Universidad Autónoma Metropolitana
Unidad Azcapotzalco
División de Ciencias Básicas e Ingeniería

Reporte Final del Proyecto de Integración

Licenciatura en Ingeniería en Computación

Proyecto tecnológico

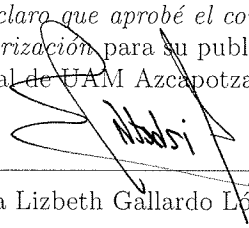
Sistema para promover la lectura de historieta digital de autores independientes

Jesús Sánchez Quiñones
209302320

Asesora:
Dra. María Lizbeth Gallardo López
No. Económico: 30761
Profesor Investigador Asociado "D"

Trimestre 2017 primavera
Fecha de entrega

Yo, María Lizbeth Gallardo López, declaro que aprobé el contenido del presente Reporte de Proyecto de Integración y doy mi autorización para su publicación en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.



Dra. María Lizbeth Gallardo López

Yo, Jesús Sánchez Quiñones, doy mi autorización a la Coordinación de Servicios de Información de la Universidad Autónoma Metropolitana, Unidad Azcapotzalco, para publicar el presente documento en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Jesús Sánchez Quiñones

Resumen

Con el uso creciente de las redes sociales y otros canales de comunicación digital, la historieta desarrollada por autores independientes ha encontrado un espacio en el que se puede distribuir y masificar. Sin embargo, en tales plataformas los autores no reciben una remuneración directa por compartir públicamente su trabajo y se enfrentan además a problemas como el robo intelectual de sus obras, pues resulta muy fácil compartir un trabajo sin dar los respectivos créditos al autor original.

Problemas como estos se deben principalmente a que dichos sitios no están diseñados específicamente para la lectura de historieta en línea. Tampoco se hace mucho énfasis en el valor que tienen los derechos de autor, por lo que se permiten muchas actividades que resultan perjudiciales para el desarrollo de historieta independiente y sus autores.

El propósito de este proyecto fue proporcionar una plataforma web adecuada para apoyar la realización y distribución de historietas independientes hechas en México, así como fomentar la lectura.

Esta plataforma web busca recompensar al lector que reconoce pertinentemente el trabajo de los autores. Para lograrlo, se plantean dos mecanismos: micro-donaciones a los autores y evaluación de las historietas por parte de los lectores.

Tabla de Contenido

Índice de figuras

1. Introducción

Muchos de los autores de historieta independiente en México se enfrentan a una serie de obstáculos que en la mayoría de los casos truncan el desarrollo de sus proyectos. En buena medida se debe a la falta de apoyo económico y la deficiente difusión en los espacios dedicados a promover a nuevos autores.

El objetivo del presente proyecto fue el de desarrollar una plataforma web, en donde autores de historieta independiente puedan publicar sus obras y darlas a conocer en un espacio en el que se promueva la lectura, la realimentación de sus obras y el apoyo capital a través de donativos por parte del público lector. Dicho objetivo se pretende alcanzar mediante el diseño y la implantación de los siguientes módulos: gestión de depósitos, gestión de donaciones, gestión de recompensas y gestión de evaluaciones.

Entre los trabajos relacionados que sirvieron de apoyo para el desarrollo de este proyecto se destacan los siguientes: "Especificación de sistemas electrónicos de microdonaciones", el cual se toma como referencia para aplicar el concepto de microdonaciones. Y "Desarrollo y mejora de las estrategias de autoevaluación formativa de la asignatura de Fundamentos de Diseño Gráfico", donde se plantean estrategias de autoevaluación, en las que participan alumnos para valorar los trabajos y artículos académicos que los profesores disponen en una plataforma institucional, bastante parecido a lo que se persigue alcanzar con este proyecto en el rubro de realimentación al autor.

Para el desarrollo de la aplicación se empleó la metodología de Proceso Unificado (PU). La aplicación se basa en el modelo MVC (Modelo Vista Controlador). Las herramientas utilizadas en el proceso de desarrollo fueron: Netbeans 8.2, MySQL workbench 6.2, Servidor de aplicaciones Apache Tomcat 8, Struts2 y Hibernate.

Como queda dicho, es importante que se impulse el desarrollo de este sector (el de la historieta) en nuestro país, que en comparación con otras naciones se queda bastante rezagado. En internet se pueden encontrar varios espacios en los cuales los autores pueden difundir sus obras, lamentablemente muchos de estos no son adecuados para la labor de la historieta. Aquellos espacios que se dedican específicamente a la publicación y lectura de historieta están, por desgracia, mas bien dirigidos al público angloparlante.

2. Antecedentes

2.1. Artículos de investigación.

En el "artículo" "Especificación de sistemas electrónicos de microdonaciones" [1] se presentan los conceptos de microdonaciones y macrodonaciones, además de mencionar las características de cada uno de ellos. La similitud que tiene con el sistema de este proyecto se tomará como referencia para aplicar las microdonaciones, en este caso será una aplicación web de historietas. La diferencia es que en el artículo manejan microdonaciones menores a un euro (1 Euro = 19.89 Pesos Mexicanos, según el cambio actual) mientras que en el sistema se contemplan donaciones poco mayores a un Euro.

En el artículo "Desarrollo y mejora de las estrategias de autoevaluación formativa de la asignatura de Fundamentos de Diseño Gráfico" [2] plantean una estrategia de autoevaluaciones formativas en la cual participan los alumnos para evaluar los trabajos y artículos académicos que los profesores disponen en la plataforma institucional. La similitud que tiene con este proyecto es que ambos requieren la realimentación con la participación de los usuarios, que en este caso se trata del lector de historieta, y así, dar a conocer a los potenciales lectores, qué tan interesante o recomendable es una obra determinada. La diferencia con el proyecto planteado en este artículo es el método de evaluación, pues mientras ellos implantan una calificación de estrellas con un rango de 0 a 10, en este proyecto se maneja una escala de hasta 5 estrellas, además de una caja de comentarios para complementar la evaluación.

En el artículo "Técnicas de gamificación aplicadas en la docencia de Ingeniería informática" [3] se aborda el tema de gamificación en el área educativa, en ella se abordan los conceptos de gamificación, las técnicas que existen y sobre todo las mecánicas que provocan el desrrollo del juego. De aquí se rescata el propósito de impulsar al usuario a interactuar con el entorno en que se encuentra, y en este caso particular orientado a las microdonaciones.

2.2. Tesis.

En la tesis "Carry Class: una plataforma de gamificación" [4] se habla de una aplicación web de gamificación enfocada en el área educativa para que los alumnos y profesores puedan aplicar este concepto en las aulas. La principal similitud con este proyecto es la de incentivar a los lectores por sus donativos a los autores. Las principales diferencias son: a) Ellos emplean PHP para el manejo de base de datos, mientras que en este proyecto se emplea hibernate; b) Ellos emplean lenguajes como AJAX, HTML y JavaScript, mientras que en este proyecto se trabaja con Java, JSP y HTML5; c) Ellos emplean un plugin gratuito llamado WP-PostRatings, este proyecto se implanta la lógica para la funcionalidad correspondiente.

2.3. Proyectos terminales.

En el proyecto "Sistemas de inventario y ventas para tlapalería con pronóstico de mercado" [5] se aborda la problemática que tiene una tlapalería para gestionar sus ventas y mercancía. La similitud con este proyecto es un módulo que dedican a las ventas, en nuestro caso es un módulo de donaciones, la diferencia con esta propuesta es el manejo de capital, mientras que ellos lo hacen en efectivo, en este proyecto se simula el manejo de sistema de pagos electrónicos vía PayPal.

En el proyecto "Aplicación móvil para la recomendación de productos para venta en comercio electrónico" [6] se implementa un algoritmo heurístico para la recomendación de compra de productos. La similitud con este proyecto es que se hacen recomendaciones basadas en la ponderación de determinadas características de un producto según las necesidades y/o gustos del consumidos, mientras que en este proyecto las recomendaciones se hacen en base a la calificación que el lector asigna a una obra, recomendándole al lector obras afines al género o autor de su preferencia.

2.4. Software.

El sitio web "neobux.com" [7] es una plataforma en la que a los usuarios se les paga por ver anuncios de internet. Cada anuncio le retribuye una fracción de dólar al usuario y puede ver una cantidad limitada de anuncios por día. Para que el usuario pueda disponer de sus ganancias necesita contar con una cuenta PayPal y registrarla junto con sus datos en este sitio, así, cuando se alcanza una cantidad mínima de dos dólares, el usuario puede hacer una transferencia a su cuenta PayPal. Dicho movimiento tiene un cargo por concepto de comisión. Al igual que en este sitio, se contempla integrar los servicios de PayPal para el manejo de transacciones siguiendo una lógica similar para otorgar los pagos a los autores de historieta.

3. Justificación

Muchos proyectos de historietas en línea se abandonan prematuramente y quedan inconclusos por falta de apoyo capital a los autores. Cuando un autor busca solventar los gastos de producción para continuar con la realización de sus historietas, recurre a la organización de eventos donde junto a otros autores independientes pueden ofrecer diversos tipos de productos, que sobre todo tienen que ver con historietas o ilustración. Lamentablemente el objetivo de recaudar fondos no siempre se logra debido a una variedad de problemas, de los cuales los más destacables son: i) la mala ubicación de los eventos; cafeterías, bares, casas o departamentos desocupados, son algunas de las locaciones más usuales para establecer dichos eventos, por lo tanto el alvence de potenciales consumidores se limitado. ii) se desvirtúa el propósito de los eventos; en muchas ocasiones con el objeto de llamar la atención del público, en estos lugares se ofrecen bebidas alcohólicas, lo cual no propicia un ambiente apto para todo público, y al final la intención original, la de apoyar a los autores, pasa a segundo plano. iii) los mejores eventos son poco accesibles; algunos eventos mucho mejor organizados o que incluso cuentan con el apoyo de organizaciones bien establecidas en el medio, cobran precios elevados por el espacio de venta, esto es una apuesta arriesgada para muchos autores independientes, pues si su trabajo no es tan conocido, pueden quedar fácilmente opacados por aquellos autores que tienen un público mas estable y en consecuencia es poco probable que recuperen la inversión.

Con la implantación de este sistema se busca que los autores cuenten con una plataforma web que les ofrezca la posibilidad de recibir apoyo económico en base a donaciones por parte de los lectores. Esto a su vez tiene el propósito de estimular una competencia más equilibrada entre los autores de historieta independiete y promover la cultura de la retribución en el público lector. Por otra parte, es de suma importancia la realimentación a los autores, por lo que además en este proyecto se integra un módulo de evaluación donde los lectores emiten una calificación y un comentario a las obras. Esto permite al autor mejorar la calidad de su trabajo y conocer mejor a su público.

4. Objetivos

Objetivo general: Construir una aplicación web que integre los mecanismos de donaciones y evaluación, para promover la lectura de historieta digital de autores independientes.

Objetivos particulares:

- Diseñar e implantar un módulo que se comunique con el servicio de pagos en línea "PayPal" para gestionar los depósitos de los lectores.
- Diseñar e implantar un módulo que gestione las donaciones a los autores de historieta.
- Diseñar e implantar un módulo que gestione recompensas a los lectores por sus donativos. Las recompensas consisten en historietas exclusivas desbloqueables.
- Diseñar e implantar un módulo de evaluación para las obras. Para valorar las obras se usará una variación de la escala Likert, con el que se medirá qué tan recomendable resulta una obra según el criterio de cada lector. A saber, los cinco niveles que comprenderá esta escala son: nada recomendable, poco recomendable, recomensable, muy recomendable y extremadamente recomendable. Cada uno de estos niveles será identificado por una cantidad de estrellas (que va de una hasta cinco).
- Integrar los módulos anteriores en una aplicación web.

La aplicación web desarrollada en este proyecto, en adelante será referida como Artico.

5. Marco teórico

Con el propósito de entender mejor el desarrollo del presente proyecto, en esta sección se definen conceptos relacionados con los sistemas de información y los menesteres propios de la historieta digital. Además se explican los mecanismos involucrados con la lógica del negocio y las herramientas empleadas para construir la aplicación.

5.1. Aplicación Web

"Las aplicaciones web se encuadran dentro de las arquitecturas cliente/servidor: un ordenador solicita servicios (cliente) y otro está a la espera de recibir solicitudes y las responde (servidor)."[8]

Las aplicaciones web nos permiten generar automáticamente el contenido de una página y personalizarla según el perfil de un usuario. Además, através de una aplicación web se puede interactuar con los sistemas de información que gestionan la lógica de un negocio específico.

5.2. Sistema de información

Se conoce como sistema de información al conjunto de personas, datos, procesos y tecnologías, que trabajan de manera conjunta para procesar información de entrada y propor-

cionar como salida una respuesta o resultado que ayude en la toma de decisiones de una organización. También permiten recopilar y guardar los datos para un fin específico.

Los sistemas de información se pueden clasificar de acuerdo a las funciones que realizan: "Los **sistemas de procesamiento de transacciones (transaction processing systems, TPS)** procesan transacciones de negocios como pedidos, tarjetas de tiempo, pagos y reservaciones. Los **sistemas de información administrativa (management information systems, MIS)** utilizan los datos de transacción para producir información necesaria por los administradores para dirigir el negocio. Los **sistemas de soporte de decisiones (decision support systems, DSS)** ayudan a diversos tomadores de decisiones a identificar y elegir entre opciones o decisiones. Los **sistemas de información ejecutiva (executive information systems, EIS)** están adaptados a las necesidades de información únicas de los ejecutivos que planean el negocio y evalúan el desempeño contra esos planes. Los **sistemas expertos** capturan y reproducen el conocimiento de un solucionador de problemas experto o un tomador de decisiones y luego simulan el "pensamiento" de ese experto. Los **sistemas de comunicación y colaboración** resaltan la comunicación y la colaboración entre las personas, tanto internas como externas de la organización. Finalmente, los **sistemas de automatización de oficina** ayudan a empleados a crear y compartir documentos que respaldan las actividades diarias de oficina."[9]

De acuerdo con las descripciones arriba mencionadas, el presente proyecto entra en dos categorías. La primera es la de un **sistema de procesamiento de transacciones**, pues dentro del sistema se propone un módulo que se encarga de gestionar depósitos y compra de puntos vía Paypal. La otra categoría es un **sistema de comunicación y colaboración**, ya que en el sistema existe un módulo que se encarga de recompensar a los usuarios lectores que realizan donativos a los autores de historieta. Además, existe un módulo en el cual los lectores pueden realimentar a los autores con una evaluación que consiste en dar una calificación y un comentario acerca de una obra específica. Estos módulos se explican con mas detalle en secciones posteriores.

5.3. Framework MVC

Una framework, o marco de trabajo, es una pieza de software que provee la automatización de tareas comunes, así como una arquitectura para el flujo de trabajo del dominio en cuestión. Todo con el fin de reducir el tiempo y facilitar la construcción de aplicaciones.

La framework utilizada para el desarrollo de este proyecto es struts 2, que contrario a lo que se podría pensar, no se trata de un lanzamiento nuevo de la framework Struts 1, sino más bien de una tecnología nueva que se basa en la framework OpenSymphony WebWork. Se trata de una framework de segunda generación, para aplicaciones web, que implementa el patrón de diseño MVC (Modelo Vista Controlador), en ocasiones referido como *Model 2*.[10]

5.4. Framework ORM

Una framework ORM (Object Relational Mapping) es la persistencia, automatizada y transparente, de objetos en una aplicación java en tablas de una base de datos relacional, usando metadatos que describen la relación entre los objetos y la base de datos.[11]

Así pues, hibernate, que es la herramienta utilizada en la construcción de Artico, es una framework ORM que permite la persistencia de los datos.

6. Desarrollo del proyecto

Esta sección se compone de las subsecciones: *Metodología empleada*, *Diseño del sistema*, Uso del sistema y Hardware y software necesario. En la subsección *Diseño del sistema* están los artefactos de diseño que ayudaron al desarrollo del sistema. Para ejemplificar el desarrollo se presentan dos casos de uso, considerados los mas significativos del sistema.

6.1. Metodología empleada

El desarrollo de este proyecto se rige por los principios del proceso unificado de software (UP por sus siglas en inglés). El proceso unificado de software consta de tres aspectos fundamentales: dirigido por casos de uso, centrado en la arquitectura y es iterativo e incremental [12]. Una herramienta clave es el Lenguaje Unificado de Modelado (Unified Model Language, UML), que sirve para preparar todos los esquemas del sistema software.

6.2. Diseño estático del sistema

Los artefactos construidos y empleados en el diseño estático del sistema son: Diagrama de casos de uso, Casos de uso de texto y Diagrama de clases.

En la figura 1 se puede apreciar el diagrama de casos de uso que definen a Artico. Se tienen tres actores primarios: Administrador, Lector y Autor, además de un actor secundario que en este caso se trata del proveedor de servicios PayPal para las transacciones de compra de puntos o retiro de donativos.

6.2.1. Diagrama de casos de uso

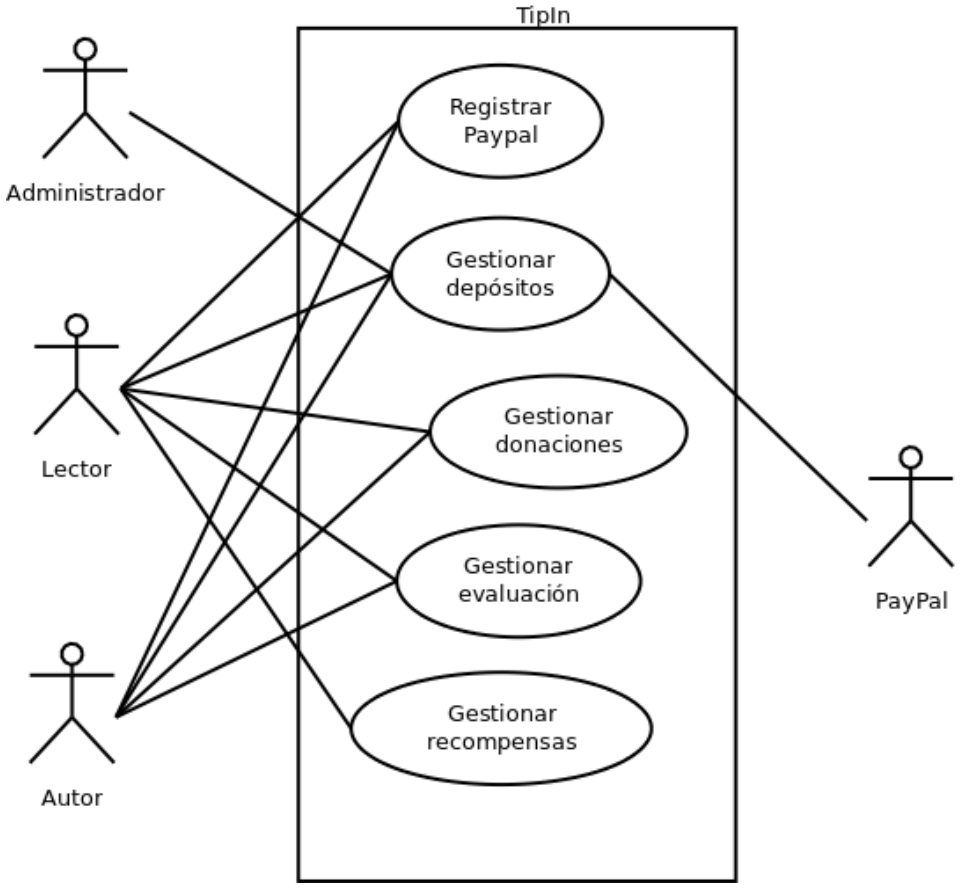


Figura 1: Diagrama de casos de uso general.

6.2.2. Casos de uso de texto

A continuación se presentan dos casos de uso: *Gestionar Donaciones* y *Gestionar Recompensas*.

Gestionar Donaciones

Resumen	El Lector gasta puntos en el Autor de su elección para obtener gemas a cambio.
Actores	Lector, Autor
Precondiciones	Contar con una cantidad mínima de puntos
Descripción	Flujo principal: 1. El Lector estando en el espacio personal del Autor de su preferencia, o en alguna de sus obras específicas, elige la opción de Donar. 2. El sistema despliega un cuadro en donde el Lector puede elegir la cantidad de puntos que desea donar al Autor. 3. El Lector confirma su elección. 4. El Autor recibe la cantidad de puntos donados por el Lector y se le especifica el Lector que ha donado exitósamente.
Excepciones	Flujo alternativo: 1.1 Si el Lector no tiene una cuenta de PayPal registrada se le redirecciona al formulario de registro. 1.2 Si el Lector no tiene puntos en su cuenta se le notifica mediante un mensaje en un cuadro de texto sugiriendo que compre puntos para poder hacer donativos.
Postcondición	Puntos reducidos y gemas ganadas.

Gestionar Recompensas

Resumen	El Lector podrá ver el estado actual de sus colecciones de figuras.
Actores	Lector
Precondiciones	Tener una cantidad mínima de gemas.
Descripción	Flujo principal: 1. El Lector accede a la sección de colecciones desde su espacio personal haciendo click en el ícono de recompensas. 2. El Lector elige alguna de las colecciones que se despliegan en pantalla. 3. Se despliega en pantalla un cuadro que pide al Lector especificar la cantidad de gemas que desea invertir para conseguir una nueva figura. La probabilidad de encontrar una figura no repetida va desde el 0 % hasta el 100 % y está en función de la cantidad de gemas que desee invertir, con una mayor cantidad de gemas aumenta la probabilidad de encontrar una figura no repetida. 4. Cuando se hayan encontrado todas las figuras de una colección específica la opción "Conseguir figura" cambia a "Historieta desbloqueada", con lo cual el usuario lector accede a este material exclusivo.
Excepciones	Flujo alternativo: 3.1 Si el Lector no tiene un mínimo de gemas el sistema se lo notificará mediante un cuadro de texto advirtiéndole que no es posible solicitar una figura aleatoria. 4.1 En el caso de que la figura obtenida sea repetida el sistema devolverá una gema gratis al usuario lector.
Postcondición	Figura nueva encontrada. Historieta desbloqueada.

6.2.3. Diagramas de clases

En esta sección se describen las clases de software que participan en los dos casos de uso arriba expuestos. En la figura 2 se puede apreciar cómo se relacionan las clases que corresponden al caso de uso *Gestionar Donaciones*. En la figura 3 están las relaciones entre las clases correspondientes al caso de uso *Gestionar Recompensas*.

`Usuario`: Contiene los datos generales del usuario, como el nombre completo, fecha de nacimiento, etc. `Lector`: Contiene el *nickname* distintivo del lector. Esta clase está compuesta de la clase `TipsLector`. Existe una relación de asociación con la clase `Coleccion`. Por la cardinalidad se puede notar que un lector puede tener asociado varias colecciones y una colección puede tener asociada varios lectores.

`Autor`: Contiene el *nickname* distintivo del autor. A través de esta clase se interactúa con la clase `Obra`. Esta clase está compuesta de la clase `TipsAutor`. Tiene una asociación de agregación con las clases `Donaciones` y `Pago`.

`Donaciones`: Contiene los datos del donante y del autor a quien dona, la cantidad y la fecha en que se efectuó el donativo.

`TipsAutor`: Contiene el correo asociado a la cuenta paypal del autor y la cantidad de puntos actuales.

`TipsLector`: Contiene el correo asociado a la cuenta paypal del lector, la cantidad de puntos adquiridos y las gemas ganadas.

`Coleccion`: Contiene los datos de la colección: nombre de la colección, cantidad de figuras, etc. Esta clase está compuesta de las clases `Recompensa`.

`Recompensa`: Esta clase contiene la recompensa asociada a una colección determinada y una breve descripción de dicha recompensa.

`Figura`: Contiene los datos de la figura: nombre de la figura, numero de la figura y la imagen representativa de la figura.

`Descubiertas`: Esta clase permite conocer el estado particular de una figura que está asociada a un lector, a saber descubierta o no descubierta. Tal condición es dada por el atributo booleano `figDescubierta`.

`DescubiertasId`: Esta clase permite establecer el *ID* para la clase `Descubiertas`. Esto es porque en la persistencia la tabla *descubiertas* tiene una llave primaria compuesta.

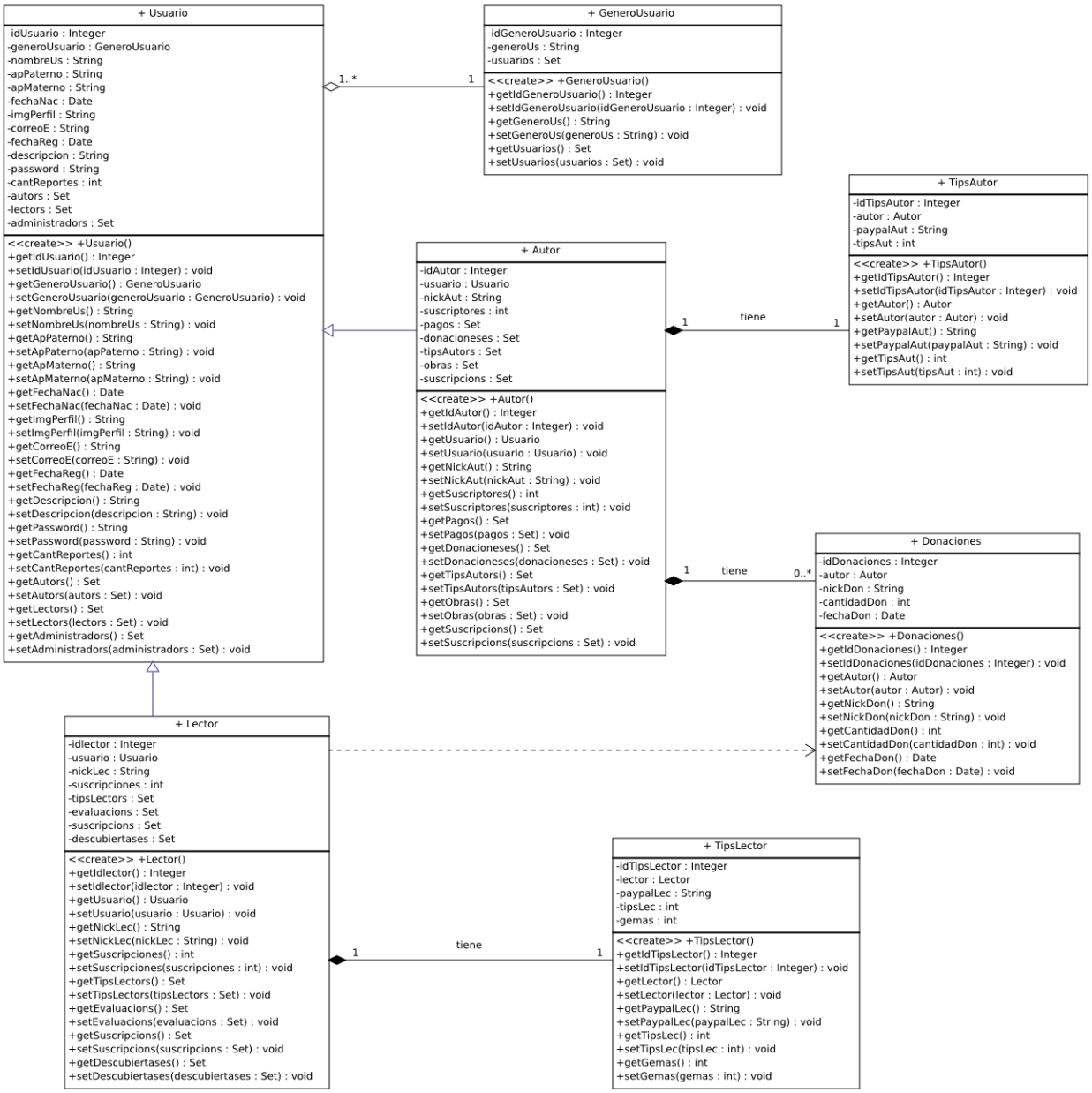


Figura 2: Diagrama de clases para el caso de uso Gestionar Donaciones.

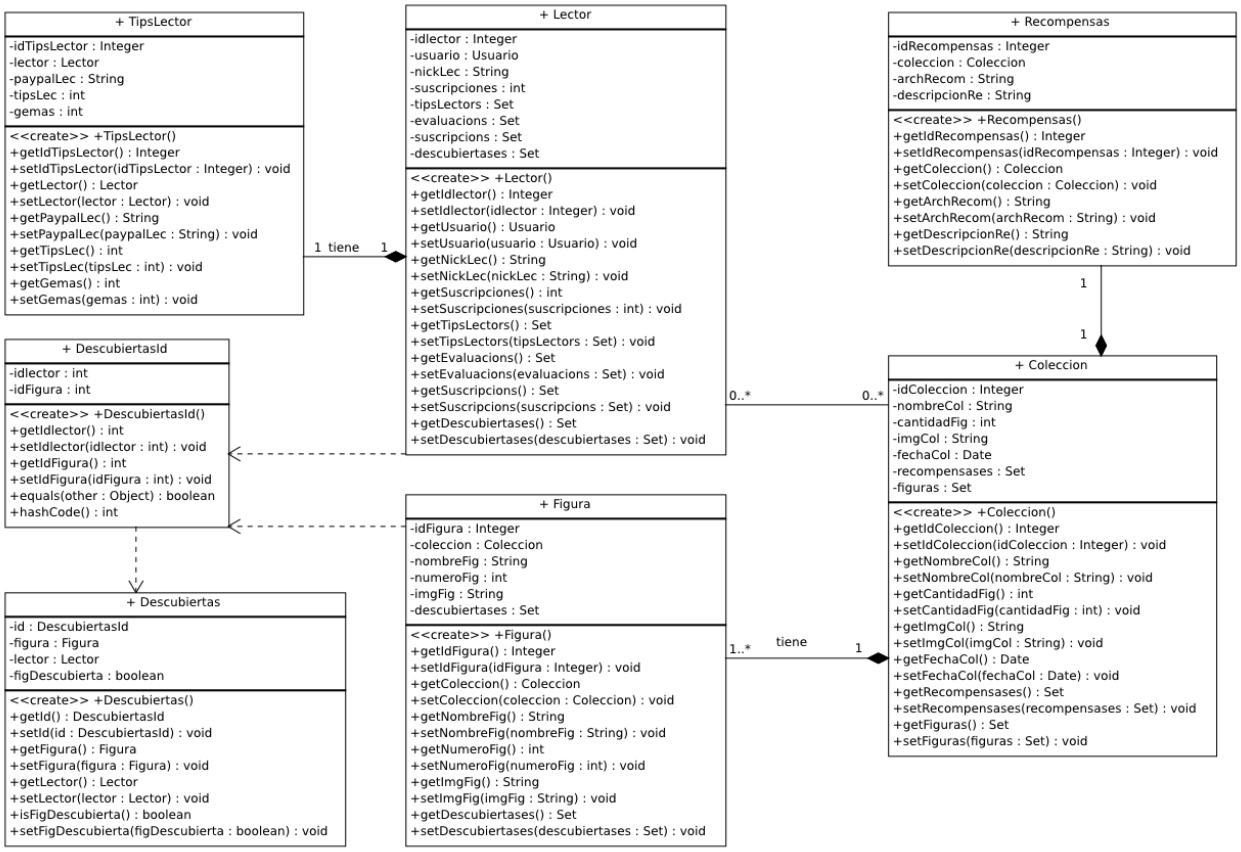


Figura 3: Diagrama de clases para el caso de uso Gestionar Recompensas.

Se construyeron también clases *DAO*[1] siguientes: `AdministradorDAO`, `AutorDAO`, `CapituloDAO`, `ClasifiDAO`, `GenObrDAO`, `DescuDAO`, `DonacionesDAO`, `ColeccDAO`, `EvaluacionDAO`, `GenUsrDAO`, `PagoDAO`, `FiguraDAO`, `ObrasDAO`, `RecompDAO`, `LectorDAO`, `UsuarioDAO`, `TipsAutDAO` y `TipsLecDAO`. En la figura 4 podemos observar cada una de estas clases. Cada una de las clases *DAO* tiene una instancia de la clase respectiva como se puede apreciar en la figura 5 a modo de ejemplo.

[1]Del acrónimo en inglés Data Access Object, es una pieza de software que provee una interfaz entre la lógica del sistema y la persistencia, por ejemplo una base de datos.

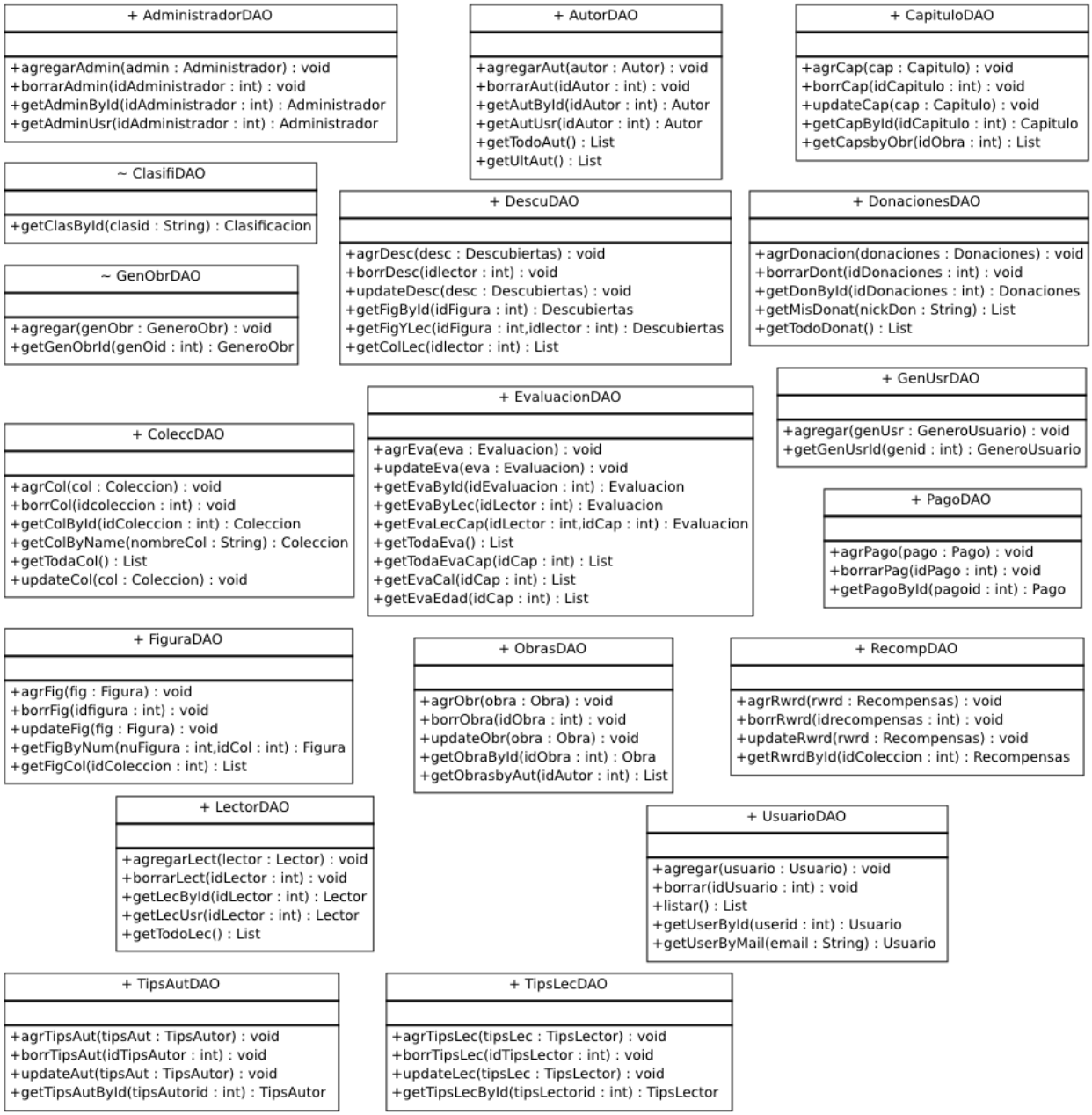


Figura 4: Clases DAO del sistema.

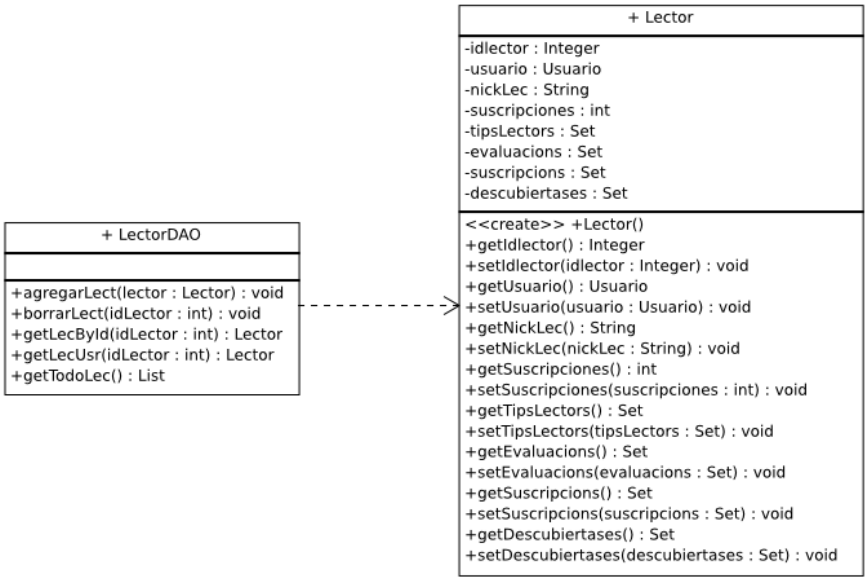


Figura 5: Dependencia entre la clase Lector y LectorDAO.

6.2.4. Arquitectura del sistema

Como se ha hecho mención, en el desarollo de Artico se usó la *framework Struts 2* que sigue el patrón de diseño MVC. El patrón de diseño MVC identifica tres elementos: modelo, vista y controlador. En *Struts 2*, estos son implantados por los elementos *action*, *result* y `FilterDispatcher` respectivamente [10]. La figura 6 muestra la forma en que *Struts 2* implanta este patrón para manejar el flujo de trabajo de las aplicaciones web. A continuación se describen los pasos que se ejecutan cuando un usuario realiza una petición, tal como lo muestra Srikanth [13]:

1. El servlet controlador maneja las peticiones de usuario. (Esto es, que el hipervínculo en el JSP debe apuntar al controlador del servlet).
2. El servlet controlador instancia los JavaBeans[2] apropiados, basados en los parámetros de petición (y opcionalmente también basado en atributos de sesión).
3. Luego el servlet controlador, siendo él mismo o a través de otro controlador, se comunica con un nivel medio o directamente con una base de datos para recuperar los datos requeridos.
4. El controlador coloca los JavaBeans resultantes en alguno de los siguientes contextos: request, session o application.
5. El controlador entonces manda a la petición a la siguiente vista en base a la URL solicitada.
6. La vista usa el JavaBean resultante del paso 4 para mostrar los datos en pantalla.

[2]Componente de arquitectura Java, que permite a componentes construidos en Java ser usados en ambientes de programación gráfica. [14]

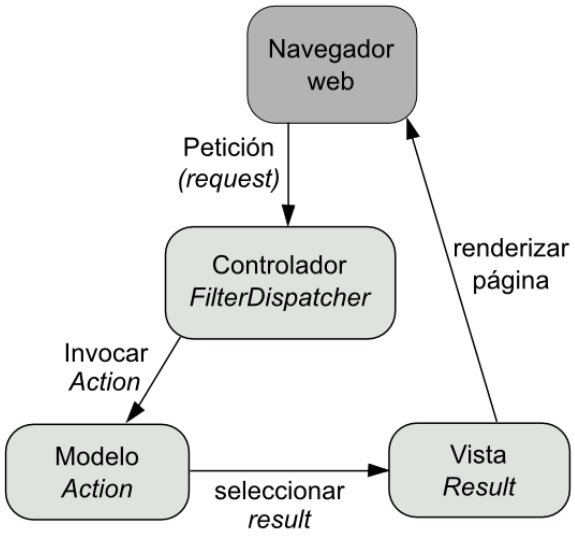


Figura 6: Modelo de la arquitectura MVC de Struts 2.

En la capa de persistencia de datos tenemos las interfaces de programación de aplicaciones (*API's*) que permiten usar la *framework Hibernate*. A continuación se presenta una clasificación de las interfaces principales de Hibernate:

- Interfaces llamadas por la aplicación para realizar operaciones *CRUD (Create Read Update Delete)* y operaciones de consulta. Incluyen `Session`, `Transaction` y `Query`.
- Interfaces llamadas por la infraestructura de la aplicación para configurar *Hibernate*, siendo la más importante la clase `Configuration`.
- Interfaces *Callback*, que permiten a la aplicación reaccionar a los eventos que ocurren dentro de Hibernate, tales como `Interceptor`, `Lifecycle` y `Validatable`.
- Interfaces que permiten la extensión de la funcionalidad de mapeo de *Hibernate*, tales como `UserType`, `CompositeUserType`, y `IdentifierGenarator`.

En la figura 7 se puede ver el Diagrama de la arquitectura de Hibernate.

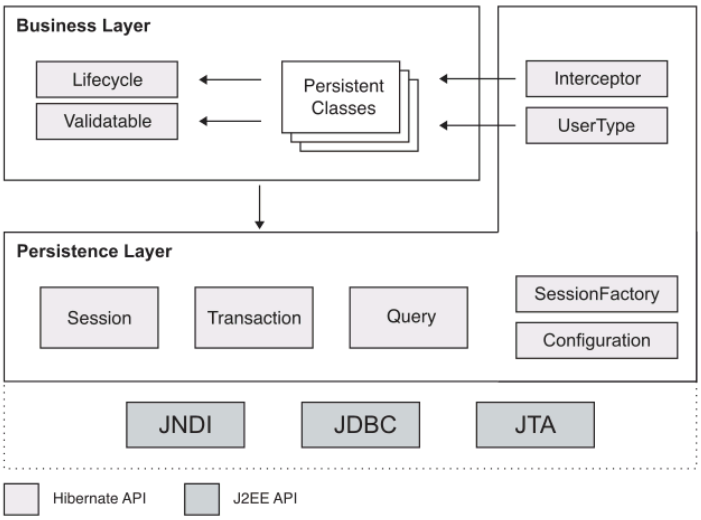


Figura 7: Estructura de directorios.

En la figura 8 se muestra la estructura de los directorios que componen el proyecto en netbeans.

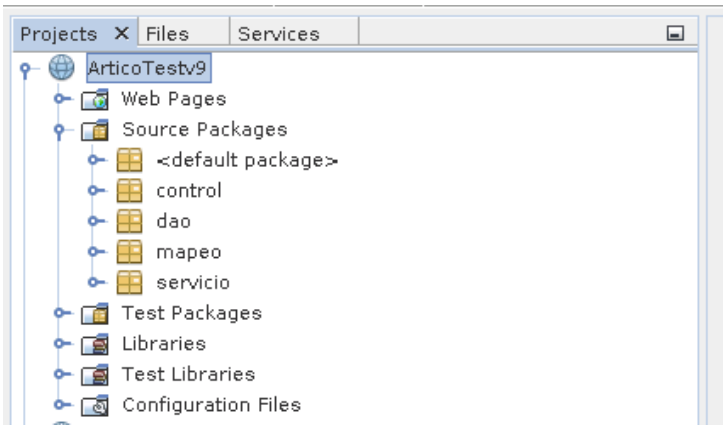


Figura 8: Estructura de directorios.

En la raíz de `Source Packages` se encuentran los archivos de configuración xml necesarios para describir la arquitectura de la framework Struts 2. También se encuentra aquí el archivo de configuración de hibernate, en el que describimos la conexión a la base de datos para recuperar y guardar la información.

El paquete `control` contiene las clases *action*. Estas clases se encargan de encapsular el trabajo actual a realizar en cada petición. Sirven también como los portadores de los datos que se mostrarán en las vistas.

En el paquete `dao` se encuentran las clases responsables de las operaciones que permiten la persistencia de los datos. Además de las operaciones típicas de altas, bajas y cambios, en estas clases se han definido operaciones de consulta específicas de acuerdo a las necesidades de Artico.

En el paquete `mapeo` se encuentran las clases generadas por Hibernate a través de ingeniería inversa, conocidas como *POJO*[3], así como los archivos de configuración xml para cada una. Esto permite asociar a cada una de las entidades existentes en la base de datos relacional, con las clases java en la aplicación.

En el paquete `servicio` se encuentran clases diseñadas para contener los datos necesarios asociados al perfil específico de un tipo de usuario, a saber: lector, autor y administrador.

6.2.5. Estructura de la base de datos

En la figura 9 se puede observar el modelo relacional de la base de datos usada por Artico. La aplicación se sirve de cada una de las relaciones mostradas, con la salvedad de las tablas `reporte` y `tipo_rep`, que son usadas en la aplicación del proyecto ***Aplicación Web para la gestión de historietas digitales***, y cuyo desarrollo corre a cargo de la alumna María Isabel Vega Mendoza.

[3]Del acrónimo en inglés Plain Old Java Objects

Figura 9: Modelo entidad relación de la base de datos.

A continuación se describen brevemente las tablas usadas por la aplicación en los casos de uso `Gestionar Donaciones` y `Gestionar Recompensas`.

Tabla `usuario`: Almacena los datos generales de un usuario como son su nombre, apellidos, correo, entre otros. En la figura 10 podemos ver la relacion que tiene con la tabla `genero_usuario` y la relación de generalización para con las tablas `administrador`, `lector` y `autor`.

Tabla `genero_usuario`: Catálogo de los géneros posibles para identificar a un usuario.

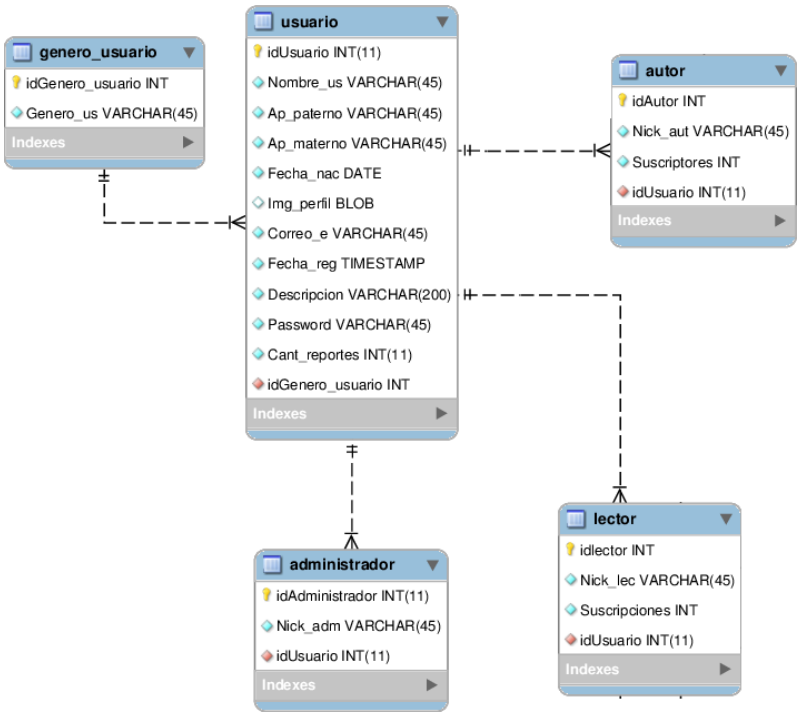


Figura 10: Relaciones con la tabla usuario.

Tabla `lector`: Almacena el *nickname* elegido por el lector y la cantidad de lectores a los que está suscrito. Se relaciona con la tabla `usuario` mediante una llave foránea.

Tabla `tips_lector`: Almacena el correo electrónico de la cuenta *PayPal* del lector, la cantidad de *tips* y *gemas*. Estos valores se asocian con un lector determinado, relacionándose con la tabla `lector` mediante una llave foránea.

Tabla `autor`: Almacena *nickname* eledigo por el autor y la cantidad de suscriptores que tiene. Al igual que el lector se relaciona con la tabla `usuario` mediante una llave foránea.

Tabla `tips_autor`: Almacena el correo electrónico de la cuenta *PayPal* del autor y la cantidad de tips que le han donado. Estos valores se asocian con un autor determinado, relacionándose con la tabla `autor` mediante una llave foránea.

Tabla `pago`: Almacena el monto de retiro del autor y la fecha en que se efectúa. Se relaciona con una cardinalidad múltiple con la tabla `autor` mediante una llave foránea.

Tabla `donaciones`: Almacena el *nickname* del lector donante, la cantidad donada y la fecha en que se efectuó la donación. Estos datos se relacionan con la tabla `autor` tal como se muestra en la figura 11, parte fundamental del caso de uso *Gestionar Donaciones*.

Figura 11: Tablas involucradas en la gestión de donaciones.

Tabla `descubiertas`: Almacena las llaves foráneas para mantener una relación entre las tablas `lector` y `figura`, tal como se muestra en la figura 12. También almacena un atributo booleano que indica si la figura en cuestión ha sido descubierta o no por un lector determinado.

Tabla `figura`: Almacena el nombre, número y la ruta del archivo de una figura. Esta tabla se relaciona con la tabla `coleccion`.

Tabla `coleccion`: Almacena los datos de una colección, como son su nombre, la cantidad de figuras, una imagen representativa, etc.

Tabla `recompensas`: Almacena el archivo de la recompensa y una breve descripción. Se relaciona con la tabla `coleccion` mediante su llave foránea.

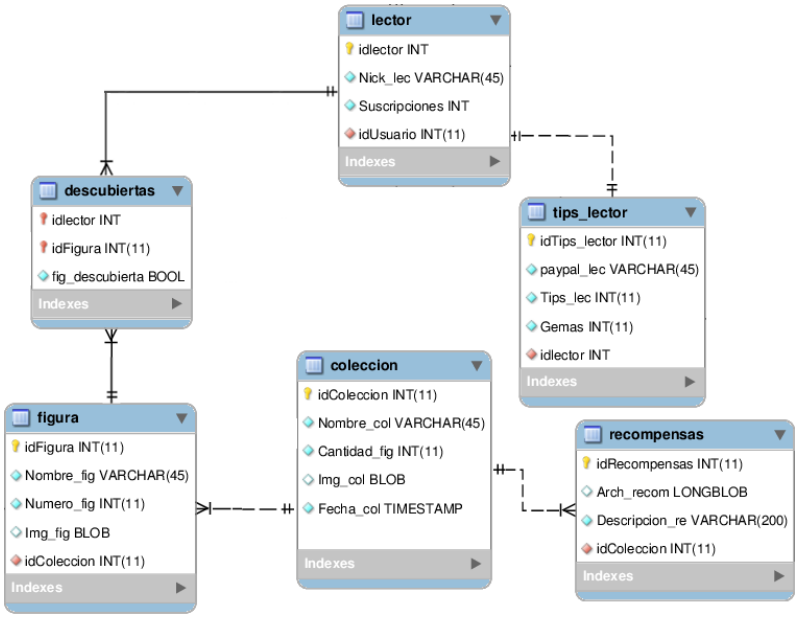


Figura 12: Tablas involucradas con la gestión de recompensas.

6.3. Diseño dinámico del sistema

En esta sección se muestran los diagramas de secuencia. Un diagrama de secuencia es un artefacto de diseño que muestra gráficamente los eventos que fluyen de los actores del sistema. [15] Se ejemplifica el curso particular de los eventos para los casos de uso *Gestionar Donaciones* y *Gestionar Recompensas* en los diagramas de secuencia a continuación presentados.

6.3.1. Diagramas de secuencia

En la figura 13 se muestra la interacción entre el `Lector` y Artico para efectuar una donación al Autor. El lector elige donar a un autor, mediante las `Donaciones` se van a verificar los datos del `Autor` y sus tips. Una vez que se conocen estos valores se indican los tips que se quieren donar. En `TipsLector` se verifican si los tips del lector son suficientes, se descuentan la cantidad de tips indicados y son estos mismos los que irán a `TipsAutor`. Se notifica de la donación al `Autor` y finalmente se despliega al `Lector` un mensaje indicando que la donación se efectuó exitosamente.

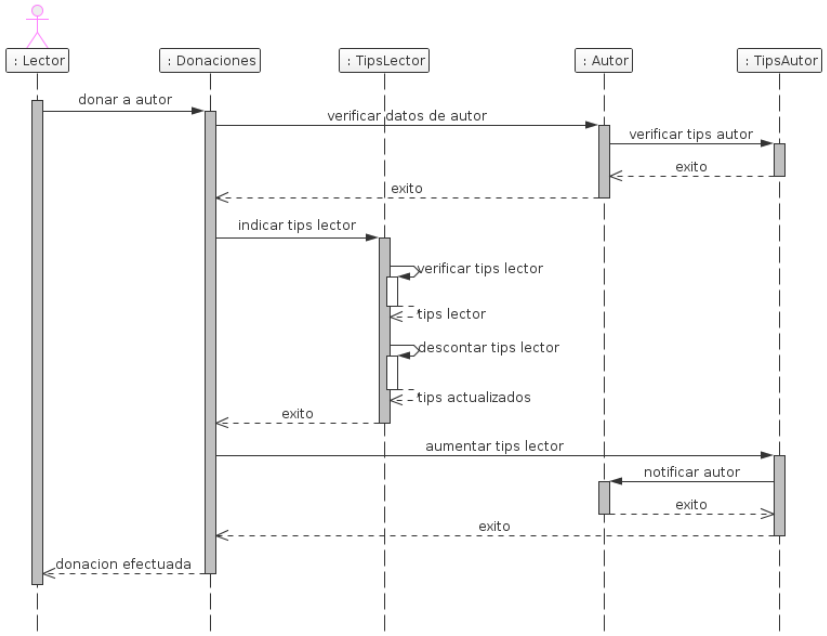


Figura 13: Diagrama de secuencia de gestión de donaciones.

En la figura 14 se muestra la interacción entre el `Lector` y Artico para gestionar sus recompensas. El recuadro *loop* indica un proceso iterativo al término del cual el `Lector` puede reclamar su recompensa.

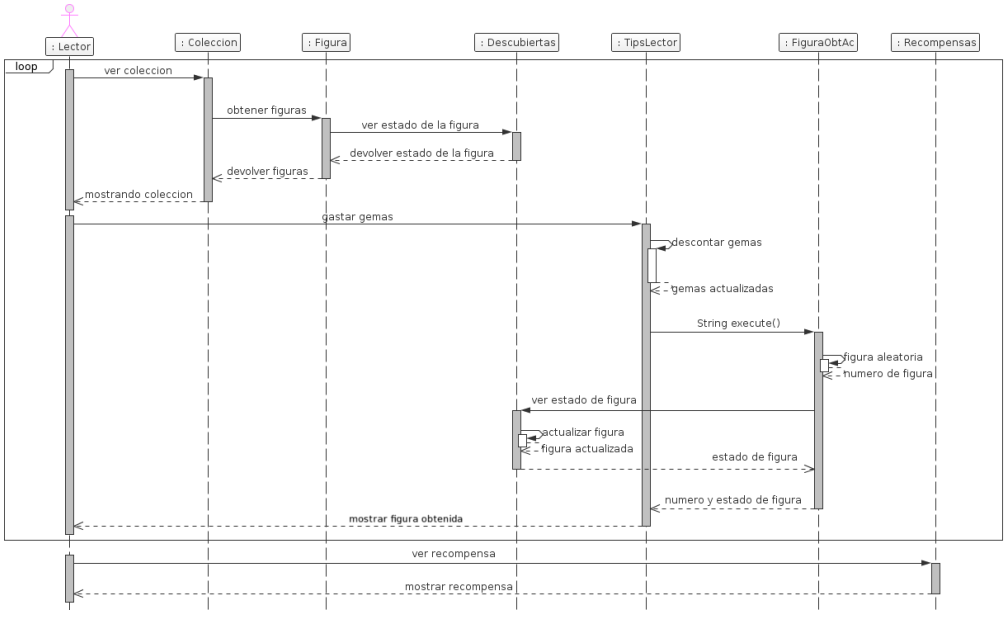


Figura 14: Diagrama de secuencia de gestión de recompensas.

6.4. Uso del sistema

6.4.1. Caso de uso Gestionar Donaciones

El usuario lector, visualizando el perfil del autor de su elección, elige la opción *Donar* (ver figura 15).

Figura 15: Eligiendo opción Donar.

A continuación se dirige al usuario lector a una vista en la que puede elegir una cantidad de puntos a donar. Por cada 5 puntos donados a un autor, Artico le ortogará al usuario lector una *gema* (ver figura 16).

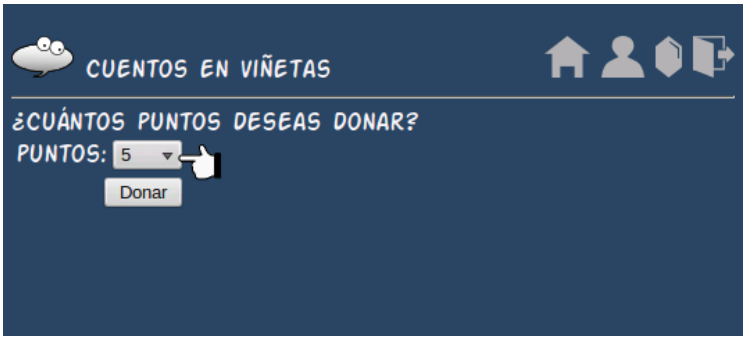


Figura 16: Eligiendo puntos a donar.

Posteriormente el usuario lector confirma su elección mediante el botón *Donar* como se muestra en la figura 17.

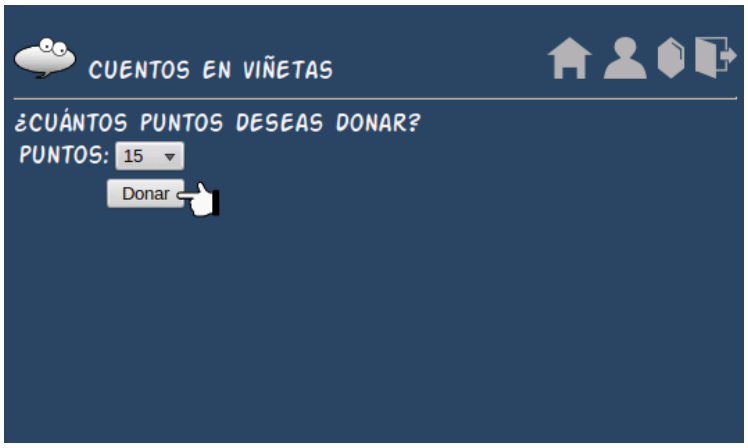


Figura 17: Confirmando elección.

Finalmente se redirige al usuario lector a la pantalla del espacio personal del autor que hubo elegido y aparece una leyenda en la que se le agradece por su donativo, confirmando además que la operación se llevó a cabo con éxito (ver figura 18)

Figura 18: Operación donar realizada con éxito.

Si el autor que elige el usuario lector no tiene aún su cuenta PayPal registrada, entonces en su perfil no aparecerá la opción *Donar*, como se muestra en la figura 19.

Figura 19: Vista del perfil de autor sin cuenta PayPal registrada.

En caso de que el usuario lector elija una cantidad de puntos superior a la cantidad actual de puntos que tiene en su cuenta, Artico lo redireccionará a una pantalla con el aviso como se puede ver en la figura 20.

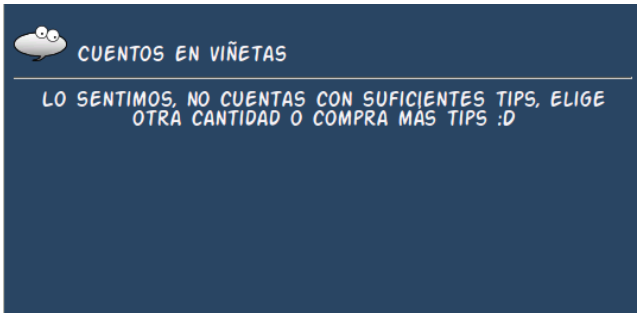


Figura 20: Usuario lector sin puntos suficientes.

Finalmente, si el usuario lector no tiene una cuenta PayPal registrada y elige la opción de *Donar* en el perfil de un autor, Artico lo redireccionará a la pantalla en donde puede hacer su registro para poder hacer uso de estas funciones en el sistema.

6.4.2. Caso de uso Gestionar Recompensas

Desde la pantalla de perfil del usuario lector se elige el ícono en forma de trofeo. Esta es la opción para ver las colecciones disponibles en Artico, eligiendo esta opción (ver figura 21) se redirige al usuario lector a la pantalla de colecciones.

Figura 21: Eligiendo ver colecciones.

En la pantalla de colecciones aparecen todas las colecciones disponibles en Artico. Cada imagen representa una colección y para acceder a las figuras que contiene basta con hacer click sobre la imagen (ver figura 22).

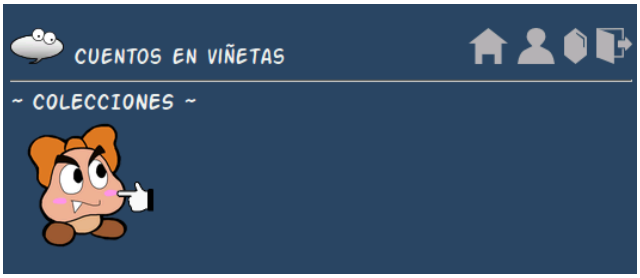


Figura 22: Eligiendo coleccion para ver figuras.

En la pantalla que corresponde a la colección elegida se muestran las figuras que el usuario lector puede obtener. Aquellas figuras que no han sido descubiertas se muestran como una silueta negra, como se puede apreciar en la figura 23. Para obtener una figura nueva se elige la opción *Conseguir Figura*, esto despliega una ventana emergente como se muestra en la figura 24. Es importante que el lector cuente con al menos tres gemas para iniciar la dinámica. En la ventana emergente se muestra la probabilidad de éxito para encontrar una figura nueva.

A medida que el usuario lector descubre figuras, la probabilidad de éxito disminuye.

Figura 23: Visualizando figuras de la colección elegida.

Figura 24: Gastando gemas extra.

Si se obtiene una figura nueva se redirecciona a una pantalla mostrando la figura obtenida y su número (ver figura 25. Luego de tres segundos Artico redirecciona al lector a la pantalla anterior, es decir, en donde se encuentran las figuras de la colección.

En la figura 26 se puede apreciar que la probabilidad de éxito para obtener una figura nueva disminuye. Para incrementar las probabilidades de éxito el usuario lector puede gastar una cantidad determinada de gemas extra.

Cuando el usuario lector completa la colección, es decir, ha obtenido todas las figuras, Artico deshabilita la opción para conseguir una figura nueva y desbloquea la opción para ir a la recompensa (ver figura 27).

Figura 25: Figura obtenida.

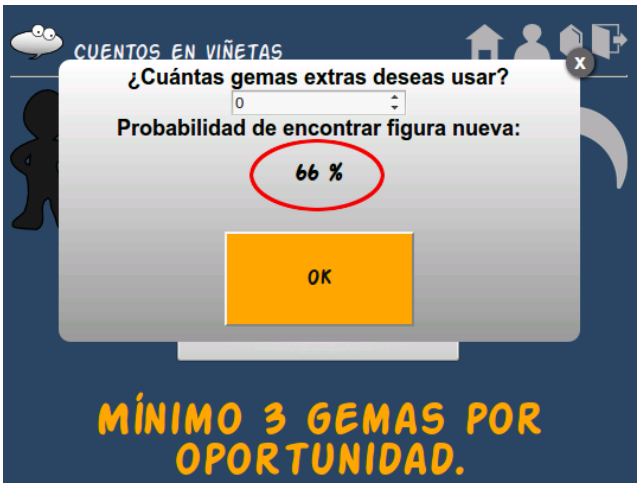


Figura 26: Gastando gemas extra.

Figura 27: Figura obtenida.

Finalmente el usuario lector tiene acceso a la historieta exclusiva que desbloqueó al juntar todas las figuras de una colección, para ver la recompensa se elije la opción *Obtener Recompensa* (ver figura 28).

Figura 28: Recompensa desbloqueada.

6.5. Hardware y software necesario

6.5.1. Tecnología para el desarrollo de la aplicación

En el desarrollo de la aplicación se utilizaron las herramientas: netbeans 8.2 y MySQL workbench 6.2. Los requisitos mínimos de harware para netbeans 8.2 son:

- Microsoft Windows XP Professional SP3/Vista SP1/Windows 7 Professional:
  - **Procesador:** 800MHz Intel Pentium III o equivalente
  - **Memoria:** 512 MB
  - **Espacio en disco:** 750 MB de espacio libre
- Ubuntu 9.10:
  - **Procesador:** 800MHz Intel Pentium III o equivalente
  - **Memoria:** 512 MB

    - **Espacio en disco:** 650 MB de espacio libre
- Macintosh OS X 10.7 Intel:
    - **Procesador:** Dual-Core Intel
    - **Memoria:** 2 GB
    - **Espacio en disco:** 650 MB de espacio libre

Los requisitos mínimos de hardware para MySQL workbench 6.2 son:

- **Procesador:** x86_32 o x86_64
- **Memoria:** 4 GB
- **Acelerador gráfico:** nVidia o ATI con soporte de OpenGL 2 o superior
- **Display:** 1024x768

El servidor de aplicaciones que se requiere para el desarrollo de la aplicación es Apache Tomcat 8, que viene integrado en el paquete de instalación de Netbeans 8.2.

Se requiere además Java SE Development Kit (JDK) 8 para la correcta instalación y configuración de Netbeans 8.2.

MySQL workbench necesita MySQL server instalado para su correcto funcionamiento.

6.5.2. Tecnología para la instalación y puesta en marcha de la aplicación

Las plataformas soportadas para la instalación de la aplicación son las siguientes (todas ellas de arquitectura x86_64:

- Apple:
    - Apple OS X v10.9+
- Microsoft:
    - Windows 10
    - Windows 8
    - Windows 2012 Server
    - Windows 7
    - Windows Vista
- Ubuntu:

- Ubuntu 16.10
- Ubuntu 16.04 LTS
- Ubuntu 14.04 LTS

La lista de plataformas soportadas está sujeta a la información que aparece en los sitios oficiales de los distintos proveedores de software. Pero cabe destacar que el sistema operativo en que se desarrolló, instaló y probó el funcionamiento de la aplicación, fue Debian 8, por lo que esta es también un plataforma perfectamente viable para la intalación y puesta en marcha de la aplicación.

7. Resultados

A continuación se detalla el grado de cumplimiento de cada uno de los módulos, de acuerdo a los objetivos particulares planteados para el desarrollo de Artico.

Módulo	**Cumplimiento**	**Descripción**
Registrar PayPal	Completo	El usuario registra su cuenta de correo electrónico de su cuenta PayPal y es guardada en la base de datos del sistema. Con la cuenta resitrada el usuario puede hacer uso de otras funciones en el sistema
Gestionar depósitos	Incompleto	Se simula dentro de la aplicación la adquisición de puntos y el retiro de efectivo a través de PayPal. Esta función no pudo llevarse a cabo, ya que para hacer uso de las API's de PayPal e integrarlos en la aplicación, se necesita una cuenta vinculada con una tarjeta de crédito. Esto es a razón de los pagos que se hacen por concepto de comisión a PayPal por cada transacción
Gestionar donaciones	Completo	Los lectores pueden donar a los autores de su preferencia y los autores ven reflejados estos donativos en sus cuentas.
Gestionar evaluación	Completo	Los lectores pueden emitir un comentario y una calificación a cada historieta. La interfaz es amigable con el usuario y el autor puede dar seguimiento de la recepción de su obra.
Gestionar recompensas	Completo	Los lectores pueden ver las distintas colecciones que tiene el sistema y al completar alguna de ellas desbloquean como recompensa una historieta exlusiva.

A modo de compensación por el módulo *Gestionar depósitos* incompleto, se desarrollaron e implantaron dos módulos extra. El primero es un módulo de *Registro de Usuario*, que permite, como su nombre lo indica, registrar en Artico usuarios de tipo lector o autor. Esto con el fin de tener una interfaz amigable para crear usuarios de prueba en el sistema. El segundo módulo es *Crear Colección*, consiste en una interfaz de usuario para los administradores del sistema, que les permite crear las colecciones con sus respectivas figuras y recompensas.

8. Análisis y discusión de resultados

A lo largo del desarrollo de Artico se realizaron pruebas unitarias para comprobar el correcto funcionamiento del sistema en cada una de sus etapas. Así se pudo verificar, por ejemplo, que la base de datos, que es responsable de la persistencia del sistema, fuera consistente con la manipulación de los datos; así se pudo verificar que la recuperación y almacenamiento de información fuera correcta en todo momento.Del mismo modo se comprobó la funcionalidad a nivel de la lógica del negocio y la usabilidad en la capa de presentación del sistema. Es importante destacar que también se llevaron a cabo pruebas de integración para validar el funcionamiento de los módulos en Artico.

El diseño y la arquitectura de la aplicación facilita su mantenibilidad y extensión, según futuras necesidades. Esto es posible gracias al desacomplamiento de las capas que lo componen: interfaz gráfica de usuario, modelo de la aplicación y persistencia de los datos.

La utilización de patrones de diseño de software, por mencionar algunos: el patrón *strategy*, *facade* o *factory*, por sus propiedades intrínsecas también facilitan la mantenibilidad y extensión del sistema. El patrón *strategy* se encuentra en el diseño de las clases *Action*, donde estas encapsulan todos los datos y los métodos necesarios para llevar a cabo una tarea específica. En las clases DAO encontramos una implantación del patrón *facade*, pues ellas encapsulan las consultas hacia la base de datos, siendo intermediarios entre la capa de datos y la capa de aplicación. Podemos encontrar el patrón *factory* en la creación y manejo de sesiones con la base de datos, gracias a la *framework* Hibernate. Por último, el patrón *Iterator* está presente en las operaciones que requieren un conjunto de datos específico.

También es destacable que las clases de software que componen la capa de aplicación son expertas en información. Este principio tiene la bondad de asignarles las responsabilidades solo a las clases que manejan la información pertinente. Así el diseño de la aplicación tiene una mayor cohesión y la información se encapsula, disminuyendo el acoplamiento.

9. Conclusiones

Artico es una plataforma web que permite recompensar a los lectores al donar a los autores, a través de los mecanismos implantados en el sistema. Este es el principio fundamental para promover la lectura de historieta digital de autores independientes, y que fue el objetivo general del proyecto. Artico también permite a los autores seguir mejorando sus proyectos al proporcionarles realimentación gracias a sus mecanismos de evaluación.

Entre los objetivos particulares que fueron propuestos en el proyecto, aquellos que se alcanzaron satisfactoriamente fueron los siguientes:

- Diseñar e implantar un módulo que gestione las donaciones a los autores de historieta.
- Diseñar e implantar un módulo que gestione recompensas a los lectores por sus donativos. Las recompensas consisten en historietas exclusivas desbloqueables.
- Diseñar e implantar un módulo de evaluación para las obras. Para valorar las obras se usará una variación de la escala Likert, con el que se medirá qué tan recomendable resulta una obra según el criterio de cada lector. A saber, los cinco niveles que comprenderá esta escala son: nada recomendable, poco recomendable, recomensable, muy recomendable y extremadamente recomendable. Cada uno de estos niveles será identificado por una cantidad de estrellas (que va de una hasta cinco).
- Integrar los módulos anteriores en una aplicación web.

El diseño e implantación de un módulo que se comunique con el servicio de pagos en línea *PayPal*, para gestionar los depósitos de los lectores, no se cumplió porque para hacer uso de las *API's* de este servicio se necesita una cuenta vinculada con una tarjeta de crédito.

Es notable cómo las buenas prácticas de desarrollo, la implantación de patrones de diseño y los artefactos de diseño de software, logran que se tenga una perspectiva clara de la evolución de un proyecto. El uso de estas herramientas facilitaron considerablemente la realización de Artico y la integración de colaboradores en pos de mejorarlo.

Con el desarrollo de Artico se integraron y aplicaron varios de los conocimientos adquiridos en las distintas UEA's cursadas a lo largo de la carrera. Pero un reto importante fue aprender a usar las *frameworks Hibernate* y *Struts 2*, pues fue la primera vez que se trabajó con ellas. Resulta una experiencia muy gratificante el utilizar la teoría y las técnicas aprendidas, para poder construir una aplicación funcional en base a las necesidades de un cliente. En un futuro, el presente proyecto podría sentar las bases para el desarrollo de una aplicación que impacte de manera positiva el mundo de la historieta mexicana independiente.

10. Perspectivas del proyecto

Por la naturaleza de su diseño, Artico es un sistema que está cerrado para la modificación, debido a que sus clases encapsulan únicamente los atributos y métodos de las que son responsables. A su vez, el modelo de tres capas permite que el sistema pueda extender sus funcionalidades, pues separa la presentación del modelo de dominio y de los datos.

Artico usa una base de datos relacional centralizada, lo cual presenta una oportunidad importante de mejora, pues se puede optar por una implantación de base de datos distribuida. Esto permitiría añadir robustez en el diseño general de la aplicación, por supuesto que esto implica nuevos retos.

La independencia de las capas que componen a Artico permiten mejorar y extender las funcionalidades de la aplicación, o bien, concluir los objetivos que no pudieron ser llevados a cabo.

Las interfaces gráficas de usuario, pese a ser funcionales, pueden mejorarse. Las representaciones de los íconos, la navegación y la distribución de los elementos se basa en el diseño de algunos sitios web, como lo son *tumblr* y *facebook*. Esto representa una oportunidad para estudiar y aplicar patrones de diseño *GUI* y proporcionar una identidad más original a Artico. Actualmente Artico despliega correctamente todos sus elementos visuales en los navegadores *Mozilla Firefox* y *Google Chrome*, por lo que también existe una oportunidad importante de expansión en cuanto a la compatibilidad con los demás navegadores, como *Opera* o *Safari*, sin mencionar los formatos *mobile*.

Por último, el siguiente paso lógico para Artico, es llevarlo a la fase de producción, es decir, que se pueda desplegar la aplicación en un servidor de aplicaciones web y probar su uso desde estaciones remotas. Con esto se puede experimentar la funcionalidad del sistema en un ambiente de trabajo real.

Bibliografía

[1] R. Roman y J. López, "Especificación de Sistemas Electrónicos de Microdonaciones", *III Simposio Español de Comercio Electrónico*, 2005.

[2] F. Esclapés Jover, M. Pérez Carrión, P.Pernias Peco, I. Ferreiro Prieto, M. Serrano Cardona, R. Pigem Boza y E. Juan Sepulcre, *Desarrollo y mejora de las estrategias de autoevaluación formativa de la asignatura de Fundamentos del Diseño Gráfico*, 1st ed. PDF, p.16.

[3] C. González González y A. Mora Carreño, "Técnicas de gamificación aplicadas en la docencia de Ingeniería Informática", *ReVisión*, vol. 8, no. 1, 2015.

[4] A. De La Roche Gutiérrez y M. Bosch Soler, "Carry Class: una plataforma de gamificación", *Universidad Politécnica de Catalunya*, 2015.

[5] J. Aguilar Herrera y O. Bañuelos Hernández, "Sistema de inventario y ventas para tlapalería con pronóstico de mercado", proyecto terminal, División de CB, Universidad Autónoma Metropolitana Azcapotzalco, México, 2009.

[6] K. Guzmán Villanueva y M. Gallardo López, "Aplicación móvil para la recomendación de productos para venta en comercio electrónico", proyecto terminal, División de CB, Universidad Autónoma Metropolitana Azcapotzalco, México, 2013.

[7] neobux.com, "The Innovation in Paid-to-Click Services". [Online]. Available: https://www.neobux.com/. [Accessed: 12-Apr-2017].

[8] Sergio Luján Mora, *Programación de aplicaciones web: historia, principios básicos y clientes web.*, Editorial Club Universitario, 2002, p. 111.

[9] Jeffrey L. Whitten y Lonnie D. Bentley, *Análisis de sistemas: diseño y métodos*, Mc Graw Hill, 2008, p. 6.

[10] Donald Brown, Chad Michael Davis y Scott Stanlick, *Struts 2 in Action*, Manning Publications Co., 2008.

[11] Christian Bauer y Gavin King, *Hibernate in Action*, Manning Publications Co., 2005.

[12] Ivar Jacobson, Grady Booch y James Rumbaugh, *El proceso unificado de desarrollo de software*, PEARSON EDUCACIÓN, 2000.

[13] Srikanth Shenoy, *Struts, survival guide*, ObjectSource LLC, 2005.

[14] Robert Englander, *Developing Java Beans*, O'Reilly, 1997.

[15] Craig Larman, *UML y Patrones, Introducción al análisis y diseño orientado a objetos*, Pearson Education, 2003.