarea No.32: Activar fuentes.

Tarea No.32		HU 9
Nombre de tarea: Activar fuentes.		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 6 de abril del 2015	Fecha de fin: 6 de abril del 2015	
Programador responsable: Ailyn Pérez Aguila		
Descripción: adicionar procedimientos necesarios a la función activar en la clase Fuente.php para activar fuentes de RSS del sitio web. Además se crea en el botón		

activar con el uso del framework Bootstrap el formulario form_activar_fuente, para mostrar la información.

Tabla 82. Tarea No.33: Realizar prueba unitaria: Test_desactivar_fuente.

Tarea No.33		HU 9
Nombre de tarea: Realizar prueba unitaria: Test_desactivar_fuente.		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 7 de abril del 2015	Fecha de fin: 7 de abril del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: implementar función Test_desactivar_fuente en la clase Test_Fuente.php para probar desactivar fuentes de RSS del sitio web.		

Tabla 83. Tarea No.34: Desactivar fuentes.

Tarea No.34		HU 9
Nombre de tarea: Desactivar fuentes.		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 7 de abril del 2015	Fecha de fin: 7 de abril del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: adicionar procedimientos necesarios a la función desactivar en la clase Fuente.php para desactivar fuentes de RSS del sitio web. Además se crea en el botón activar con el uso del framework Bootstrap.		

Tabla 84. Tarea No.35: Realizar prueba unitaria: Test_sugerir_fuente.

Tarea No.35		HU 9
Nombre de tarea: Realizar prueba unitaria: Test_sugerir_fuente.		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 8 de abril del 2015	Fecha de fin: 8 de abril del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: implementar función Test_sugerir_fuente en la clase Test_Fuente.php para sugerir fuentes de RSS de sitios web		

Tabla 85. Tarea No.36: Sugerir fuentes.

Tarea No.36		HU 9
Nombre de tarea: Sugerir fuentes.		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 8 de abril del 2015	Fecha de fin: 8 de abril del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: adicionar procedimientos necesarios a la función sugerir_fuente en la clase Fuente.php para sugerir fuentes de RSS de sitios web		

Tabla 86. Tarea No.37: Realizar prueba unitaria: Test_ insertar_fuente.

Tarea No.37		HU 9
Nombre de tarea: Realizar prueba unitaria: Test_ insertar_fuente.		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 9 de abril del 2015	Fecha de fin: 9 de abril del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: implementar función Test_insertar_fuente en la clase Test_Fuente.php para probar insertar fuentes de RSS del sitio web.		

Tabla 87. Tarea No.38: Insertar fuentes.

Tarea No.38		HU 9
Nombre de tarea: Insertar fuentes.		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 9 de abril del 2015	Fecha de fin: 9 de abril del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: adicionar procedimientos necesarios a la función insertar_fuente en la clase Fuente.php para insertar fuentes de RSS.		

Tabla 88. Tarea No.39: Realizar prueba unitaria: Test_ mostrar_fuente.

Tarea No.39		HU 9
Nombre de tarea: Realizar prueba unitaria: Test_ mostrar_fuente.		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 10 de abril del 2015	Fecha de fin: 10 de abril del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		

Descripción: implementar función Test_mostrar_fuente en la clase Test_Fuente.php para probar obtener fuentes de RSS.

Tabla 89. Tarea No.40: Mostrar fuentes.

Tarea No.40		HU 9
Nombre de tarea: Mostrar fuentes.		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 10 de abril del 2015	Fecha de fin: 10 de abril del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: Adicionar procedimientos necesarios a la función mostrar_fuente en la clase Fuente.php para mostrar las fuentes de RSS que contiene el sitio web.		

Tabla 90. Tarea No.41: Realizar prueba unitaria: Test_insertar_categoria.

Tarea No.41		HU10
Nombre de tarea: Realizar prueba unitaria: Test_insertar_categoria.		
Tipo de tarea: Implementación	Estimación: 2 días	
Fecha de inicio: 13 de abril del 2015	Fecha de fin: 14 de abril del 2015	
Programador responsable: Ailyn Pérez Aguila		
Descripción: implementar Test_función insertar_Categoria en la clase Test_Categoria.php para probar insertar una categoría a una noticia.		

Tabla 91. Tarea No.42: Insertar categoría.

Tarea No.42		HU10
Nombre de tarea: Insertar categoría.		
Tipo de tarea: Implementación	Estimación: 3 días	
Fecha de inicio: 15 de abril del 2015	Fecha de fin: 17 de abril del 2015	
Programador responsable: Ailyn Pérez Aguila		
Descripción: adicionar procedimientos necesarios a la función insertar_Categoria en la clase Categoria.php para insertar una categoría a una noticia. Además se crea en el New_view. Php con el uso del framework Bootstrap el botón Insertar categoría y se crea la vista insertar_categoria_view para que el usuario introduzca la categoría.		

Anexo 8. Casos de prueba unitarias de la segunda iteración

Tabla 92: Caso de prueba unitaria No.7.

Caso de prueba unitaria No.7					
Iteración: 2					
Descripción: Se prueba insertar un contacto					
Responsable: Ailyn Pérez Aguila					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
insertar_contacto ()	Test_insertar_contacto()	aaguila, yrsanchez	Se inserta el contacto yrsanchez a la lista de contactos da aaguila	Tests: 1, Assertion: 1, Failures 0	Método de la clase Contacto

Tabla 93: Caso de prueba unitaria No.8.

Caso de prueba unitaria No.8					
Iteración: 2					
Descripción: Se prueba eliminar un contacto					
Responsable: Ailyn Pérez Aguila					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones

eliminar_contacto ()	Test_eliminar_contacto()	aaguila, yrsanchez	Se elimina el contacto yrsanchez de la lista de contactos da aaguila	Tests: 1, Assertion s: 1, Failures 0	Método de la clase Contacto
----------------------	--------------------------	--------------------	--	--------------------------------------	-----------------------------

Tabla 94: Caso de prueba unitaria No.9.

Caso de prueba unitaria No.9					
Iteración: 2					
Descripción: Se prueba mostrar un contacto					
Responsable: Ailyn Pérez Aguila					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
mostrar_contacto ()	Test_mostrar_contacto()	yrsanchez	Se muestra el contacto yrsanchez	Tests: 1, Assertion s: 1, Failures 0	Método de la clase Contacto

Tabla 95: Caso de prueba unitaria No.10.

Caso de prueba unitaria No.10					
Iteración: 2					
Descripción: Se prueba insertar un marcador					
Responsable: Ailyn Pérez Aguila					

Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
insertar_marcador ()	Test_insertar_marcador()	new Noticia_model(1, titulo1, NULL, 1, 1, 1, 0, 2015-05-05 20:31:00, NULL, 0, Drupaleros, NULL, "http://drupaleros.uci.cu/?q=node/560"), Aguila	Se agrega la noticia a la lista de marcadores de aaguila	Tests: 1, Assertions: 1, Failures 0	Método de la clase Marcador

Tabla 96: Caso de prueba unitaria No.11.

Caso de prueba unitaria No.11					
Iteración: 2					
Descripción: Se prueba eliminar un marcador					
Responsable: Ailyn Pérez Aguila					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
eliminar_marcador ()	Test_eliminar_marcador()	new Noticia_model(1, titulo1, NULL, 1, 1, 1, 0, 2015-05-05 20:31:00, NULL, 0, Drupaleros, NULL,	Se elimina la noticia de la lista de	Tests: 1, Assertions: 1, Failures 0	Método de la clase Marcador

		"http://drupaleros.uci.cu/?q=node/560"), Aaguila	marcadores de aaguila		
--	--	---	-----------------------	--	--

Tabla 97: Caso de prueba unitaria No.12.

Caso de prueba unitaria No.12					
Iteración: 2					
Descripción: Se prueba compartir un marcador					
Responsable: Ailyn Pérez Aguila					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
compartir _marcador ()	Test_compartir _marcador()	new Noticia_model(1, titulo1, NULL, 1, 1, 1, 0, 2015-05-05 20:31:00, NULL, 0, Drupaleros, NULL, "http://drupaleros.uci.cu/?q=node/560"),aaguila,yrsanchez	Se comparte la noticia de aaguila, con el contacto yrsanchez, mostrando se en su lista de sugerencias	Tests : 1, Assertions : 1, Failures 0	Método de la clase Compartir Marcador

Tabla 98: Caso de prueba unitaria No.13.

Caso de prueba unitaria No.13					
Iteración: 2					
Descripción: Se prueba mostrar un marcador					
Responsable: Ailyn Pérez Aguila					

Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
mostrar_marcador()	Test_mostrar_marcador()	new Noticia_model(1, titulo1, NULL, 1, 1, 1, 0, 2015-05-05 20:31:00, NULL, 0, Drupaleros, NULL, "http://drupaleros.uci.cu/?q=node/560"),aguila	Se muestra la noticia en la lista de marcadores de aguila.	Tests: 1, Assertions: 1, Failures 0	Método de la clase Marcador

Tabla 99: Caso de prueba unitaria No. 14.

Caso de prueba unitaria No.14					
Iteración: 2					
Descripción: Se prueba activar una fuente					
Responsable: Ailyn Pérez Aguila					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
activar_fuente()	Test_activar_fuente()	new Fuente_model (1, Drupaleros, "http://drupaleros.uci.cu/rss.xml", 0)	Se activa la fuente	Tests: 1, Assertions: 1,	Método de la clase Administrador

				Failures 0	
--	--	--	--	---------------	--

Tabla 100: Caso de prueba unitaria No.15.

Caso de prueba unitaria No.15					
Iteración: 2					
Descripción: Se prueba desactivar una fuente					
Responsable: Ailyn Pérez Aguila					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
desactivar_fuente()	Test_desactivar_fuente()	new Fuente_model (1, Drupaleros, "http://drupaleros.uci.cu/rss.xml", 0)	Se desactiva la fuente	Tests: 1, Assertions: 1, Failures 0	Método de la clase Administrador

Tabla 101: Caso de prueba unitaria No.16.

Caso de prueba unitaria No.16					
Iteración: 2					
Descripción: Se prueba sugerir una fuente					
Responsable: Ailyn Pérez Aguila					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones

sugerir_fuente()	Test_sugerir_fuente()	Drupaleros,new Fuente_model (1, Drupaleros, "http://drupaleros.uci.cu/rss.xml", 0)	Se inserta la fuente	Tests: 1, Assertions: 1, Failures 0	Método de la clase Administrador
------------------	-----------------------	--	----------------------	-------------------------------------	----------------------------------

Tabla 102: Caso de prueba unitaria No.17.

Caso de prueba unitaria No.17					
Iteración: 2					
Descripción: Se prueba mostrar una fuente					
Responsable: Ailyn Pérez Aguila					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
mostrar_fuente()	Test_mostrar_fuente()	Drupaleros, new Fuente_model (1, Drupaleros, "http://drupaleros.uci.cu/rss.xml", 0)	Se muestra la fuente	Tests: 1, Assertions: 1, Failures 0	Método de la clase Administrador

Tabla 103: Caso de prueba unitaria No.18.

Caso de prueba unitaria No.18					
Iteración: 2					
Descripción: Se prueba insertar una categoría a una noticia					
Responsable: Ailyn Pérez Aguila					
Funcionalidad	Método Utilizado	Recibe	Respuesta	Respuesta	Observaciones

			esper ada del métod o	real del métod o	
insertar_categoria()	Test_ insertar_categoria()	new Noticia_model(1, titulo1, NULL, 1, 1, 1, 0, 2015-05-05 20:31:00, NULL, 0, Drupaleros, NULL, "http://drupaleros.uci.cu/?q =node/560"),drupal	Se le asigna la catego ria drupal a la noticia	Tests: 1, Assert ions: 1, Failur es 0	Método de la clase Categori a

Anexo 9. Casos de prueba de aceptación, versión 0.2

Tabla 104: Caso de prueba de aceptación 8 HU 7

Caso de prueba de aceptación	
Código: PA8HU7	Historia de usuario: Gestionar contactos
Nombre: Insertar contacto	
Descripción: Se inserta un contacto en la lista de contactos del usuario.	
Condición de ejecución: El usuario debe estar autenticado	
Entrada/Pasos de ejecución: El usuario selecciona la opción Nuevo contacto. Introduce el usuario del contacto Da clic en la opción Enviar.	
Resultado esperado: Se inserta un contacto.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 105: Caso de prueba de aceptación 9 HU Gestionar contactos.

Caso de prueba de aceptación	
Código: PA9HU7	Historia de usuario: Gestionar contactos
Nombre: Eliminar contacto	
Descripción: Se elimina un contacto en la lista de contactos del usuario.	
Condición de ejecución: El usuario debe estar autenticado	
Entrada/Pasos de ejecución: El usuario selecciona la opción Eliminar.	
Resultado esperado: Se elimina un contacto.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 106: Caso de prueba de aceptación 10 HU Gestionar contactos.

Caso de prueba de aceptación	
Código: PA10HU7	Historia de usuario: Gestionar contactos
Nombre: Mostrar contacto	
Descripción: Se muestran los contactos de un usuario.	
Condición de ejecución: El usuario debe estar autenticado	

Entrada/Pasos de ejecución: El usuario selecciona la opción Mis contactos.
Resultado esperado: Se muestran los contactos del usuario.
Evaluación de la prueba: Prueba satisfactoria.

Tabla 107: Caso de prueba de aceptación 11 HU Gestionar marcador social.

Caso de prueba de aceptación	
Código: PA11HU8	Historia de usuario: Gestionar marcador social
Nombre: Insertar marcador social	
Descripción: Se inserta un marcador social	
Condición de ejecución: El usuario debe estar autenticado	
Entrada/Pasos de ejecución: El usuario da clic en la opción insertar marcador (Estrella)	
Resultado esperado: Se guarda el marcador en la lista de marcadores del usuario	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 108: Caso de prueba de aceptación 12 HU Gestionar marcador social.

Caso de prueba de aceptación	
Código: PA12HU8	Historia de usuario: Gestionar marcador social
Nombre: Compartir marcador social	
Descripción: Se comparte un marcador social con otros usuarios	
Condición de ejecución: El usuario debe estar autenticado	
Entrada/Pasos de ejecución: El usuario da clic en la opción Compartir. Se muestran los contactos del usuario. El usuario selecciona los contactos a los que va a compartir el marcador. Da clic en la opción Compartir.	
Resultado esperado: Se comparte el marcador con otros usuarios	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 109: Caso de prueba de aceptación 13 HU Gestionar marcador social.

Caso de prueba de aceptación	
Código: PA13HU8	Historia de usuario: Gestionar marcador social
Nombre: Eliminar marcador social	
Descripción: Se elimina un marcador social	
Condición de ejecución: El usuario debe estar autenticado	
Entrada/Pasos de ejecución: El usuario da clic en la opción Eliminar.	
Resultado esperado: Se elimina el marcador en la lista de marcadores del usuario	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 110: Caso de prueba de aceptación 14 HU Gestionar marcador social.

Caso de prueba de aceptación	
Código: PA14HU8	Historia de usuario: Gestionar marcador social
Nombre: Mostrar marcador social	
Descripción: Se muestran los marcadores de un usuario.	
Condición de ejecución: El usuario debe estar autenticado	
Entrada/Pasos de ejecución: El usuario da clic en la opción Mis marcadores.	
Resultado esperado: Se muestran los marcadores del usuario.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 111: Caso de prueba de aceptación 15 HU Administrar fuente.

Caso de prueba de aceptación	
Código: PA15HU9	Historia de usuario: Administrar fuente
Nombre: Activar fuente.	
Descripción: Se activa una fuente.	
Condición de ejecución: El usuario debe estar autenticado como administrador.	
Entrada/Pasos de ejecución: El usuario selecciona la opción Activar.	

Resultado esperado: Se activa la fuente
Evaluación de la prueba: Prueba satisfactoria.

Tabla 112: Caso de prueba de aceptación 16 HU Administrar fuente.

Caso de prueba de aceptación	
Código: PA16HU9	Historia de usuario: Administrar fuente
Nombre: Desactivar fuente.	
Descripción: Se desactiva una fuente.	
Condición de ejecución: El usuario debe estar autenticado como administrador.	
Entrada/Pasos de ejecución: El usuario selecciona la opción Desactivar.	
Resultado esperado: Se desactiva la fuente	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 113: Caso de prueba de aceptación 17 HU Administrar fuente.

Caso de prueba de aceptación	
Código: PA17HU9	Historia de usuario: Administrar fuente.
Nombre: Sugerir fuente	
Descripción: Se sugiere una fuente a tener en cuenta por el administrador para insertarla en el sitio.	
Condición de ejecución: El usuario debe estar autenticado	
Entrada/Pasos de ejecución: El usuario entra la fuente y url. Da clic en la opción Enviar.	
Resultado esperado: Se guarda en la base de datos la fuente sugerida por el usuario y se envía al administrador.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 114: Caso de prueba de aceptación 18 HU Administrar fuente.

Caso de prueba de aceptación	
Código: PA18HU9	Historia de usuario: Administrar fuente
Nombre: Insertar fuente.	

Descripción: Se inserta una fuente.
Condición de ejecución: El usuario debe estar autenticado como administrador.
Entrada/Pasos de ejecución: El usuario selecciona la opción Insertar.
Resultado esperado: Se inserta la fuente
Evaluación de la prueba: Prueba satisfactoria.

Tabla 115: Caso de prueba de aceptación 19 HU Insertar categoría.

Caso de prueba de aceptación	
Código: PA19HU10	Historia de usuario: Insertar categoría.
Nombre: Insertar categoría.	
Descripción: Se inserta una fuente.	
Condición de ejecución: Solo se puede insertar una categoría a una noticia si esta no tiene categoría.	
Entrada/Pasos de ejecución: El usuario selecciona la opción Insertar categoría. Introduce la categoría. Da clic en la opción Guardar. Se muestra la noticia con la categoría insertada	
Resultado esperado: Se inserta la categoría en la noticia.	
Evaluación de la prueba: Prueba satisfactoria.	

Anexo 10. Tareas de ingeniería, tercera iteración

Tabla 116. Tarea No.43: Realizar prueba unitaria: Test_ mostrar_votadas.

Tarea No.43		HU 11
Nombre de tarea: Realizar prueba unitaria: Test_ mostrar_votadas.		
Tipo de tarea: Implementación	Estimación: 1 días	
Fecha de inicio: 20 de abril del 2015	Fecha de fin: 20 de abril del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: implementar función Test_mostrar_votadas en la clase Test_tendencias.php para probar obtener las noticias más votadas.		

Tabla 117. Tarea No.44: Mostrar noticias más votadas.

Tarea No.44		HU 11
Nombre de tarea: Mostrar noticias más votadas.		
Tipo de tarea: Implementación	Estimación: 1 días	
Fecha de inicio: 20 de abril del 2015	Fecha de fin: 20 de abril del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: adicionar procedimientos necesarios a la función noticiasMasVotadas en la clase Tendencias.php para mostrar las noticias más votadas por los usuarios del sitio web. Además se crea Tendencias_view. Php para mostrar la información.		

Tabla 118. Tarea No.45: Realizar prueba unitaria: Test_ mostrar_comentadas.

Tarea No.45		HU 11
Nombre de tarea: Realizar prueba unitaria: Test_ mostrar_comentadas.		
Tipo de tarea: Implementación	Estimación: 1 días	
Fecha de inicio: 21 de abril del 2015	Fecha de fin: 21 de abril del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: implementar función Test_ mostrar_comentadas en la clase Test_tendencias.php para probar obtener las noticias más comentadas del sitio web.		

Tabla 119. Tarea No.46: Mostrar noticias más comentadas.

Tarea No.46		HU 11
Nombre de tarea: Mostrar noticias más votadas.		

Tipo de tarea: Implementación	Estimación: 1 días
Fecha de inicio: 21 de abril del 2015	Fecha de fin: 21 de abril del 2015
Programador responsable: Yasiel Rodríguez Sánchez	
Descripción: adicionar procedimientos necesarios a la función noticiasMasComentadas en la clase Tendencias.php para mostrar las noticias más votadas del sitio web.	

Tabla 120. Tarea No.47: Realizar prueba unitaria: Test_ mostrar_usuarios.

Tarea No.47		HU 11
Nombre de tarea: Realizar prueba unitaria: Test_ mostrar_usuarios.		
Tipo de tarea: Implementación	Estimación: 1 días	
Fecha de inicio: 22 de abril del 2015	Fecha de fin: 22 de abril del 2015	
Programador responsable: Ailyn Pérez Aguila		
Descripción: implementar función Test_ mostrar_usuarios en la clase Test_Tendencias.php para mostrar los usuarios que más comentan.		

Tabla 121. Tarea No.48: Mostrar usuarios que más comentan.

Tarea No.48		HU 11
Nombre de tarea: Mostrar usuarios que más comentan.		
Tipo de tarea: Implementación	Estimación: 1 días	
Fecha de inicio: 22 de abril del 2015	Fecha de fin: 22 de abril del 2015	
Programador responsable: Ailyn Pérez Aguila		
Descripción: Adicionar procedimientos necesarios a la función ordenarUserXComentarios en la clase Tendencias.php para mostrar los usuarios que más comentan.		

Tabla 122. Tarea No.49: Realizar prueba unitaria: Test_ mostrar_sitios.

Tarea No.49		HU 11
Nombre de tarea: Realizar prueba unitaria: Test_ mostrar_sitios.		
Tipo de tarea: Implementación	Estimación: 1 días	
Fecha de inicio: 23 de abril del 2015	Fecha de fin: 23 de abril del 2015	
Programador responsable: Ailyn Pérez Aguila		

Descripción: implementar función Test_ mostrar_sitios en la clase Test_Tendencias.php para mostrar los sitios que más noticias publican.

Tabla 123. Tarea No.50: Mostrar sitios que más noticias publican.

Tarea No.50		HU 11
Nombre de tarea: Mostrar sitios que más noticias publican.		
Tipo de tarea: Implementación	Estimación: 2 días	
Fecha de inicio: 23 de abril del 2015	Fecha de fin: 24 de abril del 2015	
Programador responsable: Ailyn Pérez Aguila		
Descripción: Adicionar procedimientos necesarios a la función ordenarXCantNoticias en la clase Tendencias.php para mostrar los sitios que más noticias publican.		

Tabla 124. Tarea No.51: Realizar prueba unitaria: Test_ buscar.

Tarea No.51		HU 12
Nombre de tarea: Realizar prueba unitaria: Test_ buscar.		
Tipo de tarea: Implementación	Estimación: 2 días	
Fecha de inicio: 27 de abril del 2015	Fecha de fin: 28 de abril del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: implementar función Test_ buscar en la clase Test_Busqueda.php para probar obtener las noticias que cumplan con el criterio de búsqueda introducido por el usuario.		

Tabla 125. Tarea No.52: Buscar noticias.

Tarea No.52		HU 12
Nombre de tarea: Buscar noticias.		
Tipo de tarea: Implementación	Estimación: 3 días	
Fecha de inicio: 29 de abril del 2015	Fecha de fin: 1 de mayo del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: adicionar procedimientos necesarios a la función buscar_noticias en la clase Busqueda.php para mostrar las noticias que cumplan con el criterio de		

búsqueda introducido por el usuario. Además se crea la vista busqueda_view para mostrar la información.

Tabla 126. Tarea No.53: Realizar prueba unitaria: Test_ mostrar_cantN.

Tarea No.53		HU 13
Nombre de tarea: Realizar prueba unitaria: Test_ mostrar_cantN.		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 4 de mayo del 2015	Fecha de fin: 4 de mayo del 2015	
Programador responsable: Ailyn Pérez Aguila		
Descripción: implementar función Test_ mostrar_cantN en la clase Test_Estadisticas_dao.php para probar obtener la cantidad de noticias que existen en el sitio web.		

Tabla 127. Tarea No.54: Mostrar cantidad de noticias.

Tarea No.54		HU 13
Nombre de tarea: Mostrar cantidad de noticias.		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 5 de mayo del 2015	Fecha de fin: 5 de mayo del 2015	
Programador responsable: Ailyn Pérez Aguila		
Descripción: adicionar procedimientos necesarios a la función Cantidadnoticias en la clase estadisticas_dao.php para mostrar la cantidad de noticias que existen en el sitio web. Además se crea en el Estadisticas_view.php para mostrar la información.		

Tabla 128. Tarea No.55: Realizar prueba unitaria: Test_ mostrar_cantF.

Tarea No.55		HU 13
Nombre de tarea: Realizar prueba unitaria: Test_ mostrar_cantF.		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 6 de mayo del 2015	Fecha de fin: 6 de mayo del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: implementar función Test_ mostrar_cantF en la clase Test_Estadisticas_dao.php para probar obtener la cantidad de noticias que existen en el sitio web.		

Tabla 129. Tarea No.56: Mostrar cantidad de fuentes activas.

Tarea No.56		HU 13
Nombre de tarea: Mostrar cantidad de fuentes activas.		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 7 de mayo del 2015	Fecha de fin: 7 de mayo del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: Adicionar procedimientos necesarios a la función Cantidadfuentes en la clase estadisticas_dao.php para mostrar la cantidad de noticias que existen en el sitio web.		

Tabla 130. Tarea No.57: Realizar prueba unitaria: Test_ mostrar_cantC.

Tarea No.57		HU 13
Nombre de tarea: Realizar prueba unitaria: Test_ mostrar_cantC.		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 8 de mayo del 2015	Fecha de fin: 8 de mayo del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: implementar función Test_ mostrar_cantC en la clase Test_Estadisticas_dao.php para probar obtener la cantidad de categorías que existen en el sitio web.		

Tabla 131. Tarea No.58: Mostrar cantidad de categorías de noticias.

Tarea No.58		HU 13
Nombre de tarea: Mostrar cantidad de categorías de noticias.		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 8 de mayo del 2015	Fecha de fin: 8 de mayo del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: adicionar procedimientos necesarios a la función Cantidadcategorias en la clase estadisticas_dao.php para mostrar la cantidad de categorías que existen en el sitio web.		

Tabla 132. Tarea No.59: Realizar prueba unitaria: Test_ marcar_fuente.

Tarea No.59		HU 14
-------------	--	-------

Nombre de tarea: Realizar prueba unitaria: Test_ marcar_fuente.	
Tipo de tarea: Implementación	Estimación: 1 día
Fecha de inicio: 11 de mayo del 2015	Fecha de fin: 11 de mayo del 2015
Programador responsable: Yasiel Rodríguez Sánchez	
Descripción: implementar función Test_ marcar_fuente en la clase Test_autenticado.php para probar marcar una fuente de RSS.	

Tabla 133. Tarea No.60: Marcar fuentes.

Tarea No.60		HU 14
Nombre de tarea: Marcar fuentes.		
Tipo de tarea: Implementación	Estimación: 2 día	
Fecha de inicio: 11 de mayo del 2015	Fecha de fin: 12 de mayo del 2015	
Programador responsable: Yasiel Rodríguez Sánchez		
Descripción: Adicionar procedimientos necesarios en la clase Test_autenticado.php para marcar una fuente de RSS. Además se crea en el Test_view.php con el uso del framework Bootstrap el botón Sí.		

Tabla 134. Tarea No.61: Realizar prueba unitaria: Test_ desmarcar_fuente.

Tarea No.61		HU 14
Nombre de tarea: Realizar prueba unitaria: Test_ desmarcar_fuente.		
Tipo de tarea: Implementación	Estimación: 1 día	
Fecha de inicio: 13 de mayo del 2015	Fecha de fin: 13 de mayo del 2015	
Programador responsable: Ailyn Pérez Aguila		
Descripción: implementar función Test_ desmarcar_fuente en la clase Test_autenticado.php para probar desmarcar una fuente de RSS.		

Tabla 135. Tarea No.62: Desmarcar fuentes.

Tarea No.62		HU 14
Nombre de tarea: Desmarcar fuentes.		
Tipo de tarea: Implementación	Estimación: 2 día	
Fecha de inicio: 14 de mayo del 2015	Fecha de fin: 15 de mayo del 2015	
Programador responsable: Ailyn Pérez Aguila		
Descripción: Adicionar procedimientos necesarios en la clase Test_autenticado.php para desmarcar una fuente de RSS. Además se crea en el Test_view.php con el uso del framework Bootstrap el botón No.		

Anexo 11. Casos de prueba unitarias de la tercera iteración

Tabla 136: Caso de prueba unitaria No.19.

Caso de prueba unitaria No.19					
Iteración: 3					
Descripción: Se prueba mostrar noticias más votadas.					
Responsable: Ailyn Pérez Aguila					
Funcionali dad	Método Utilizado	Recibe	Respu esta esper ada del métod o	Respu esta real del métod o	Observa ciones
mostrar_vo tadas()	Test_ mostrar_vo tadas()	new Noticia_model(1, titulo1, NULL, 1, 1, 1, 0, 2015-05-05 20:31:00, NULL, 0, Drupaleros, NULL, "http://drupaleros.uci.cu/? q=node/560"),5	Se muestr an los noticia s con cantid ad de votos 5	Tests: 1, Asserti ons: 1, Failure s 0	Método de la clase Tendenci as

Tabla 137: Caso de prueba unitaria No.20.

Caso de prueba unitaria No.20					
Iteración: 3					
Descripción: Se prueba mostrar noticias más comentadas.					
Responsable: Ailyn Pérez Aguila					
Funcionalid ad	Método Utilizado	Recibe	Respu esta esper ada del	Resp uesta real del	Observa ciones

			método	método	
mostrar_comentadas()	Test_mostrar_comentadas()	new Noticia_model(1, titulo1, NULL, 1, 1, 1, 0, 2015-05-05 20:31:00, NULL, 0, Drupaleros, NULL, "http://drupaleros.uci.cu/?q=node/560"),5	Se muestran las noticias con cantidad de comentarios 5	Tests: 1, Assertions: 1, Failures 0	Método de la clase Tendencias

Tabla 138: Caso de prueba unitaria No.21.

Caso de prueba unitaria No.21					
Iteración: 3					
Descripción: Se prueba mostrar los usuarios que más comentan					
Responsable: Ailyn Pérez Aguila					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
mostrar_usuarios()	Test_mostrar_usuarios()	Aguila ,5	Se muestran los usuarios que con cantidad de comentarios 5	Tests: 1, Assertion: 1, Failures 0	Método de la clase Tendencias

Tabla 139: Caso de prueba unitaria No.22.

Caso de prueba unitaria No.22					
Iteración: 3					
Descripción: Se prueba mostrar los sitios web que más noticias publican					
Responsable: Ailyn Pérez Aguila					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
mostrar_sitios()	Test_mostrar_sitios()	Drupaleros ,5	Se muestran los el sitio con cantidad de noticias publicadas, 5	Tests: 1, Assertion: 1, Failures 0	Método de la clase Tendencias

Tabla 140: Caso de prueba unitaria No.23.

Caso de prueba unitaria No.23					
Iteración: 3					
Descripción: Se prueba buscar noticias					
Responsable: Ailyn Pérez Aguila					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
buscar()	Test_buscar()	Cuba	Se muestran todas las noticias que	Tests: 1, Assertions: 1, Failures 0	Método de la clase Buscador

			contengan la palabra cuba/Cuba en su contenido		
--	--	--	--	--	--

Tabla 141: Caso de prueba unitaria No.24.

Caso de prueba unitaria No.24					
Iteración: 3					
Descripción: Se prueba mostrar cantidad de noticias del sitio					
Responsable: Ailyn Pérez Aguila					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
mostrar_cant N()	Test_mostrar_cant N()	Drupaleros ,5	Se muestra la cantidad de noticas que contiene el sitio hasta el momento	Tests: 1, Assertion: 1, Failures 0	Método de la clase Estadísticas

Tabla 142: Caso de prueba unitaria No.25.

Caso de prueba unitaria No.25					
Iteración: 3					
Descripción: Se prueba mostrar la cantidad de categorías					
Responsable: Ailyn Pérez Aguila					

Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
mostrar_cantC()	Test_mostrar_cantC()	Drupaleros, 10	Se muestra la cantidad de categorías	Tests: 1, Assertions: 1, Failures 0	Método de la clase Estadísticas

Tabla 143: Caso de prueba unitaria No.26.

Caso de prueba unitaria No.26					
Iteración: 3					
Descripción: Se prueba mostrar la cantidad de fuentes					
Responsable: Ailyn Pérez Aguila					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
mostrar_cantF()	Test_mostrar_cantF()	12	Se muestra la cantidad de fuentes	Tests: 1, Assertions: 1, Failures 0	Método de la clase Estadísticas

Tabla 144: Caso de prueba unitaria No.27.

Caso de prueba unitaria No.27					
Iteración: 3					
Descripción: Se prueba marcar una fuente					
Responsable: Ailyn Pérez Aguila					

Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
marcar_fuente()	Test_marcar_fuente()	new Fuente_model (1, Drupaleros, "http://drupaleros.uci.cu/rss.xml", 0)	Se marca la fuente	Tests: 1, Assertions: 1, Failures 0	Método de la clase Perfil

Tabla 145: Caso de prueba unitaria No.28.

Caso de prueba unitaria No.28					
Iteración: 3					
Descripción: Se prueba desmarcar una fuente					
Responsable: Ailyn Pérez Aguila					
Funcionalidad	Método Utilizado	Recibe	Respuesta esperada del método	Respuesta real del método	Observaciones
desmarcar_fuente()	Test_marcar_fuente()	new Fuente_model (1, Drupaleros, "http://drupaleros.uci.cu/rss.xml", 0)	Se desmarca la fuente	Tests: 1, Assertions: 1, Failures 0	Método de la clase Perfil

Anexo 12. Casos de prueba de aceptación, versión 0.3

Tabla 146: Caso de prueba de aceptación 20 HU Mostrar tendencias.

Caso de prueba de aceptación	
Código: PA20HU11	Historia de usuario: Mostrar tendencias
Nombre: Mostrar noticias más votadas.	
Descripción: Se muestran las noticias más votadas	
Condición de ejecución:	
Entrada/Pasos de ejecución: El usuario selecciona la opción Tendencias Se muestran las noticias más votadas	
Resultado esperado: Se muestran las noticias más votadas.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 147: Caso de prueba de aceptación 21 HU Mostrar tendencias.

Caso de prueba de aceptación	
Código: PA21HU11	Historia de usuario: Mostrar tendencias
Nombre: Mostrar noticias más comentadas.	
Descripción: Se muestran las noticias más comentadas	
Condición de ejecución:	
Entrada/Pasos de ejecución: El usuario selecciona la opción Tendencias Se muestran las noticias más comentadas	
Resultado esperado: Se muestran las noticias más comentadas.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 148: Caso de prueba de aceptación 22 HU Mostrar tendencias.

Caso de prueba de aceptación	
Código: PA22HU11	Historia de usuario: Mostrar tendencias
Nombre: Mostrar usuarios que más comentan.	
Descripción: Se muestran los usuarios que más comentan	
Condición de ejecución:	

<p>Entrada/Pasos de ejecución: El usuario selecciona la opción Tendencias Se muestran los usuarios que más comentarios han hecho.</p>
<p>Resultado esperado: Se muestran los usuarios que más comentan.</p>
<p>Evaluación de la prueba: Prueba satisfactoria.</p>

Tabla 149: Caso de prueba de aceptación 23 HU Mostrar tendencias.

Caso de prueba de aceptación	
Código: PA23HU11	Historia de usuario: Mostrar tendencias
Nombre: Mostrar sitios que más noticias publican	
Descripción: Se muestran los sitios que más noticias publican	
Condición de ejecución:	
<p>Entrada/Pasos de ejecución: El usuario selecciona la opción Tendencias Se muestran los sitios que más noticias han publicado.</p>	
Resultado esperado: Se muestran los sitios que más noticias publican	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 150: Caso de prueba de aceptación 24 HU: Buscar noticias.

Caso de prueba de aceptación	
Código: PA24HU12	Historia de usuario: Buscar noticias.
Descripción: Se buscan noticias filtrando por cualquier palabra, todas las palabras, frase exacta.	
Condición de ejecución:	
<p>Entrada/Pasos de ejecución: El usuario selecciona la opción Buscar Entra las palabras. Selecciona la opción Buscar</p>	
Resultado esperado: Se muestran las noticias que cumplen con los criterios de búsqueda introducidos.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 151: Caso de prueba de aceptación 25 HU Mostrar estadísticas.

Caso de prueba de aceptación	
Código: PA25HU13	Historia de usuario: Mostrar estadísticas
Nombre: Mostrar cantidad de noticias.	
Descripción: Se muestra la cantidad de noticias existentes en el sitio.	
Condición de ejecución:	
Entrada/Pasos de ejecución:	
Resultado esperado: Se muestra la cantidad de noticias existentes en el sitio.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 152: Caso de prueba de aceptación 26 HU Mostrar estadísticas

Caso de prueba de aceptación	
Código: PA26HU13	Historia de usuario: Mostrar estadísticas
Nombre: Mostrar cantidad de fuentes activas.	
Descripción: Se muestra la cantidad de fuentes activas en el sitio.	
Condición de ejecución:	
Entrada/Pasos de ejecución:	
Resultado esperado: Se muestra la cantidad de noticias activas en el sitio.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 153: Caso de prueba de aceptación 27 HU Mostrar estadísticas

Caso de prueba de aceptación	
Código: PA27HU13	Historia de usuario: Mostrar estadísticas
Nombre: Mostrar cantidad de categorías de noticias.	
Descripción: Se muestra la cantidad de categorías de noticias existentes en el sitio.	
Condición de ejecución:	
Entrada/Pasos de ejecución:	
Resultado esperado: Se muestra la cantidad de categorías de noticias existentes en el sitio.	

Evaluación de la prueba: Prueba satisfactoria.

Tabla 154: Caso de prueba de aceptación 28 HU Configurar preferencias.

Caso de prueba de aceptación	
Código: PA28HU14	Historia de usuario: Configurar preferencias
Nombre: Marcar fuentes	
Descripción: Se marcan las fuentes de las que el usuario desea que se muestren noticias.	
Condición de ejecución: El usuario tiene que estar autenticado	
Entrada/Pasos de ejecución: El usuario marca la opción Preferencias El usuario marca el botón Sí.	
Resultado esperado: Se muestran las noticias de las fuentes marcadas.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 155: Caso de prueba de aceptación 29 HU Configurar preferencias.

Caso de prueba de aceptación	
Código: PA29HU14	Historia de usuario: Configurar preferencias
Nombre: Desmarcar fuentes	
Descripción: Se desmarcan las fuentes de las que el usuario no desea que se muestren noticias.	
Condición de ejecución: El usuario tiene que estar autenticado	
Entrada/Pasos de ejecución: El usuario marca la opción Preferencias El usuario marca el botón No.	
Resultado esperado: No se muestran las noticias de las fuentes desmarcadas.	
Evaluación de la prueba: Prueba satisfactoria.	