Máster en Software Libre

Trabajo Final de Máster

Incorporación de funcionalidades a la red social *k-PAX*

Especialidad: **Administración web y comercio electrónico**

Institución: **FUOC – EIMT**

Alumno: **Juan Carlos Brocca Freydoz**

Consultor: **Francisco Javier Noguera Otero**

Tutor externo**: Daniel Riera Terren**

Fecha: **15 de junio de 2015**

Licencia de publicación

Copyright (c) 2015 Juan Carlos Brocca. Se otorga permiso para copiar, distribuir y/o modificar este documento bajo los términos de la Licencia de Documentación Libre de GNU, Versión 1.3 o cualquier otra versión posterior publicada por la Free Software Foundation; sin Secciones Invariantes ni Textos de Cubierta Delantera ni Textos de Cubierta Trasera. Una copia de la licencia está incluida en el *Anexo 1*, bajo el título *GNU Free Documentation License.* Puede obtenerse una copia de la última versión en https://www.gnu.org/copyleft/fdl.html

Resumen del proyecto

El presente trabajo se desarrolla a partir de una propuesta de los Estudios de Informática, Multimedia y Telecomunicación de la UOC, consistente en la incorporación de funcionalidades a la plataforma *k-PAX*[1].

En trabajos anteriores, otros alumnos han implementado nuevas competencias mediante modificaciones al código y/o incorporando *plugins*.

Pero al trabajar aislados y de manera independiente, estos desarrolladores pudieron realizar alteraciones que al momento de incorporarlas en conjunto, presenten inconvenientes de compatibilidad.

El objetivo principal del proyecto consiste en integrarlas, logrando sinergia entre ellas, sin afectar negativamente las propiedades y características de la plataforma.

En concreto, este proyecto parte de la versión estable del producto e incorpora dos trabajos:

- Una mejora visual para la presentación de juegos en la plataforma[2], que a la vez agrega funcionalidades para la administración y gestión de los mismos.
- Un módulo de búsquedas de juegos[3] que incluye el filtrado de acuerdo con diferentes criterios, algunas mejoras visuales y la determinación de semejanzas con otros juegos.

1 *k-PAX* es un Proyecto de Innovación Docente con financiación interna de la UOC, que tiene por objetivo el desarrollo de una red social para el aprendizaje basado en juegos, con posibilidades de ampliar sus funcionalidades mediante la incorporación de módulos.

2 Trabajo de Sergio Melgar Gonzalez.

3 Autoría de Víctor Corral Blanch.

Índice de Contenido

Licencia de publicación....2
Resumen del proyecto....3
1. Introducción....7
1.1 Estructura del documento....8
2. Estudio de viabilidad....9
2.1. Alcance del sistema....9
2.1.1 Objetivos del proyecto....9
2.1.2 Equipo de desarrollo....10
2.2. Estudio de la situación actual....10
2.2.1 El núcleo de *k-PAX*....10
2.2.2 Elgg....11
2.2.3 La base de datos *kpax*....11
2.2.4 Funcionalidades a integrar....12
2.2.4.1 Módulo desarrolladores....12
2.2.4.2 Módulo de búsqueda de juegos....13
2.3. Definición de los requisitos del sistema....13
2.4. Estudio de las alternativas de solución....13
2.5. Riesgos del proyecto....14
3. Análisis del sistema....15
3.1. Definición del sistema....15
3.1.1 Entorno tecnológico del proyecto....15
3.1.1.1 Elgg....15
3.1.1.2 El núcleo....17
3.1.1.3 La base de datos....18
3.1.2 Ambiente de desarrollo....19
3.1.3 Estándares y normas a aplicar....*20*
3.2. Usuarios del sistema....20
3.3. Módulos a integrar....21
3.3.1 Módulo desarrolladores....21
3.3.2 Módulo de búsqueda....24
3.4. Establecimiento de requisitos....26
3.5. Especificación del plan de pruebas....26
4. Diseño del sistema....27
4.1. Arquitectura inicial....27
4.1.1 Plugin kpax....28
4.1.2 Servicio SvrKpax....29
4.1.3 Base de datos kpax....30
4.2. Integración de módulos....31

4.2.1 Modificaciones sobre el plugin kpax.....31
4.2.1 Modificaciones al servicio SvrKpax.....33
4.2.3 Modificaciones en la base de datos.....35
5. Implementación.....38
5.1. Planificación de las actividades de desarrollo e integración de sistema.....38
5.2. Desarrollo.....38
5.2.1 Versión inicial.....38
5.2.2 Integración del módulo desarrolladores.....40
5.2.2.1 Reparación.....40
5.2.2.2 Incorporación.....43
5.2.3 Integración del módulo de búsqueda.....45
5.3 Producto final.....47
6. Conclusiones.....54
Bibliografía.....56

Índice de Figuras

Figura 1: Esquema de la plataforma (D. Riera et al).....7
Figura 2: Modelo Vista Controlador.....16
Figura 3: k-PAX. Versión oficial.....21
Figura 4: Pestaña Games – Módulo desarrolladores.....22
Figura 5: Pantalla de listado de juegos – Módulo de búsqueda.....25
Figura 6: Interconexión entre Elgg y el núcleo k-PAX.....27
Figura 7: Diagrama de clases.....29
Figura 8: Diseño inicial de la base de datos kpax.....30
Figura 9: Botón de acceso a la búsqueda de juegos.....31
Figura 10: Modulo desarrolladores – Evolución información de los juegos.....35
Figura 11: Modulo de búsqueda – Evolución información de los juegos.....36
Figura 12: Evolución de la tabla Game.....36
Figura 13: Diseño definitivo de la base de datos kpax.....37
Figura 14: Carga de un juego en la versión estable de k-PAX.....39
Figura 15: Formulario para la carga de un juego.....42
Figura 16: Vista de juegos no autorizados.....42
Figura 17: Consultas en MySQL.....43
Figura 18: Vista de la página de un juego.....44
Figura 19: Vista de juegos.....48
Figura 20: Búsqueda de juegos.....49
Figura 21: Filtrado en la búsqueda de juegos.....50
Figura 22: Descripción de un juego.....51
Figura 23: Edición de un juego.....52
Figura 24: Juegos del usuario.....53

Índice de Tablas

Tabla1.....11
Tabla2.....17
Tabla3.....19
Tabla4.....23
Tabla5.....24
Tabla6.....29
Tabla7.....32
Tabla8.....33
Tabla9.....45
Tabla10.....46
Tabla11.....47

1. Introducción

La *ludificación* [1] es una herramienta poderosa que ofrece muy buenas posibilidades de formación a diferentes colectivos.

Puede pensarse que un juego es un sistema en el que los jugadores participan en un desafío abstracto, definido por reglas, interactividad y retroalimentación, que se traduce en un resultado cuantificable, provocando a menudo una reacción emocional y donde cada uno de esos factores influye sobre los otros [2].

Los denominados *juegos serios* tienen un propósito educativo explícito, cuidadosamente planeado y no están pensados para ser jugados únicamente por diversión [3].

Con la finalidad de potenciar el aprendizaje basado en juegos haciendo uso de la capacidad que ofrecen las redes sociales actuales y la proliferación de dispositivos electrónicos de comunicación, nació el proyecto de innovación k-*PAX*, una plataforma tecnológica libre[4], concebida como red social, para facilitar la acción de aprender jugando en red, allí donde exista conectividad y se disponga de algún dispositivo adecuado [4].

Tal lo explicitado en el documento de presentación [5], la idea principal de ese proyecto es ofrecer un núcleo y un conjunto de servicios que permitan a la comunidad acceder a juegos, jugar, competir, colaborar, o crear nuevos juegos, entre otros.

La Figura 1 – tomada del mismo material – esquematiza la plataforma.

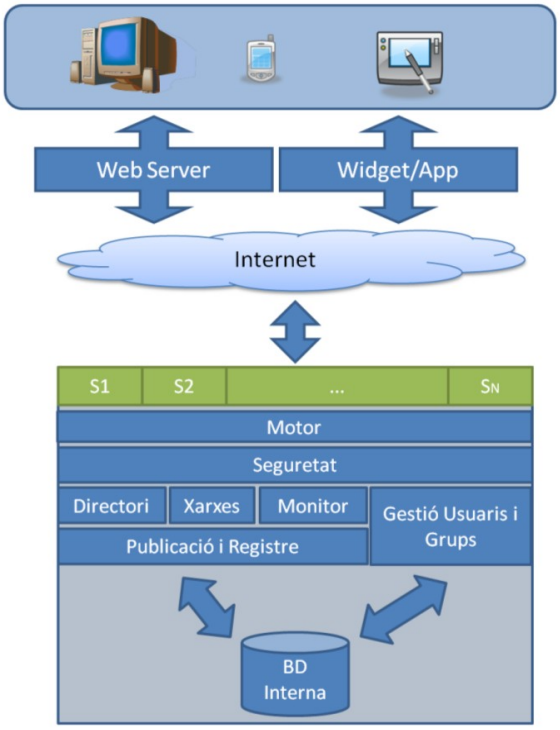

Figura 1: Esquema de la plataforma (D. Riera et al)

Los clientes se conectan con el núcleo mediante llamadas a los servicios, por lo que el

4 Publicada bajo la licencia *GNU GPL v2* de la *Free Software Foundation*.

funcionamiento interno resulta totalmente transparente.

Actualmente *k-PAX* está implementada sobre la plataforma *Elgg* [6], una red social versátil que permite modificar su comportamiento y/o potenciar su funcionalidad mediante el agregado de complementos o *plugins*[5]. Dispone así de sus propias funcionalidades y las que aporta *Elgg*.

Al momento, algunos de los requerimientos del proyecto original no han sido incorporados, pero las características de diseño y la licencia facilitan su futura inclusión.

Para ello se dispone de la versión oficial del código – alojado en la forja *Github*[6] – *y* de funcionalidades, nuevas o faltantes, desarrolladas por otros estudiantes que deberán ser integradas con condiciones de usabilidad.

Particularmente se ha trabajado con el código original y dos desarrollos de terceros:

- *Módulo desarrolladores*, implementación de Sergio Melgar [7] – propone una mejora visual respecto de la presentación de juegos y una rápida gestión de los mismos por parte de los desarrolladores y/o administradores de la plataforma.
- *Módulo de búsqueda de juegos*, desarrollo de Víctor Corral [8] – mejora el módulo de búsqueda de juegos de acuerdo con diferentes criterios.

1.1 Estructura del documento

La presente memoria consta de los siguientes capítulos:

- *Introducción*. Breve presentación del trabajo.
- *Estudio de viabilidad.* Establece el alcance del sistema, los objetivos del proyecto, el estado del arte de la plataforma y la solución propuesta.
- *Análisis del sistema*. Especificación detallada de la solución escogida, enmarcada en el entorno tecnológico de la plataforma.
- *Diseño del sistema.* Describe la arquitectura del sistema, el diseño de la base de datos y las diferentes modificaciones propuestas.
- *Implementación*. Se explica el proceso de integración, detallando problemas y soluciones.
- *Conclusiones*. Detalla los objetivos alcanzados y aquellos que quedaron pendientes, propuestas de mejoras futuras y la opinión personal.

5 Por defecto, *Elgg* incorpora un conjunto de complementos que le aportan funcionalidades básicas de red social.

6 Implementa el software de control de versiones *Git* basado en la web y también ofrece varias herramientas de colaboración.

2. Estudio de viabilidad

k-PAX presenta un ambicioso conjunto de requerimientos iniciales y, para mejorar su funcionalidad, una lista – no cerrada – de tareas pendientes con propuestas a realizar[7].

Al estar el proyecto dentro del ámbito académico, algunas de dichas funcionalidades han sido desarrolladas en otros trabajos, pero no incorporadas aún al código base de la plataforma.

En este capítulo se analiza la factibilidad de incorporar dos de ellas, de acuerdo con restricciones establecidas.

2.1. Alcance del sistema

El proyecto tiene como propósito el análisis de extensiones propuestas por el cliente[8] y su incorporación a *k-PAX* solucionando problemas de integración y compatibilidad.

Con cada integración se genera una nueva versión operativa de la plataforma, que incluye estas funcionalidades.

La viabilidad técnica y legal del proyecto está garantizada, tanto por la licencia de la plataforma como por las herramientas empleadas. Ninguno de los módulos analizados presenta incompatibilidades al respecto.

Desde el punto de vista de la viabilidad operativa se ha contemplado que:

- La integración de cada nueva funcionalidad no interfiera con el funcionamiento general de la plataforma.
- Cada funcionalidad incorporada, no debe perjudicar a las existentes.

Por otra parte, no ha tenido en cuenta:

- El desarrollo de nuevas prestaciones.
- El análisis de costos y/o presupuesto asignado.

2.1.1 Objetivos del proyecto

Se establece como principal objetivo, lograr compatibilidad en la integración de distintas funcionalidades ya desarrolladas para la plataforma *k-PAX* y que no están disponibles en la versión inicial.

Para lograrlo, es necesario:

- Comprender la arquitectura de *k-PAX* y los componentes que la forman.
- Entender la conexión entre los módulos, la base de datos y los servicios.
- Reconocer el ambiente de implantación.
- Definir la metodología de desarrollo del proyecto a utilizar.

7 https://github.com/jsanchezramos/k-pax/wiki/TODO-list

8 Actividad representada por el tutor externo.

- Obtener la versión inicial de *k-PAX*.
- Recopilar y analizar las diferentes funcionalidades a implementar.
- Incorporarlas de manera incremental, obteniendo una nueva versión estable que las incluya.
- Añadir la nueva versión a la rama pública estable en *GitHub.*

2.1.2 Equipo de desarrollo

Dadas las características laborales que presentan los integrantes del curso y las conversaciones mantenidas con los tutores, se acuerda en realizar parte de los trabajos en equipo.

Se propone llevar a cabo la actividad mediante la aplicación de una metodología ágil – adaptada desde *SCRUM* [9] – estableciendo subobjetivos a partir de cada integración, que permitan realizar progresivamente la tarea.

La elección de esta metodología evita tener que establecer precisiones con muy alto nivel de detalle sobre las cuestiones de diseño y desarrollo en un momento donde el dominio del problema no está del todo claro.

Los principales roles recaen en:

Daniel Riera Terren – *Product Owner*

Francisco Javier Noguera Otero – *ScrumMaster*

Cecilia Cámera López y Juan Carlos Brocca Freydoz – *Team.*

2.2. Estudio de la situación actual

Actualmente *k-PAX*:

- Está implementada sobre la red social *Elgg,* encargada de gestionar algunos de los servicios ofrecidos.
- Dispone de un núcleo de servicios Web – *srvKpax* – que permite gestionar usuarios, desarrolladores, juegos y partidas.
- Cuenta con su propia base de datos que almacena la información necesaria para el correcto funcionamiento de la plataforma.

2.2.1 El núcleo de *k-PAX*

El núcleo *srvKpax s*e ejecuta desde un servidor de aplicaciones encargado de proporcionar servicios web de tipo *REST* [10] para acceder a sus funcionalidades.

Además, ofrece operaciones *Commit, Read, Update, Delete (CRUD)* [11] sobre la base de datos, la que es administrada mediante el DBMS *MySQL* – versión 5.

Está desarrollado como una aplicación java EE mediante el gestor de proyectos *Apache Maven* [12] que, además de definir su estructura, facilita la compilación y gestión de dependencias.

Se despliega sobre un servidor *jBoss*, [13] implementando los servicios REST mediante *Jersey* [14], el contenedor de beans *Spring [15]* y, para la persistencia, utiliza la biblioteca *Hibernate* [16] como Mapeador Objeto-Relacional[9].

2.2.2 Elgg

La red social *Elgg* – sobre la que está implementada *k-PAX* – es un sistema de gestión de contenidos de código abierto[10] compatible con una serie de estándares[11] y que corre en entornos del tipo *XAMP*[12], permitiendo crear y administrar redes sociales a la vez que ofrece un alto grado de control sobre el acceso a los contenidos.

En la documentación de *Elgg* se desalienta enfáticamente a modificar su código, esgrimiendo como razones las posteriores dificultades de actualización, agujeros de seguridad o roturas de funcionalidades existentes. Propone que los cambios y nuevas características se implementen mediante *plugins*.

Así, mediante un conjunto de conectores se desarrollaron las diferentes características de *k-PAX*, que se comunican con el núcleo mediante llamadas a los servicios web.

En la Tabla 1 se describen los cuatro *plugins* básicos que permiten el funcionamiento de *k-PAX* [17].

Conector	Descripción
loginrequired	Oculta a los usuarios no autenticados todas las páginas de *Elgg*, a excepción de la de inicio, la de registro y la de recuperación de credenciales.
kpax	Añade a *Elgg* las funcionalidades de gestión de juegos, siendo el encargado de comunicar, efectuando las correspondientes llamadas, con el núcleo de servicios de *k-PAX.*
apiadmin	Se ocupa de la gestión de los certificados de autenticación.
likekpax	Gestiona las anotaciones *like this* en los objetos *k-PAX.*

2.2.3 La base de datos *kpax*

El diseño inicial de la base de datos de *k-PAX* [5], permite almacenar información relacionada con los usuarios, los juegos y las partidas.

Las tablas de esa base pueden agruparse contemplando las siguientes características:

9 Más conocido por sus siglas en inglés, *ORM*, mapea las clases a tablas generando automáticamente todo el SQL necesario para almacenar el objeto en una base de datos.

10 Disponible bajo licencia dual, *GNU GPL v2* y la *licencia MIT*. Sin embargo, los *plugins* solo pueden utilizarse bajo la licencia *GPL.*

11 Entre otros, RSS, LDAP, FOAF, y XML-RPC.

12 *XAMP*: Apache – MySQL - PHP, disponible para varias plataformas, tal el caso de *LAMP* – GNU/Linux – o *WAMP*, corriendo sobre Windows®.

- **Usuarios**

  Mantiene registro de los usuarios del sistema, quienes pueden acceder a la plataforma de forma directa o remota, a través de otra red social.

- **Grupos**

  Para gestionar el acceso a juegos, los usuarios pueden organizarse en grupos. Se refleja la asociación de los usuarios a distintos grupos mediante una tabla.

- **Juegos**

  Para ser accesibles desde la plataforma, los juegos deben estar registrados. Se requiere almacenar sus características.

- **Acceso a los juegos**

  Es posible asignar permisos de acceso a determinados juegos a los diferentes grupos. La situación se refleja en una tabla.

Ya con la plataforma en funcionamiento, se genera información dinámica que permite garantizar la seguridad del sistema y gestionar las diferentes partidas que se estén jugando:

- **Sesión**

  Cuando un usuario se conecta al sistema, se crea una sesión asociada a esa conexión que se mantiene almacenada mientras dure.

- **Partidas**

  Para poder jugar, los usuarios deben unirse a una instancia de un determinado juego.

2.2.4 Funcionalidades a integrar

Partiendo del código original disponible en *Github* [18] se realizarán las tareas correspondientes para la integración y puesta en funcionamiento en conjunto de desarrollos de terceros, de acuerdo con los requerimientos del cliente y cuyas características se detallan a continuación.

2.2.4.1 Módulo desarrolladores

Se trata del proyecto final de carrera de Sergio Melgar González en la Universitat Autònoma de Barcelona [7], desarrollado sobre la base de trabajos previos.

Representa una mejora visual en la presentación de la información de los juegos y un sistema de gestión para los mismos.

También pone en valor la función del desarrollador[13] que ahora puede añadir detalles al juego que desea publicar[14], presentándolo de manera más intuitiva a la vista.

13 Se define a un desarrollador como cualquier usuario que ha publicado al menos un juego.

14 Tales como imágenes, un vídeo o una descripción.

Lamentablemente, la copia que pudo obtenerse del código de este producto no estaba operativa.

2.2.4.2 Módulo de búsqueda de juegos

En su Trabajo Final de Máster, Víctor Corral Blanch [8] desarrolló un sistema de búsquedas con capacidad de filtrado, ordenación y paginación, basándose a la vez en trabajos similares preexistentes.

Esta funcionalidad – disponible para todos perfiles de usuario – permite realizar búsquedas de juegos por categorías, plataforma, habilidad y por etiquetas. Establece semejanzas con otros juegos y opcionalmente, permite realizar búsquedas por metadatos.

Las funciones están implementadas y operativas, pero la opción de añadir juegos a la plataforma no funciona, por lo que requiere que la gestión de juegos se realice directamente a través del acceso manual a la base de datos.

Tampoco están disponibles interfaces de usuario para introducir datos de categorías, habilidades y plataformas.

2.3. Definición de los requisitos del sistema

Para cumplir con los objetivos del proyecto, sobre cada funcionalidad a incorporar se tienen en cuenta los siguientes requisitos:

R1. Todo módulo a integrar debe estar operativo, al menos en las funciones básicas para las que fueron diseñados.

R2. La comunicación con la base de datos debe realizarse siempre a través de los servicios de *srvKpax.*

R3. La modificación de alguna tabla de la base de datos no debe interferir con las prestaciones de otros módulos que las utilicen.

R4. Los cambios en el código – tanto del núcleo como de los *plugins* – no deben obstaculizar a otros servicios o funcionalidades.

R5. La incorporación de un nuevo módulo no debe interferir con las funcionalidades existentes, salvo que se trate de un reemplazo o modificación específica.

2.4. Estudio de las alternativas de solución

Dado que durante el trabajo no se llevan a cabo tareas de diseño y se utiliza código desarrollado por terceros, debe partirse necesariamente de la versión estable del producto e incorporar uno a uno los distintos módulos, verificando previamente su funcionamiento.

Por requerimientos del cliente, el orden de integración es el mismo que el de la descripción en el punto 2.2.4.

Asimismo, fue consensuado que la instalación inicial y la incorporación del primer módulo se realice en equipo, mientras que el trabajo sobre segundo se ejecute en forma independiente.

2.5. Riesgos del proyecto

Atentan contra el desarrollo del proyecto principalmente:

- La falta de experiencia de los participantes en este tipo de actividad, dando lugar a estimaciones de tiempo y esfuerzos inadecuados.
- La superposición de la tarea con las actividades laborales propias de los participantes.
- Retrasos en la comunicación entre tutores y participantes.
- Desconocimiento sobre el grado de calidad de los desarrollos a incorporar.
- Dificultades de interoperabilidad.

3. Análisis del sistema

Sobre la base de los requerimientos descritos anteriormente, en este capítulo se realiza el planteamiento tecnológico detallado de la solución.

3.1. Definición del sistema

Como fue mencionado, el punto de partida para el proyecto será la rama estable del proyecto, para el caso la bifurcación del código original que realizara el tutor externo[15], desde donde se crea un *fork* tanto para el núcleo como para los módulos que permiten el funcionamiento de *k-PAX* sobre la plataforma *Elgg.*

El código de los trabajos de terceros es provisto por el *Product Owner*.

Por cada funcionalidad a incorporar se realizarán los cambios necesarios al código y las bases de datos, realizando las pruebas correspondientes.

Finalizado el trabajo se enviará al propietario del repositorio la petición para que incorpore los *commits.*

3.1.1 Entorno tecnológico del proyecto

Tal como se ha mencionado en el capítulo anterior, *k-PAX* se ha diseñado utilizando diversas tecnologías y herramientas, entre las que corresponde destacar:

- *Elgg*, un marco de red social desarrollado íntegramente en PHP igual que sus conectores, que le permiten a *k-PAX* contar con una red social en producción que gestiona servicios no ofrecidos aun por su núcleo y facilita la visualización de los datos.
- La plataforma *Java Enterprise Edition* – Java EE – un conjunto de especificaciones que facilitan el desarrollo y despliegue de aplicaciones web multi-capa.
- El Sistema Gestor de Bases de Datos MySQL, utilizado tanto por *Elgg* como por *k-PAX*.

Las secciones siguientes describen el entorno con mayor precisión.

3.1.1.1 Elgg

Elgg sigue un modelo *MVC*[16] para su desarrollo. Se trata de un patrón de arquitectura de software que separa la lógica de negocio de la interfaz de usuario mediante tres componentes distintos, facilitando la evolución por separado de cada aspecto y ofreciendo a la vez muy buena flexibilidad para el desarrollador.

En *MVC* [19], El *Modelo* representa los datos de la aplicación junto con las reglas de negocio que se utilizan para manipularlos, la *Vista* es alguna forma de visualización

15 https://github.com/drierat

16 Modelo Vista Controlador.

del estado del modelo y el *Controlador* se encarga de capturar y propagar los eventos a la vista y al modelo.

El funcionamiento, esquematizado en la Figura 2, es el siguiente:

1. El usuario, vía URL, envía una solicitud al *Controlador*.
2. El *Controlador* captura el evento y efectúa la llamada al *Modelo* de acuerdo con el pedido recibido.
3. El *Modelo*, luego de interactuar con la base de datos, envía los datos al *Controlador*.
4. El *Controlador* envía los datos a la *Vista*.
5. La *Vista* procesará los datos según el diseño solicitado para la visualización y los entregará al usuario en el formato adecuado.

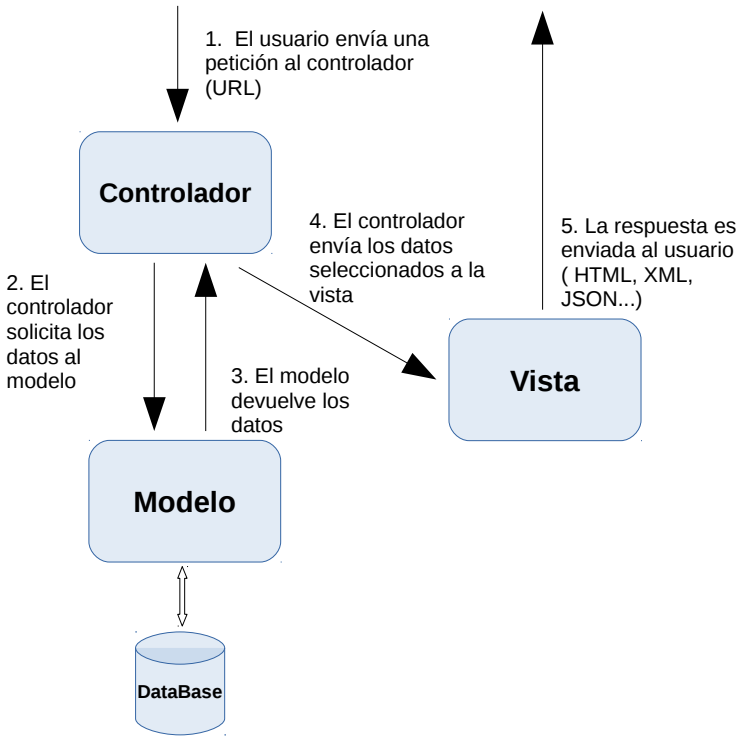

Figura 2: Modelo Vista Controlador

También se ha mencionado que en *Elgg* es factible la implementación de nuevas funcionalidades o modificación de sus características mediante conectores.

Esos *plugins* deben responder a una estructura específica [20], que obliga a incluirlos en el directorio *mod* del sistema de archivos de *Elgg,* dentro de un subdirectorio de igual nombre que el *plugin*, al que se considera su directorio raíz.

En el mismo deben estar contenidos los archivos *start.php* y *manifest.xml* para permitir que el plugin sea reconocido por el motor *Elgg*.

Los principales elementos están resumidos en la Tabla 2.

Elemento	Descripción
Archivo ***start.php***	Script en PHP que constituye el centro de control del *plugin*. Contiene la función de inicio y establece llamadas que definirán el comportamiento general del mismo.
Archivo ***manifest.xml***	Metadatos referidos al *plugin*. Contiene información sobre las dependencias.
Directorio ***actions***	Contiene archivos en PHP que permiten ejecutar las acciones definidas en el *plugin*.
Directorio ***lib***	En este directorio se implementan funciones necesarias para la funcionalidad del *plugin*.
Directorio ***graphics***	Contenedor de archivos de imágenes.
Directorio ***views***	En este directorio se organizan los formularios, vistas y archivos de estilo que permiten presentar la información. Las vistas aquí definidas anulan automáticamente a las que pudieran existir en el núcleo.
Directorio ***languages***	Contiene archivos PHP con las traducciones de las frases incluidas en el tema. Existen tantos archivos como idiomas contemplados.
Directorio ***pages***	Contenedor de páginas definidas para el *plugin*.

3.1.1.2 El núcleo

Está desarrollado como una aplicación web Java EE, agrupado en cinco paquetes:

- **VO**[17] *Representación de tablas en objetos.*

  Establece la definición de cada tabla de la base de datos como un objeto.

  Contiene clases serializables[18] – al menos una por cada tabla de la base – con propiedades accesibles mediante métodos *setter* y *getter*. Están almacenadas en archivos con nomenclatura *NombreTabla*.java.

- **DAO**[19] *Acceso a datos.*

  Independiza la aplicación de la forma de acceder a la base de datos. Permite obtener o modificar datos, con independencia del lugar en que se encuentren y cómo están almacenados. Es el único lugar donde debe haber código para acceder al repositorio de datos.

17 Virtual Object.

18 La serialización de un objeto consiste en obtener una secuencia de bytes que represente su estado.

19 Data Access Object.

Para cada clase **VO** se crean dos clases **DAO**:

- *NombreTabla*DAO.java, se trata de una clase abstracta que contiene la interfaz de definición[20].
- *NombreTabla*DAOImpl.java, implementa los métodos de la clase anterior, sobrescribiéndolos.

- **BO**[21] *Lógica de negocio*.

  Conjunto de clases con la lógica de negocio, separadas del resto mediante interfaces.

  También para cada tabla representada como objeto se definen dos clases en **BO**, *NombreTabla*BO.java y *NombreTabla*BOImpl.java.

  Estos métodos procesan los parámetros enviados desde **Rest** e invocan los métodos de las clases **DAO.**

- **Rest** *Servicios Web*.

  Contiene los métodos que son ofrecidos por el servicio Web.

- **Util**

  Módulos para seguridad, autenticación, etc.

Mediante *Hibernate* se realizan las tareas de mapear y acceder[22] a las tablas de la base de datos *kpax*, de manera que puedan ser tratadas como objetos en java.

La creación de los servicios *REST* está a cargo de *Jersey*, una implementación de la API JAX-RS [21] sobre los que establece qué parámetros intercambia, el tipo de mensajes y la clase de **BO** debe ejecutar el método solicitado.

Spring mantiene, en tiempo de ejecución, instancias de las clases **BO**, **DAO** y **VO** encargándose de la comunicación y relaciones entre ellas haciendo uso de la inyección de dependencias, esto es, colocar dentro de un objeto otros que puedan cambiar su comportamiento, sin que esto implique volver a crear el objeto.

3.1.1.3 La base de datos

Se utiliza un único gestor de bases de datos MySQL, que mantiene tanto la base de datos de *Elgg* como la propia de *k-PAX*.

Esta última – *kpax* – cuenta en la versión inicial con una docena de tablas que cubren los diferentes aspectos relacionados con usuarios, juegos y partidas mencionados en el punto 2.2.3. La Tabla 3 resume las características de la base de datos original.

20 Contiene la declaración de los métodos sin estar implementados.

21 Business Object.

22 Utiliza el lenguaje HQL.

Agrupamiento	Tabla	Descripción
Usuarios	*User*	Autenticación directa.
	Realm	Servicios externos autorizados para la autenticación remota.
	UserRealm	Vincula la información del servicio externo con el usuario local.
Grupos	*Group*	Registra los distintos grupos.
	UserGroup	Usuarios afiliados a los distintos grupos.
Juegos	*Game*	Información para describir los juegos.
	GameScore	Puntaje de un usuario en un determinado juego.
	GameLike	Permite expresar el agrado de un usuario por un juego.
Acceso a juegos	*GameAccess*	Permisos de acceso a juegos para los grupos.
Sesión	*Session*	Registra las sesiones de juego.
Partidas	*GameInstance*	Registra las instancias de juego.
	UserGameInstance	Registra a los jugadores que están jugando en una determinada partida.

3.1.2 Ambiente de desarrollo

Para el desarrollo de la actividad de manera local se cuenta con dos entornos virtuales[23] provistos por el tutor externo – uno corriendo el sistema operativo Windows XP ® y el otro con Ubuntu 12.10 – ambos con las instalaciones necesarias para el trabajo, debidamente configuradas:

- Servidor *WAMP* o *LAMP* dependiendo del sistema operativo instalado.
- Entorno de desarrollo Java, *Eclipse* [22] versión *Juno*.
- Servidor de aplicaciones *Jboss* versión 4.2.3.GA donde se despliega el núcleo *svrKpax.*
- *Apache Maven* 3.0.4.
- El software de control de versiones *git* version 1.7.10.4
- La herramienta administración de MySQL a través de páginas web *phpMyadmin*
- *Elgg* 1.8.14 instalado y en funcionamiento.
- Los *plugins* necesarios para implementar *k-PAX.*
- Las bases de datos *elgg* y *kpax* ya creadas y con algunos datos de prueba.

Al analizar las distintas máquinas se detecta que ya han sido utilizadas en otros desarrollos.

Con la finalidad de aprendizaje y fundamentalmente evitar errores durante el trabajo de implementación[24], se decide construir un nuevo ambiente sobre la última distribu-

23 Ejecutadas mediante Oracle VM *VirtualBox.*

24 Por ejemplo se han observado inconvenientes en el manejo de las mayúsculas y minúsculas entre un sistema y el otro, como también, en la implementación inicial, donde una misma tabla se encuentra dos veces, una vez con su nombre en minúsculas y la otra con el nombre respetando la convención de iniciar con mayúscula.

ción de *Ubuntu* que cuente con las mismas herramientas que las anteriores.

No obstante, una vez realizada una nueva incorporación, por seguridad se realizan pruebas de funcionamiento sobre aquellas virtualizaciones.

Asimismo, por las características del proyecto, se decidió incorporar las siguientes herramientas:

- *Meld* 1.8.4 un visor de diferencias y fusión que permite comparar archivos, directorios y realizar control de versiones de proyectos.
- *MySQL Workbench 6.0* una herramienta de diseño y visualización de bases de datos *MySQL.*
- Sobre la máquina virtual con sistema operativo privativo se instaló *Windows PowerShell*™, un shell de línea de comandos diseñado para los administradores de sistemas.

3.1.3 Estándares y normas a aplicar

La aplicación ha sido pensada para que esté disponible sobre cualquier dispositivo, debiendo entonces ser accesible tanto mediante aplicaciones de escritorio como desde la web.

Por ello su diseño cumple con las especificaciones técnicas y directrices *W3C* [23], utilizando lenguajes estándares de *frontend*[25] además de disponer que el acceso a los datos se realice mediante servicios web, con un modelo de datos fácilmente extensible para incluir futuras ampliaciones.

3.2. Usuarios del sistema

En el sistema están establecidos diferentes tipos de usuarios, de acuerdo con los privilegios otorgados.

Todos ellos, con independencia del rol asignado, podrán realizar actividades de redes sociales, jugar o administrar su propio perfil. Se establecen tres categorías de usuarios:

- **Administrador**

  Es el usuario con mayores privilegios dentro de la plataforma. Tiene control total del sistema, el acceso a datos críticos y permisos para administrar la propia plataforma.

- **Usuario común**

  Usuario del sistema que no tiene privilegios de administrador. Puede realizar las actividades normales de la red.

- **Usuario Desarrollador**

  Todo usuario común que tenga o suba algún juego a la plataforma. Este rol se concede automáticamente a los usuarios que cuenten con juegos asociados a su perfil.

25 HTML, CSS3, Javascript.

3.3. Módulos a integrar

Tal como se ha indicado, solo un subconjunto de las funcionalidades previstas en *k-PAX* están implementadas.

En la figura 3 puede observarse de la versión oficial, la página de juegos – pestaña *Games* – para un usuario administrador, cuando en la plataforma ya se ha incorporado un juego.

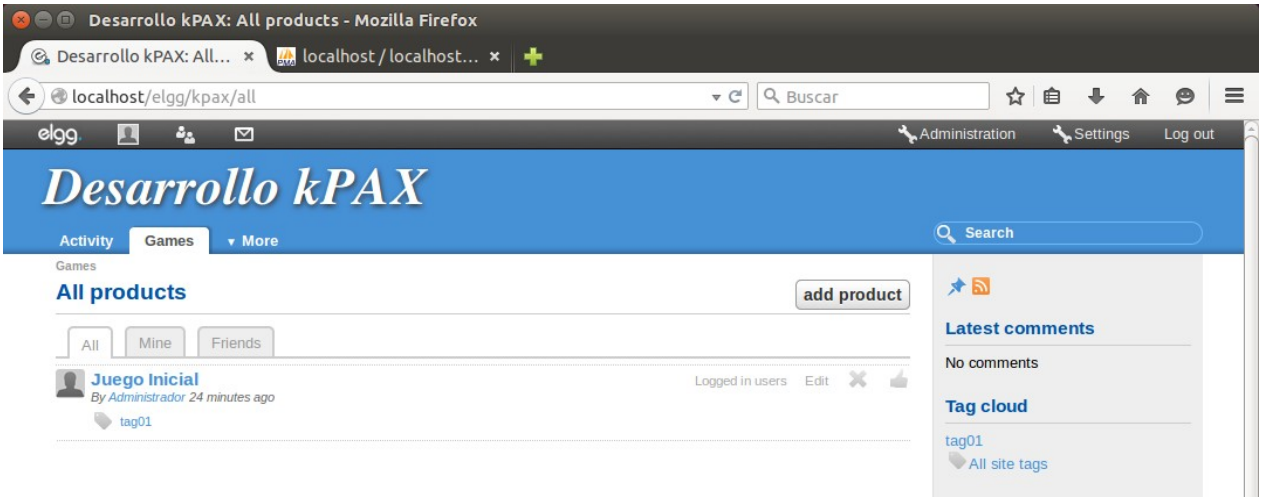

Figura 3: k-PAX. Versión oficial

Como puede apreciarse y se mostrará más adelante, la información registrada sobre el juego es escasa y visualizada con alta dependencia de las características de *Elgg*.

Luego de la integración en condiciones de funcionales de los dos módulos de terceros, estarán disponibles más prestaciones, tales como una visualización más completa y una opción de búsquedas de juegos por diferentes criterios

Para llevar a cabo la tarea, resultó necesario analizar cada uno ellos por separado y de forma independiente, determinando el estado de funcionamiento y cotejando tanto el código como la memoria descriptiva.

3.3.1 Módulo desarrolladores

Este módulo se ha realizado sobre la base de trabajos previos, por lo que su punto de partida no es la versión oficial.

Sin embargo, a efectos del presente proyecto se lo ha tomado como una unidad modificadora del código inicial, sin que ello implique desconocer los créditos de quiénes trabajaron previamente[26].

26 Las referencias a sus aportes se han mantenido en el código.

El trabajo permite que los desarrolladores de juegos puedan, además de publicarlos, añadir detalles como imágenes, un vídeo, una descripción y algunos elementos de clasificación.

Los juegos creados no estarán disponibles para los usuarios comunes hasta tanto un administrador lo autorice.

Sin embargo, el desarrollador podrá ver un listado de sus propios juegos – con independencia de la autorización – y modificar su aspecto o eliminarlos.

Todas estas acciones estarán accesibles mediante botones, disponibles o no de acuerdo con el rol del usuario.

La Figura 4, tomada desde la memoria del diseñador, muestra una captura de pantalla de la página donde pueden observarse algunas de las características mencionadas, cuando el usuario es administrador.

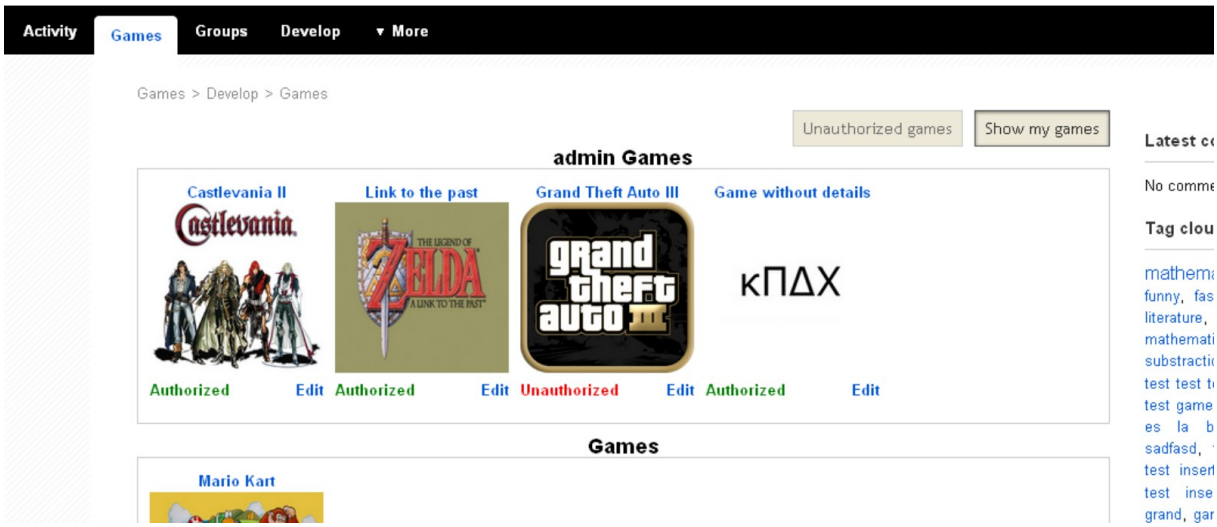

Figura 4: Pestaña Games – Módulo desarrolladores

Para lograr estas modificaciones, el autor realizó cambios sobre la plataforma:

- Alteró el *plugin kpax* para producir los cambios en la interfaz y efectuar las llamadas a los servicios.

  Básicamente cambió el listado de juegos, la página principal del juego y las acciones sobre éste, junto con el archivo de llamadas a servicios.

- Adecuó el núcleo de *k-PAX* a esas tres nuevas exigencias.

- A nivel de base de datos añadió nuevas entidades y modificó algunos campos de la tabla *Game.*

Esto obligó a las consecuentes modificaciones de las interfaces **DAO**, sus implementaciones y las clases de intercambio de información.

Concretamente el proyecto incorporó en la tabla *Game* información sobre la categoría del juego, la fecha de creación y su desarrollador. Relacionada con ésta, agregó dos nuevas tablas:

- *GameDetail* representando toda la información extra de un juego que lo detalla en alguna forma.
- *GameImage* sobre la que se almacenan las *URL* de las distintas imágenes del juego.

También se incorporan tres tablas y tres vistas adicionales como elementos necesarios para el funcionamiento y que son consecuencia de los trabajos anteriores.

La Tabla 4 presenta un resumen de esas modificaciones, resaltando el nombre de la tabla o vista que se incorpora. Solamente se indican aquellas tablas que tienen relación con el módulo.

Agrupamiento	Tabla/Vista	Descripción
Juegos	*Game*	Modificada – Información para describir los juegos.
	Category	Representa la categoría del juego
	Comment	Almacena distintos los comentarios realizados sobre un juego.
	Tag	Guarda la información de una etiqueta y el juego al que está asignada.
	GameDetail	Almacena detalles de los juegos.
	GameImage	Guarda las URL de las imágenes del juego.
	GameView	Vista de la tabla *Game* con algunos datos sobre el uso y comentarios.
	GameSimilitudeView	Vista Sobre *Game* mostrando categorías o etiquetas similares entre juegos.
	TotalGameSimilitudeView	Vista que cuenta las similitudes con un juego.

Como ya fue mencionado, el código recibido no estaba operativo, encontrándose los principales errores en la descripción de las tablas de la base de datos y en el *plugin kpax*.

Sin embargo, algunas características o no funcionan o bien no están disponibles[27] y repercuten sobre el presente proyecto. Entre ellas se mencionan:

- Al cargar o editar un juego, se solicitan datos sobre las habilidades del juego y

27 No atribuibles a su desarrollador.

la plataforma sobre la que corre, pero luego esos datos no son tratados de ninguna manera.

- Al realizar la búsqueda de juegos del usuario mediante el botón *My Games* en la pestaña *Developers* se informa de un error de listado y muestra juegos inexistentes. Claramente denota inconsistencia con la base de datos propia de *Elgg*.
- La modificación de la tabla *Category* debe hacerse por fuera de la aplicación.

3.3.2 Módulo de búsqueda

Mediante este trabajo el proyectista realizó mejoras a la opción de búsquedas preexistente[28], incorporando características tales como:

- Filtrado de acuerdo a distintos criterios, tales como etiquetas, categorías, plataforma, habilidad y metadatos.
- Determinación de semejanzas con otros juegos, basadas en sus categorías y etiquetas.
- Ordenación y paginación.

Al igual que en el caso anterior, se producen las modificaciones sin desarrollar nuevos *plugins.*

Sobre la base de datos solamente creó una tabla para almacenar los metadatos de los juegos, modificó la tabla *Game* para incorporar la dirección de la imagen del juego y también la vista *GameView* para reflejar esa nueva situación.

Evidentemente queda claro que, para el autor, existían de otros proyectos las tablas *Platform* y *Skill* como también las modificaciones en la tabla *Game* para incluir campos que permitan relacionar el juego con esas características.

En la Tabla 5 se resumen esas actuaciones.

Agrupamiento	Tabla/Vista	Descripción
Juegos	*Game*	Modificada para incluir los nuevos datos.
	MetaData	Propia del autor – Almacena nombre y valor de los metadatos asociados a un juego.
	Platform	Almacena nombre e identificador de las distintas plataformas.
	Skill	Guarda la información sobre las destrezas.
	GameView	Modificada – Vista sobre la tabla *Game.*

La Figura 5 muestra la pantalla de listado de juegos luego de incorporar el módulo.

28 Menciona taxativamente los trabajos de Joan Farrerons i Herrero, Albert Giró Oriol y Rubén Viguera Marañón.

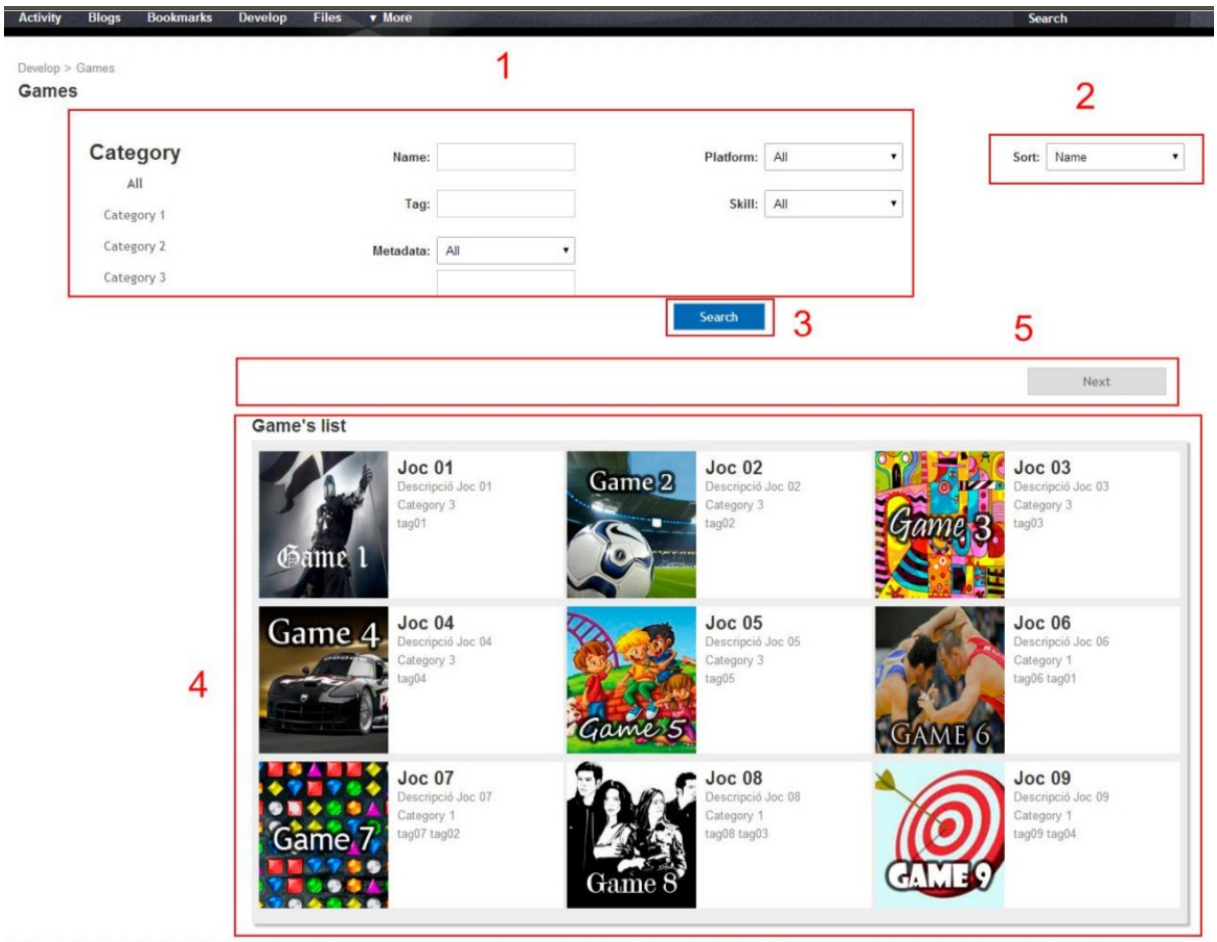

Figura 5: Pantalla de listado de juegos – Módulo de búsqueda

Corresponde mencionar que la misma se ha tomado directamente del trabajo del autor, quién ha marcado sobre ella cinco secciones que representan:

1. Criterios de filtrado.
2. Criterios de ordenación.
3. Botón de búsqueda según lo establecido en 1.
4. Listado de los primeros nueve juegos.
5. Módulo de paginación.

También destaca en la memoria que la acción de añadir juegos a la plataforma no funciona, por lo que se debe realizar la carga manual de éstos.

3.4. Establecimiento de requisitos

Teniendo en cuenta el marco del proyecto y los requerimientos establecidos en el punto 2.3, para cada módulo a incorporar, deberán ser tenidos en cuenta sus propios requisitos, de acuerdo a su documentación. Luego de la integración, éstos deben ser verificados.

3.5. Especificación del plan de pruebas

Al tiempo que se incorporen nuevas funcionalidades se realizan pruebas:

- *Unitarias*, validando cada una de las nuevas funcionalidades.
- *De integración*, para comprobar que no existen interferencias.

Dado que se trata de un producto en funcionamiento, las pruebas de integración involucran a la totalidad del sistema de información. Al llevarlas a cabo se está probando al sistema en su conjunto.

4. Diseño del sistema

Para planificar el desarrollo del sistema fue necesario realizar instalaciones previas, tanto del la versión inicial como de cada uno de los módulos, solucionando los inconvenientes que se presentaron en cada situación de manera particular[29].

El diseño está pensado de manera que, sobre la base de la versión estable del producto, se incorporen los módulos de forma iterativa e incremental hasta lograr la integración completa al sistema.

4.1. Arquitectura inicial

La Figura 6 esquematiza la interconexión entre *Elgg* y el núcleo de *k-PAX*, a través del *plugin kpax*.

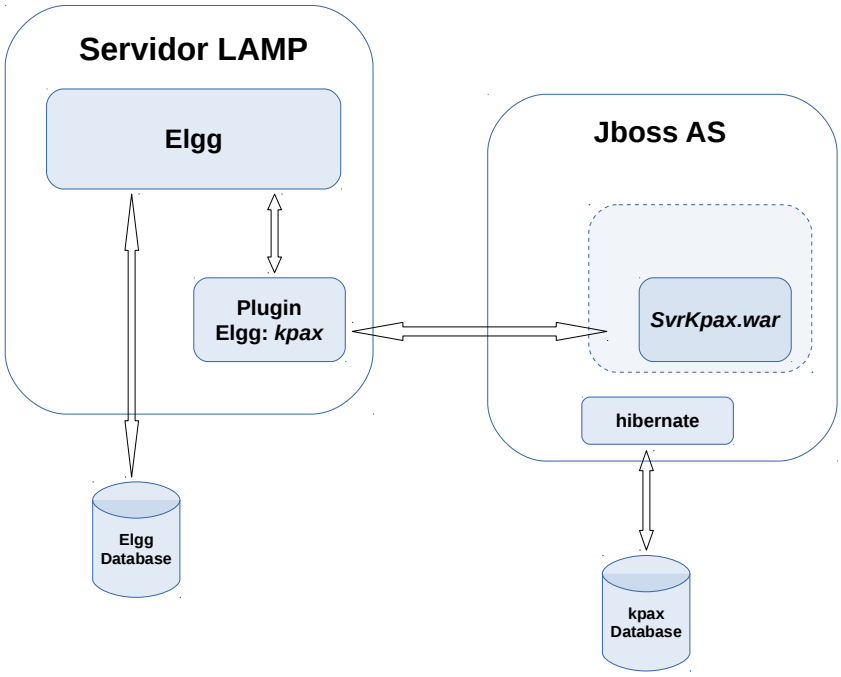

Figura 6: Interconexión entre Elgg y el núcleo k-PAX

Pensando fundamentalmente en la escalabilidad y la portabilidad, la plataforma *k-PAX* ha sido desarrollada respetando una arquitectura multicapa, presentando tres bien definidas, donde cada una solamente tiene relación con la(s) vecina(s) y cumple una determinada función específica.

29 En el capítulo 5 se detallan esas actividades.

- **Capa de presentación**

  Encargada de la interacción entre el usuario y a plataforma, conformada por:

  - La plataforma *Elgg.*
  - La interfaz gráfica de usuario, implementada por módulos o *plugins* de *Elgg*.

- **Capa de lógica de negocio**

  Donde se establecen todas las reglas que deben cumplirse. Recibe las peticiones del usuario desde la capa de presentación, efectúa el procesamiento de la información interactuando con la capa de persistencia y envía las respuestas nuevamente a la capa de presentación. Compuesta por:

  - Las Interfaces **BO** entre la capa de presentación y de persistencia, junto con sus correspondientes implementaciones.
  - Los Servicios web **Rest**, para ser accedidos desde la capa de presentación por *Elgg*, mediante algún *plugin.*
  - El paquete **Util**.

- **Capa de persistencia**

  Establece la conexión con la base de datos, realizando operaciones *CRUD.* Implementada mediante:

  - El servidor MySQL.
  - La(s) bases de datos.
  - Las clases de intercambio de información **VO.**
  - Las interfaces **DAO** de acceso a los datos.

Tanto la capa de presentación y lógica de negocio utilizan el patrón de diseño *MVC* para mostrar y controlar las interacciones con el usuario.

4.1.1 Plugin kpax

k-PAX implementa un tipo nuevo de objeto en *Elgg*, el juego. El objeto juego se deriva del objeto blog y se da de alta en la base de datos *Elgg* como nueva entidad.

De los cuatro *plugins* básicos descritos en el punto 2.2.2, *kpax* es el encargado de facilitar que desde *Elgg* pueda realizarse la gestión de juegos, mediante llamadas para comunicarse con el núcleo de servicios de *k-PAX*.

En la tabla 6 se muestran archivos relevantes de ese módulo, algunos de ellos serán modificados al momento de la integración.

Subdirectorio	Archivo	Descripción
actions/kpax	delete.php	Borrar el juego.
	save.php	Guardar el juego.
lib	Oauth.php	Servicio de autenticación.
	kpax.php	Biblioteca auxiliar.
	kpaxOauth.php	Servicio de autenticación[30].
	kpaxSrv.php	Funciones con las llamadas al servicio Web.
pages/kpax	add.php	Formulario para incorporar el juego.
	all.php	Formulario de búsqueda y listado de juegos.
	edit.php	Formulario para editar el juego.
	friend.php	Formulario que permite mostrar los juegos añadidos por los amigos del usuario.
	owner.php	Formulario para mostrar los juegos añadidos por el usuario.
	view.php	Formulario para mostrar información del juego.
views/default	forms/kpax	Contiene archivos con las vistas de las páginas.

4.1.2 Servicio SvrKpax

Como ya se dijo, el núcleo de *k-PAX* agrupa las clases en cinco paquetes.

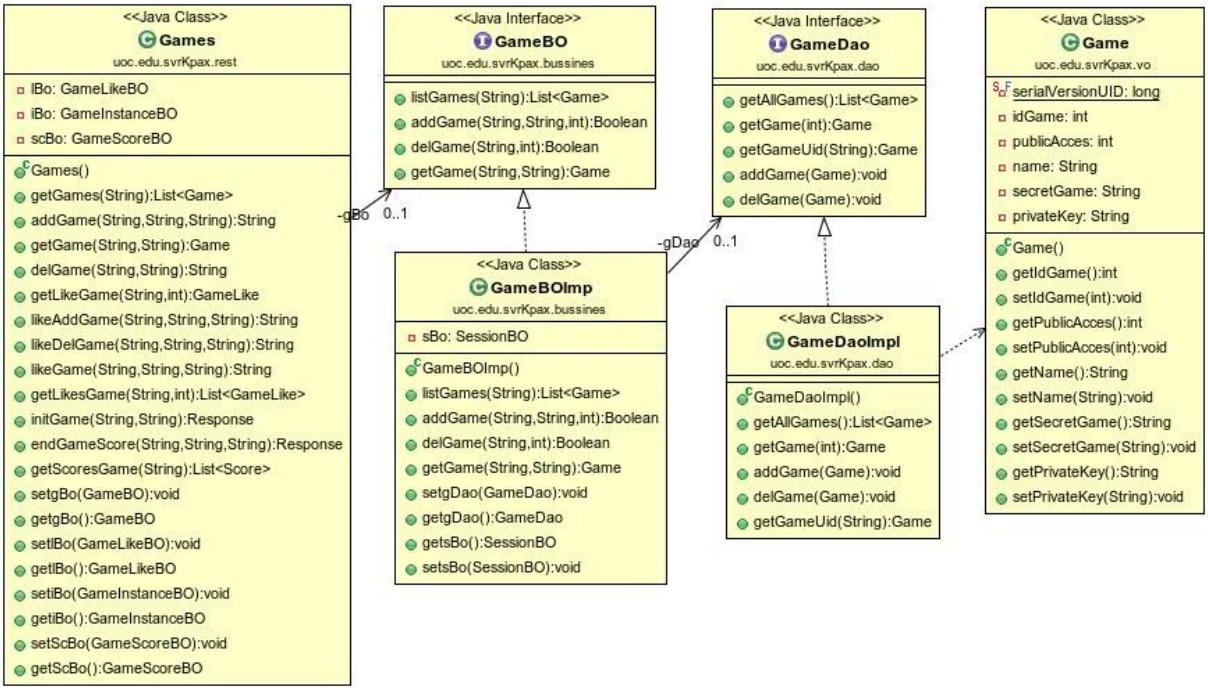

Figura 7: Diagrama de clases

La Figura 7 muestra dichas clases e interfaces, partiendo desde la clase *Games* del paquete **Rest.**

30 El *plugin* posee un sistema de autenticación y validación que no será analizado en este trabajo.

Debido a la naturaleza del proyecto, en el dibujo solamente se detallan aquellas clases que tienen relación con las funcionalidades a incorporar o modificar.

La clase *Games* contiene los métodos publicados por el servicio y que serán accedidos desde *Elgg.* Está asociada con otras, pero para el caso solamente interesa la asociación unidireccional con la clase *GameBO* – interface y clase implementadora – en **Business**.

Esta última interactúa, ya a nivel de persistencia, con la clase *GameDao* que a la vez lo hace con *GameVo*, para acceder a la base de datos kpax.

4.1.3 Base de datos kpax

El esquema inicial de la base – apartado 3.1.1.3 – se representa en la Figura 8.

Figura 8: Diseño inicial de la base de datos kpax

4.2. Integración de módulos

La integración de los módulos requiere modificar el sistema en las tres capas de la arquitectura. Tal lo mencionado, el proceso será incremental hasta obtener la funcionalidad completa e involucra al *plugin kpax*, al núcleo *SvrKpax* y a la base de datos *kpax*. Se describen a continuación esos cambios.

4.2.1 Modificaciones sobre el plugin kpax

Para unificar ambos trabajos, se ha decidido que la mejor forma de integrarlos visualmente consiste en incorporar nuevo botón a partir de la pestaña *Games* para que despliegue una página de búsqueda.

Al incorporar el primer módulo, esa página permite visualizar los juegos de terceros autorizados. En la parte superior muestra un botón que al presionarlo, despliega además los juegos del propio usuario, autorizados o no. Si la página es accedida por un administrador, otro botón permite ver y actuar sobre los juegos no autorizados.

Para este proyecto se incorpora un segundo – tercero, según el caso – botón, que al ser pulsado lleva la página de búsqueda. En el esquema de la Figura 9 se observa tal modificación, para el caso de un usuario administrador.

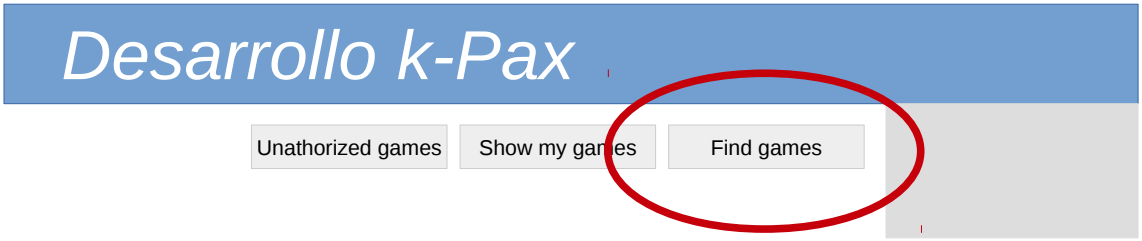

Figura 9: Botón de acceso a la búsqueda de juegos

Para lograr los objetivos se requiere de importantes modificaciones sobre el módulo que implican cambios en algunos archivos y otros que deben incorporarse.

La Tabla 7 describe las principales acciones a realizar, de manera sucesiva, sobre los diferentes archivos del conector. La misma debe interpretarse de la siguiente manera:

La primer columna hace referencia a archivos que tiene y tendrá el módulo. La segunda, a las acciones necesarias para incorporar el primer trabajo y la tercera, las actuaciones al momento de agregar el segundo.

Los nombres de archivo en negrita indican que deben incorporarse al código original, mientras que los que están en cursiva sufrirán modificaciones.

Archivo	Desarrolladores	Búsqueda
kpax		
actions		
kpax		
save.php	Modificar	Modificar
graphics		
banner.png	Incorporar	– – –
logo.php	Incorporar	– – –
languages		
ca.php	Modificar	Modificar
en.php	Modificar	Modificar
es.php	Modificar	Modificar
lib		
kpax.php	Modificar	Modificar
kpaxSrv.php	Modificar	Modificar
pages		
kpax		
add.php	Modificar	Sin Modificar
all.php	Modificar	Modificar
find.php	– – –	Incorporar
friends.php	Modificar	Sin Modificar
my_dev_games.php	Incorporar	Modificar
owner.php	Modificar	Modificar
play.php	Incorporar	Modificar
view.php	Modificar	Modificar
views		
default		
css		
elements		
game.css	– – –	Incorporar
gamelist.css	– – –	Incorporar
forms		
kpax		
save.php	Modificar	Modificar
kpax		
devs_explanations.php	Incorporar	Sin Modificar
find_games_list	– – –	Incorporar
games_list.php	Incorporar	Modificar
my_devs_games_list.php	Incorporar	Modificar
score.php	Sin Modificar	Sin Modificar
sidebar.php	Modificar	– – –
object		
kpax.php	Modificar	Modificar
rss		
object		
kpax.php	Sin Modificar	Sin Modificar
manifest.xml	Modificar	Modificar
start.php	Modificar	Modificar

4.2.1 Modificaciones al servicio SvrKpax

El proceso requiere incorporar nuevas clases y modificar algunas existentes.

De manera similar al caso anterior, en la Tabla 8 se refleja la situación con la diferencia que aquí se muestran todos los archivos.

Clase	Desarrolladores	Búsqueda
business		
CategoryBO.java	Incorporar	Sin modificar
CategoryBOImp.java	Incorporar	Sin modificar
CommentBO.java	Incorporar	Sin modificar
CommentBOImp.java	Incorporar	Sin modificar
GameBO.java	Modificar	Modificar
GameBOImp.java	Modificar	Modificar
GameDetailBO.java	Incorporar	– – –
GameDetailBOImp.java	Incorporar	– – –
GameImageBO.java	Incorporar	– – –
GameImageBOImp.java	Incorporar	– – –
GameInstanceBO.java	Sin modificar	Sin modificar
GameInstanceBOImp.java	Sin modificar	Sin modificar
GameLikeBO.java	Sin modificar	Sin modificar
GameLikeBOImp.java	Sin modificar	Sin modificar
GameScoreBO.java	Sin modificar	Sin modificar
GameScoreBOImp.java	Sin modificar	Sin modificar
MetaDataBO.java	– – –	Incorporar
MetaDataBOImp.java	– – –	Incorporar
PlatformBO.java	– – –	Incorporar
PlatformBOImp.java	– – –86	Incorporar
SessionBO.java	Sin modificar	Sin modificar
SessionBOImp.java	Sin modificar	Sin modificar
SkillBO.java	– – –	Incorporar
SkillBOImp.java	– – –	Incorporar
TagBO.java	Incorporar	Sin modificar
TagBOImp.java	Incorporar	Modificar
UserBOImp.java	Sin modificar	Sin modificar
UserBO.java	Sin modificar	Sin modificar
dao		
CategoryDao.java	Incorporar	Sin modificar
CategoryDaoImpl.java	Incorporar	Sin modificar
CommentDao.java	Incorporar	Sin modificar
CommentDaoImpl.java	Incorporar	Sin modificar
GameDao.java	Modificar	Modificar
GameDaoImpl.java	Modificar	Modificar
GameDetailDao.java	Incorporar	– – –
GameDetailDaoImpl.java	Incorporar	– – –
GameImageDao.java	Incorporar	– – –
GameImageDaoImpl.java	Incorporar	– – –
GameInstanceDao.java	Sin modificar	Sin modificar
GameInstanceDaoImpl.java	Sin modificar	Sin modificar

Clase	Desarrolladores	Búsqueda
GameLikeDao.java	Sin modificar	Sin modificar
GameLikeDaoImpl.java	Sin modificar	Sin modificar
GameScoreDao.java	Sin modificar	Sin modificar
GameScoreDaoImpl.java	Sin modificar	Sin modificar
GameViewDao.java	Incorporar	Sin modificar
GameViewDaoImpl.java	Incorporar	Modificar
MetaDataDao.java	– – –	Incorporar
MetaDataDaoImp.java	– – –	Incorporar
PlatformDao.java	– – –	Incorporar
PlatformDaoImp.java	– – –	Incorporar
RealmDao.java	Sin modificar	Sin modificar
RealmDaoImpl.java	Sin modificar	Sin modificar
SessionDao.java	Sin modificar	Sin modificar
SessionDaoImpl.java	Sin modificar	Sin modificar
SkillDao.java	– – –	Incorporar
SkillDaoImp.java	– – –	Incorporar
TagDao.java	Incorporar	Sin modificar
TagDaoImpl.java	Incorporar	Modificar
UserDao.java	Sin modificar	Sin modificar
UserDaoImp.java	Sin modificar	Sin modificar
rest		
Games.java	Modificar	Modificar
Jsonp.java	Modificar	Modificar
User.java	Sin modificar	Sin modificar
util		
AES.java	Sin modificar	Sin modificar
CosntantsKPAX.java	Sin modificar	Sin modificar
IntegerWrapper.java	– – –	Incorporar
Oauth.java	Sin modificar	Sin modificar
Security.java	Sin modificar	Sin modificar
vo		
Category.java	Incorporar	Sin modificar
Comment.java	Incorporar	Sin modificar
Game.java	Modificar	Modificar
GameDetail.java	Incorporar	– – –
GameImage.java	Incorporar	– – –
GameInstance.java	Sin modificar	Sin modificar
GameLike.java	Sin modificar	Sin modificar
GamePagination.java	– – –	Incorporar
GameScore.java	Sin modificar	Sin modificar
GameView.java	Incorporar	Modificar
MetaData.java	– – –	Incorporar
Platform.java	– – –	Incorporar
Realm.java	Sin modificar	Modificar
Score.java	Sin modificar	Sin modificar
Session.java	Sin modificar	Sin modificar
Skill.java	– – –	Incorporar
Tag.java	Incorporar	Modificar
TotalGameSimilitudeView.java	Incorporar	Sin modificar
User.java	Sin modificar	Sin modificar

4.2.3 Modificaciones en la base de datos

La persistencia es el punto crucial de la aplicación. La modificación de la base de datos representa el eje central del desarrollo porque a partir de ellas se diseñan los objetos que intervienen en el servicio.

Para almacenar las nuevas características incorporadas a los juegos, se adoptaron criterios diferentes.

En el primero de los módulos – desarrolladores – se optó por incorporar los detalles y las imágenes del juego en tablas separadas. Se incorporó información de la categoría del juego desde una tabla externa[31]. También se agregaron relaciones para comentarios y etiquetas.

Se muestra en la Figura 10 esa evolución, partiendo desde la tabla *Game* en su versión inicial, representada a la izquierda de la imagen en celeste, hasta alcanzar su diseño final, indicado en color verde.

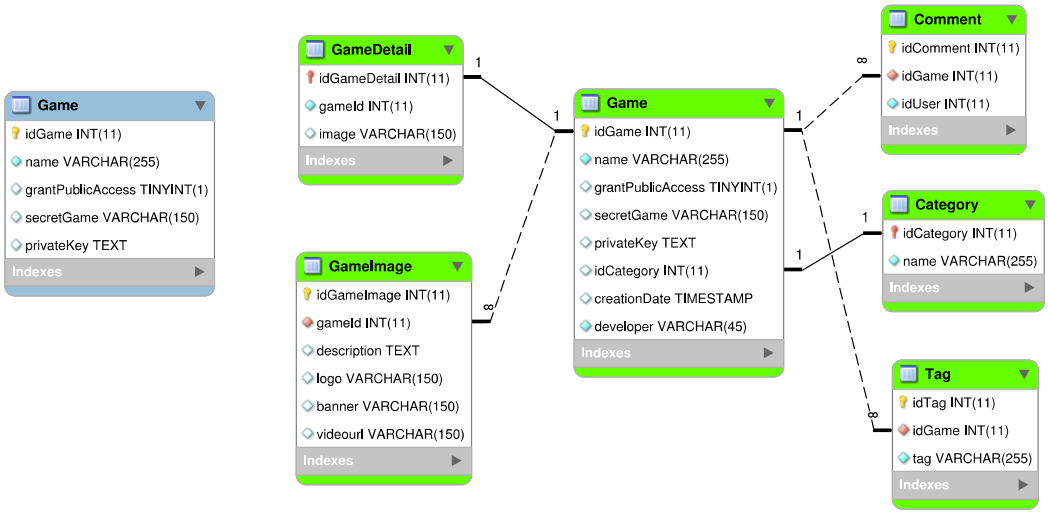

Figura 10: Modulo desarrolladores – Evolución información de los juegos.

En el otro módulo se optó por una solución diferente. El desarrollador compatibilizó los cambios que realizaran en trabajos previos Albert Giró[32] y Rubén Viguera[33], incorporando de su autoría la tabla *MetaData.*

Como en el caso anterior, la Figura 11 muestra la misma comparación, ahora indicando en color rojo el diseño utilizado en el módulo de búsqueda.

31 Previo a su desarrollo, posiblemente de los trabajos de Albert Giró y Rubén Viguera.

32 Agrega las tablas *Category*, *Skill*, *Platform* y sus referencias en *Game*.

33 Incorporó las tablas *Tag* y *Comments.*

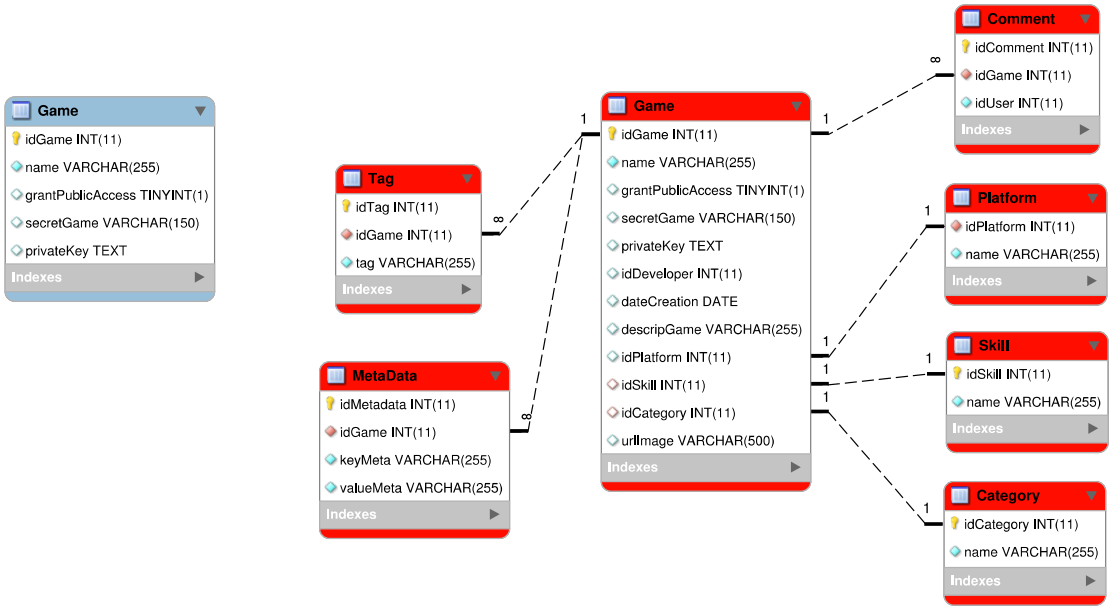

Figura 11: Modulo de búsqueda – Evolución información de los juegos.

Comparando las figuras previas puede notarse que es necesario el rediseño de la base de datos para incorporar las modificaciones. La Figura 12 contrapone la tabla *Game* del diseño definitivo – en azul – con las anteriores.

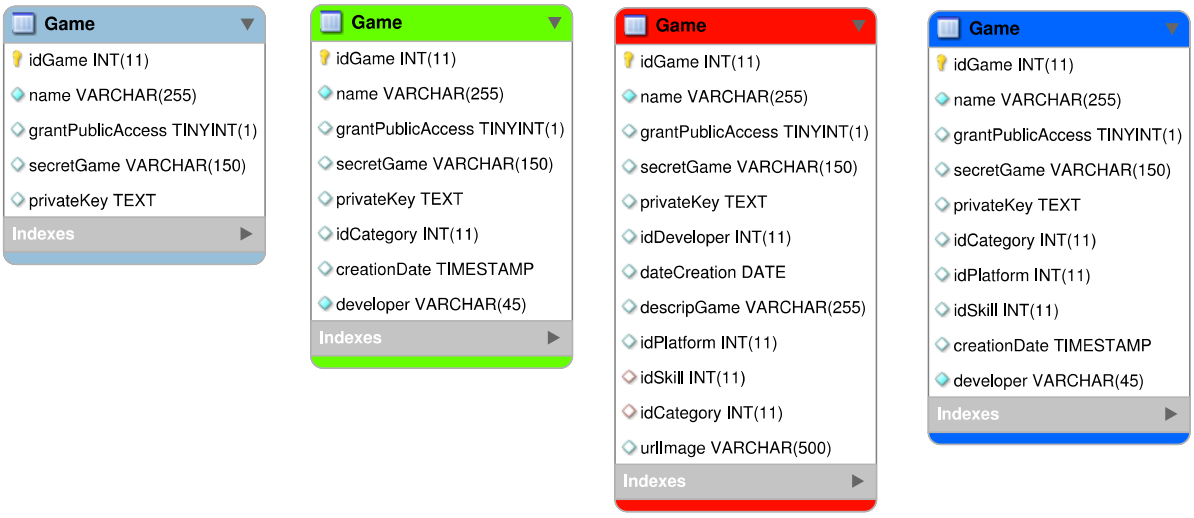

Figura 12: Evolución de la tabla Game.

Finalmente, en la Figura 13 se puede observar el esquema definitivo de las tablas del proyecto, modificadas según lo expresado, manteniendo la codificación de colores.

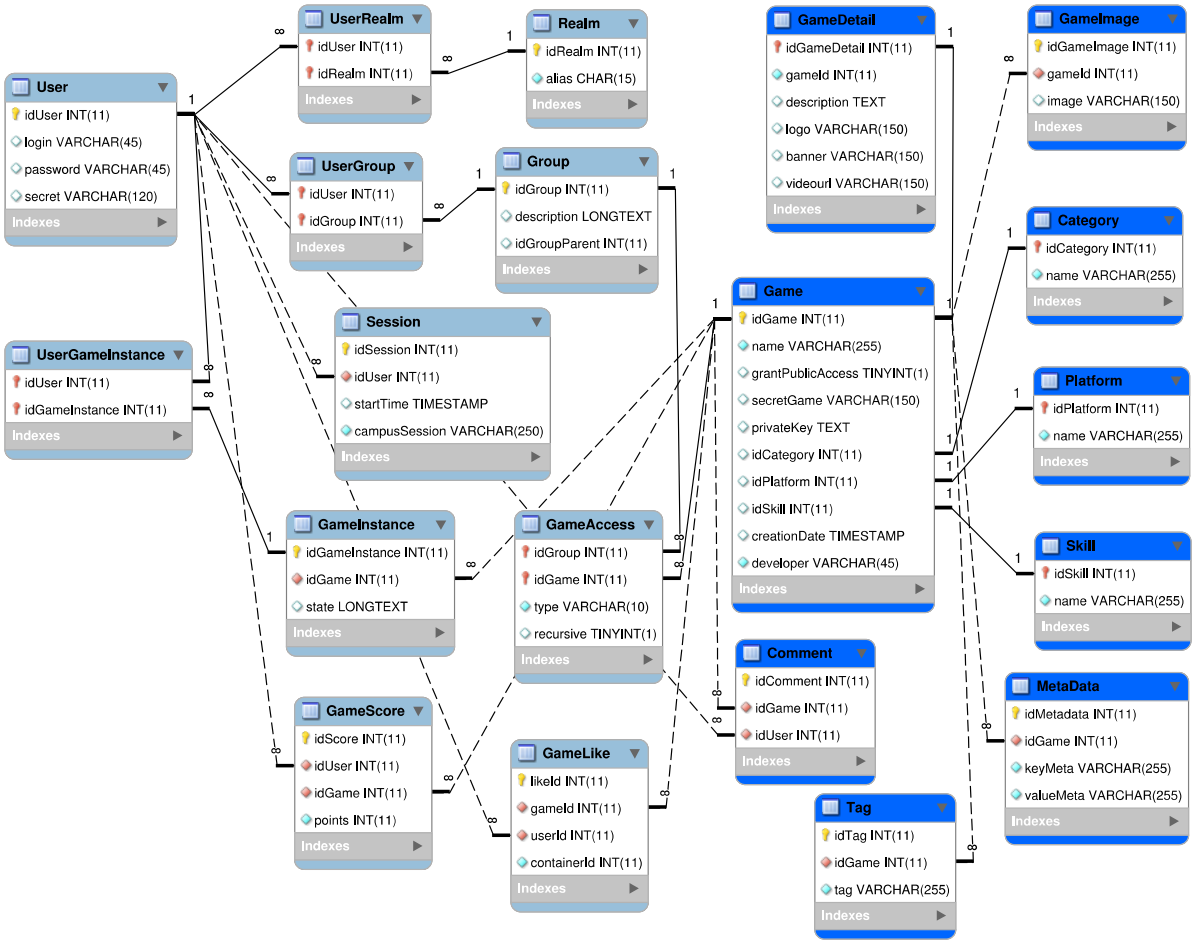

Figura 13: Diseño definitivo de la base de datos kpax

5. Implementación.

Tal lo consensuado con la cátedra, las tareas de puesta en marcha e implementación del primer módulo se realizaron en equipo, en un principio trabajando de manera casi conjunta y alcanzando mayor grado de independencia a medida que avanzaba el proyecto.

Ya para la incorporación de la segunda funcionalidad, el trabajo se realizó totalmente de manera individual.

5.1. Planificación de las actividades de desarrollo e integración de sistema

En concordancia con lo detallado en el punto 2.1.2, la metodología ágil elegida para llevar a cabo el trabajo se basa en el desarrollo iterativo e incremental, donde tanto los requisitos como las soluciones evolucionan a lo largo del proceso de acuerdo con lo aprendido a lo largo del ciclo de desarrollo anterior.

Se parte, entonces desde el código de base, incorporando sucesivamente a cada uno de los módulos y realizando – en los ciclos sucesivos – los cambios necesarios que mejoren las funcionalidades y capacidades del sistema. Así, el código irá evolucionando hasta que la aplicación se considere completa.

5.2. Desarrollo

Independientemente de la metodología de trabajo empleada, el desarrollo puede describirse contemplando tres fases principales:

- La puesta en marcha de la versión inicial.
- La reparación e incorporación del módulo desarrolladores
- La incorporación del módulo de búsqueda.

5.2.1 Versión inicial

El trabajo inicia con la puesta la puesta en servicio de la versión estable de *k-PAX* [17] [18].

La instalación se realizó con éxito sobre las distintas plataformas, donde pudo observarse una situación altamente indeseable respecto de *MySQL,* debido a la diferencia de sensibilidad a las mayúsculas/minúsculas entre los sistemas operativos GNU/Linux y Windows®.

La máquina virtual privativa presentaba la dificultad que, al instalar las tablas, los nombres de las mismas eran automáticamente incorporados en minúsculas. Por lo tanto, al migrarlas luego a un sistema GNU/Linux, éstas no podrán ser accedidas si no existe correspondencia entre las capitalidades en los nombres de tablas y el código de acceso.

Para mitigar la situación[34], luego de adecuar la denominación de tablas, se decidió modificar la configuración del gestor de bases de datos en la máquina con Windows® incorporando en el archivo *my.ini* la variable del sistema:

```
lower_case_table_names=2
```

Ya en servicio y realizadas las pruebas correspondientes, se verificó que el sistema funciona dentro de los parámetros previstos, esto es, se almacena información muy básica sobre el juego.

La figura 14 resume la acción de la carga de un juego al sistema y la persistencia de los mismos. Puede notarse claramente que el proceso es altamente dependiente de *Elgg* y que la información almacenada en la base de datos *kpax* es bastante escasa.

Figura 14: Carga de un juego en la versión estable de k-PAX

No se observaron inconvenientes respecto del núcleo de la aplicación, pero con relación a la base de datos, se pudo constatar la existencia de una tabla duplicada[35] y cuya única diferencia era el nombre, una vez escrito con la primera letra en mayúsculas y otra totalmente en minúsculas.

Como muy probablemente sea una consecuencia de la situación de compatibilidad ya mencionada, la solución consistió en eliminar la segunda tabla.

34 Lamentablemente la modificación no resuelve el problema sobre las vistas.

35 *Session*

Por otra parte, durante el análisis de *logs* de *Apache* se observaron errores:

```
[Wed Apr 15 08:07:05.151295 2015] [:error] [pid 2603] [client 127.0.0.1:55646] Class 'ElggKpax' was not found, missing plugin?, referer: http://localhost/elgg/kpax/add/54

[Wed Apr 15 08:07:05.665940 2015] [:error] [pid 1186] [client 127.0.0.1:55648] DEBUG: 2015-04-15 08:07:05 (ART): "Undefined index: HTTP_X_ELGG_APIKEY" in file /home/ubuntu/www/elgg/engine/lib/web_services.php (line 556)
```

Respecto del primero, desde el archivo *start.php* del conector *kpax* se hace referencia a dicha clase, por lo tanto se la crea siguiendo el modelo desde donde deriva el objeto. Esto implicó la incorporación de un nuevo archivo – *ElggKpax.php* – dentro de un subdirectorio *classes*.

Con referencia al segundo, el análisis de la literatura permitió detectar que se trata de una situación propia de la versión 1.8.x de *Elgg.* Teniendo en cuenta que al momento está ya disponible la versión 1.11, realmente intentar corregirlo no tiene sentido, pero fundamentalmente porque implicaría modificar su núcleo[36].

Otra corrección, irrelevante si se quiere, consistió en adecuar la ortografía en el nombre del directorio *business*. Este detalle implicó un recorrido exhaustivo del código de base como de los módulos a incorporar.

La instalación de esta versión permitió adquirir conocimiento, tanto a nivel de programación como sobre las tecnologías empleadas, y dimensionar adecuadamente la magnitud del proyecto. Particularmente resultó importante la visión que se obtuvo respecto del funcionamiento interno de la plataforma *Elgg*.

5.2.2 Integración del módulo desarrolladores

Una vez en servicio la versión de partida, se procedió a incorporar el módulo desarrolladores, tarea que puede considerase realizada en dos grandes etapas: la puesta en marcha del complemento y su integración al proyecto actual.

5.2.2.1 Reparación

Tal como fue comentado, la única copia del producto con que se contaba no estaba totalmente operativa, debiendo investigarse y corregirse previamente la situación.

También fue mencionado que el desarrollador inició su trabajo desde versiones ya modificadas del código y si bien se lo consideró como una unidad, hubo que deconstruir el proceso.

Esto trajo aparejada una importante demora sobre los tiempos planificados y como contrapartida, resultó de suma utilidad para comprender en profundidad la arquitectura de la aplicación.

La base de datos *kpax*

Se detectaron anomalías sobre el archivo de volcado de las tablas de la base de datos que impedían que éstas pudieran reconstruirse.

36 A partir de la versión 1.9 esa función se quita del núcleo y se implementa como un módulo.

Durante el análisis del código se encontraron errores de correspondencia entre esa copia de seguridad y la capa de persistencia del núcleo. Fue necesario realizar un minucioso trabajo para recomponerlas en condiciones de funcionalidad.

También se determinó que, para funcionar, la tabla *Category* debía contener datos y que éstos debían incorporarse manualmente.

Se limitó la cantidad de tablas incorporadas a las estrictamente necesarias para garantizar el funcionamiento.

El core

Sobre el núcleo fue necesario modificar el método *getUserGames(String username)* en la implementación de la clase *GameDao,* adecuando la capitalización en los nombres de tablas, pudiéndose así obtener correctamente los listados.

Una situación similar ocurrió con las clases de intercambio de información *GameImage* y *GameDetail* donde la referencia a las tablas estaba en minúsculas.

El plugin kpax

El código del conector sufrió varias adecuaciones:

- El archivo *manifest.xml* había sido editado de manera incorrecta e impedía que el *plugin* pudiera ser activado desde *Elgg*.
- Al intentar guardar un juego se generaba una condición de error, visualizada a través de los *logs* de *Apache:*

  ```
  [Fri Apr 24 10:18:03 2015] [error] [client 127.0.0.1] PHP Fatal error:  Call to undefined function string_to_url_array() in E:\\wamp\\www\\elgg-1.8.14\\mod\\kpax\\actions\\kpax\\save.php on line 13, referer: http://localhost/elgg-1.8.14/kpax/add
  ```

  Fue necesario recrear la imprescindible función.
- El autor original incorporó un par de imágenes para utilizar por defecto cuando el juego no cuenta con gráficos para el logo o para la pancarta. Se ajustó el uso de las mayúsculas en los nombres de esos archivos[37].
- Una misma vista – *games_list.php* – se utiliza como plantilla tanto para el listado de los juegos y como para la visualización en detalle de éstos. Ha sido proyectada para mostrar las imágenes en una galería mediante *Fotorama* [24], un plugin que utiliza la biblioteca de JavaScript *jQuery* [25] para esa función. En producción, ese conector se instalaría directamente sobre el servidor Web pero en este caso se lo invoca desde un sitio remoto y cuyos enlaces estaban rotos.

Efectuadas esas correcciones, el sistema estuvo operativo en las condiciones indicadas en la memoria del proyecto.

La Figura 15 muestra el formulario de carga de un juego mientras la aplicación está corriendo sobre una máquina virtual privativa, dividido por razones de espacio, y la captura de pantalla en la Figura 16, tomada al momento de pulsar el botón *Unauthorized*, permite observar que el juego no está disponible para otros usuarios hasta tanto no sea debidamente autorizado[38].

37 Ubicados en el subdirectorio *graphics*.

38 En este caso, el desarrollador es el propio administrador, por lo tanto tiene la opción de autorizarlo.

Activity Games Groups Develop ▾ More

Games > Develop > Games > Add game

Add game

Game name (*)

Juego Inicial Desarrolladores

Description

Juego de prueba, corriendo en la VM privativa incluyendo el módulo de

Word count: 12

Logo (square image url 150px x 150px)

http://localhost/images/game01.jpg

Game banner (image url)

http://localhost/images/banner-01.jpeg

Youtube id video

hjTOaZ2Q6lc

Game category/ies

1

Creation date

24-04-2015 11:09:23

Available platforms

- [x] web
- [ ] android
- [ ] iOS
- [ ] Nintendo DS
- [ ] PSP
- [ ] Nintendo Wii
- [ ] XBox

Skills (*)

1

Related tags

tag01

Images

Insert as many images as you want, put the URLs separated by a spa

http://localhost/images/game01-1.jpg http://localhost/images/game0

Word count: 2

Security

In order to connect your game to kPAX, it is necessary to create a specific public/p
openSSL. You only need to run the command:

```
openssl req -out requestUser.csr -new -newkey rsa:2048 -nodes -keyou
```

Once you have run it, you should have two text files: the private key (.pml), which i
request file (.csr), which you must include below. kPAX will create and store a certi

Certificate request file [.csr] (*)

Examinar... requestUser.csr

Send my game!

Report this

Figura 15: Formulario para la carga de un juego

kPAX

Activity Games Groups Develop ▾ More

Games > Develop > Games

Unauthorized games Show my games

Unauthorized Games

Juego Inicial Desarrolladores

Authorize Edit

Authorize

Figura 16: Vista de juegos no autorizados

Una vez almacenado el juego, desde MySQL se comprueba que las tablas contengan los valores ingresados, mostrándose el resultado en la Figura 17[39].

```
mysql> use kpax
Database changed
mysql> select * from Game;
+--------+-------------------------------+-------------------+-----------------
| idGame | name                          | grantPublicAccess | secretGame
+--------+-------------------------------+-------------------+-----------------
|    246 | Juego Inicial Desarrolladores |                 0 | GAME.78abb73
+--------+-------------------------------+-------------------+-----------------
1 row in set (0.00 sec)

mysql> mysql> select * from GameDetail;
+--------------+--------+-----------------------------------------------------
| idGameDetail | gameId | description
+--------------+--------+-----------------------------------------------------
|           34 |    246 |<p>Juego de prueba, corriendo en la VM privativa i
+--------------+--------+-----------------------------------------------------
3 rows in set (0.00 sec)

mysql> select * from GameImage;
+-------------+--------+-------------------------------------+
| idGameImage | gameId | image                               |
+-------------+--------+-------------------------------------+
|          16 |    246 | http://localhost/images/game01-1.jpg |
|          17 |    246 | http://localhost/images/game01-2.jpg |
|          18 |    246 | http://localhost/images/game01-3.jpg |
|          19 |    246 | http://localhost/images/game01-4.jpg |
|          20 |    246 | http://localhost/images/game01_5.jpg |
+-------------+--------+-------------------------------------+
5 rows in set (0.00 sec)

mysql> select * from Tag;
+-------+--------+-------+
| idTag | idGame | tag   |
+-------+--------+-------+
|    28 |    246 | tag01 |
+-------+--------+-------+
1 row in set (0.00 sec)
```

```
--+-----------------------------------------+-----------+---------------------+----------+
  | privateKey                              | idCategory |  creationDate       | developer |
--+-----------------------------------------+-----------+---------------------+----------+
9 | 8d19e7eac174d06d3e0058c38b8c316cae184cee |          1 |  2015-04-24 11:09:23 | admin     |
--+-----------------------------------------+-----------+---------------------+----------+

-+-----------------------------------+-----------------------------------+-----------+
 | logo                              | banner                            | videourl  |
-+-----------------------------------+-----------------------------------+-----------+
 | http://localhost/images/game01.jpg | http://localhost/images/banner-01.jpeg| hjTOaZ2Q6Ic |
-+-----------------------------------+-----------------------------------+-----------+
```

Figura 17: Consultas en MySQL

La Figura 18 contiene una captura de pantalla de la página principal del mismo juego[40] correspondiéndose con el diseño original. Nótese que en la sección *otros juegos del mismo desarrollador* se visualiza el propio juego.

5.2.2.2 Incorporación

En primer lugar, las clases e implementaciones en el paquete de lógica de negocio debieron adaptarse al nombre – corregido – del subdirectorio *business,* así como cualquier referencia directa a ellas.

Más allá de este detalle, la integración del módulo ya reparado no presentó mayores inconvenientes. Sin embargo, se llevaron a cabo algunas modificaciones.

La base de datos y el núcleo

Pudo constatarse que durante la carga de un juego se solicitaban datos tales como habilidades o plataformas, pero que esos valores no se registran.

Tomando como base el procedimiento realizado con la clase *Category* y la solución implementada en el trabajo de Víctor Corral Blanch, se incorporaron las tablas y clases *Skill* y *Platform* junto con las funcionalidades para operar con ellas.

También *Game* debió modificarse para registrar los identificadores de esos atributos, como también las clases *Games* y *Jsonp* en los servicios **Rest** para permitir la persistencia de los mismos.

No obstante y por cuestión de tiempo, no se desarrolla el código para la gestión de al-

39 Algunos campos fueron recortados para facilitar la visualización.

40 Las imágenes que se muestran han sido tomas del trabajo de Víctor Corral Blanch, en algunos casos coloreadas y escaladas según la necesidad. El vídeo fue elegido al azar con alguna similitud visual.

tas bajas y modificaciones de datos sobre las nuevas tablas desde *kPAX*, por ello al momento esas tareas deben realizarse externamente. Lo mismo se mantiene para la tabla *Category*.

Figura 18: Vista de la página de un juego

El *plugin kpax*

El conector recibió un tratamiento algo más extenso, comenzando por una depuración de archivos de resguardo o no utilizados, para luego actuar sobre un conjunto de vistas y controladores. La tabla 9 resume ese trabajo.

Archivo	Acción
/actions/kpax/save.php[41]	Se modifica para permitir incorporar al juego los valores de *skill* y *platform.*
	Se realiza una discriminación sobre los posibles errores de almacenamiento, que antes estaban agrupados bajo una sola denominación.
/languages/	Adecuación de los diccionarios en concordancia con las modificaciones en otros archivos.
/lib/kpax.php	Se agrupan en una sola función las acciones para obtener la categoría, plataforma y las habilidades de un juego.
/lib/kpaxSrv.php	Se agregan las funciones para obtener plataformas y habilidades, a la vez que se modifica la función *addGame()*.
/pages/kpax/my_dev_games.php	Prácticamente reescrito con características similares a la forma de visualizar los juegos en la pestaña *Games*.
/pages/kpax/play.php	Modificaciones sobre las distintas listas de juego.
/pages/kpax/view.php	Se modifica la acción que permite incorporar las imágenes por defecto cuando al obtener un juego, éste no contiene un *logo* o un *banner.*
	Se reemplazó *$page_owner→name* por *$page_owner→username*
	Se evita que en la sección *otros juegos del mismo desarrollador* se muestre el mismo juego que se está visualizando.
/views/default/forms/kpax/save.php	Se cambia el código para obtener los valores de *category, skill* y *platform* desde un menú desplegable.
kpax/games_list.php	Se modifica la referencia a las imágenes por defecto utilizando la función *elgg_get_site_url()* en lugar de la mención directa al servidor.
	Se modifica la referencia incorrecta a la edición de un juego.
	Se impide que se ofrezca la posibilidad de editar juegos a quién no sea su desarrollador.
	Se cambia la visualización de los botones utilizando para ello las clases definidas en *Elgg*.

5.2.3 Integración del módulo de búsqueda

A diferencia del caso anterior, el módulo se encontraba totalmente operativo, contando incluso con datos almacenados en las tablas de la base que permitían comprobar el funcionamiento.

41 Un desarrollador previo decidió que, al finalizar la acción de guardado mediante save.php, se ejecutara el código de la página *my_dev_games.php.* Como la misma no funcionaba adecuadamente, se referencia a *play.php*. Si bien el problema se soluciona en este trabajo, por cuestiones funcionales se prefiere mantener a *play*.

La tarea debió centrarse solamente en compatibilizar el trabajo con lo que se había llevado a cabo hasta el momento.

La base de datos

Como el desarrollo de Víctor Corral Blanch se llevó a cabo a partir de un par de proyectos previos, el autor se preocupó en mantener algunas tablas sobre las que su código no actúa específicamente.

Permanecen solamente aquellas que resulten indispensables para el funcionamiento de la aplicación en las condiciones de este trabajo.

La tabla *Game* recibió el foco de la atención puesto que debió ser compatibilizada con el módulo anterior.

Se hizo prevalecer la nomenclatura y tipos de campos adoptados en el primer trabajo, aunque hubiera sido correcto tratar al atributo *developer* como se hace en este módulo[42]. Por la forma en que se desarrolló el trabajo y fundamentalmente por cuestiones de tiempo, el resultado fue el que se comentó en el punto 4.2.3.

Gracias a que parte del trabajo de incorporación de nuevas relaciones se efectuó durante la integración anterior, solamente fue necesario agregar la tabla *MetaData*[43].

El núcleo

Se detallan en la Tabla 10 las principales modificaciones que permitieron la incorporación del nuevo módulo, agrupadas según los paquetes de clases

Paquete	Acción
VO	Modificación de las clases *Tag* y *GameView.*
	Incorporación de las clases *MetaData* y *GamePagination*.
	Adecuación de la clase *Game*.
DAO	Agregado de la clase e implementación *MetaDataDao*.
	En la clase *TagDao* y su implementación se incorpora el método que permite obtener un listado con todas las etiquetas de un juego[44].
	Sobre *GameDao* se incorporan los listados de juegos según los criterios de búsqueda y de juegos similares por categoría y etiquetas.
BO	En *Game* se agregan métodos que implementan las funcionalidades de listar los juegos y los juegos similares.
Rest	Agregado en la clase *Games* de los nuevos servicios.
Util	Sin cambios significativos

42 Representar al desarrollador mediante un identifcador numérico relacionado con la tabla *User* y no por el nombre de usuario.

43 Al momento, los valores almacenados por la tabla deben gestionarse externamente.

44 *getAllTagsGame(int idGame)*

El conector *kpax*

La tabla 11 resume las acciones más importantes llevadas a cabo.

Archivo	Acción
/languages/	Adecuación de los diccionarios.
/lib/kpaxSrv.php	Agregado de las funciones de búsqueda de juegos y funciones para la gestión de metadatos.
/pages/kpax/all.php	Incorporación de los criterios de búsqueda. Prácticamente reescrito.
/pages/kpax/play.php	Modificaciones menores
/pages/kpax/find.php	Se crea una nueva página conteniendo el código de búsqueda de juegos[45].
	Modificación del código para listar solamente los juegos autorizados.
/views/default/css/elements	Incorporación de las hojas de estilo.
	Modificación del estilo del botón buscar.
/views/default/kpax/find_games_list.php	Vista creada para mostrar la página *find.php*.
/views/default/kpax/games_list.php	Incorporación del botón *Find Games* para acceder al módulo de búsqueda.
	Mejoras en la visualización de la pagina.

5.3 Producto final

Una vez integrados ambos módulos, y en sucesivos ciclos de prueba, se fueron modificando algunas características adicionales de visualización.

Por ejemplo, ahora un desarrollador puede acceder a editar sus juegos, pero la opción desaparece cuando el trabajo no le pertenece.

Al momento de acercarse la fecha de entrega, se estaba modificando la opción de borrado de un juego – y todas sus implicancias sobre la base de datos – pero la tarea queda inconclusa.

En la Figura 19 se muestra una comparación de la pestaña *Games* cuando el administrador accede a ella y luego pulsa los botones *Show my games* y *Unauthorized Games*. Allí puede apreciarse el mensaje – también incorporado en este trabajo – que informa de la inexistencia de archivos para autorizar.

Al pulsar el botón *Find Games* se accede a la página de búsqueda de juegos y sobre la que también se han realizado algunos cambios.

La Figura 20 documenta esa situación y la Figura 21, muestra un ejemplo de una búsqueda de juegos que corresponden con la Categoría 1 y cuyas etiquetas contienen un número 1.

45 Originalmente incluido en /pages/kpax/play.php

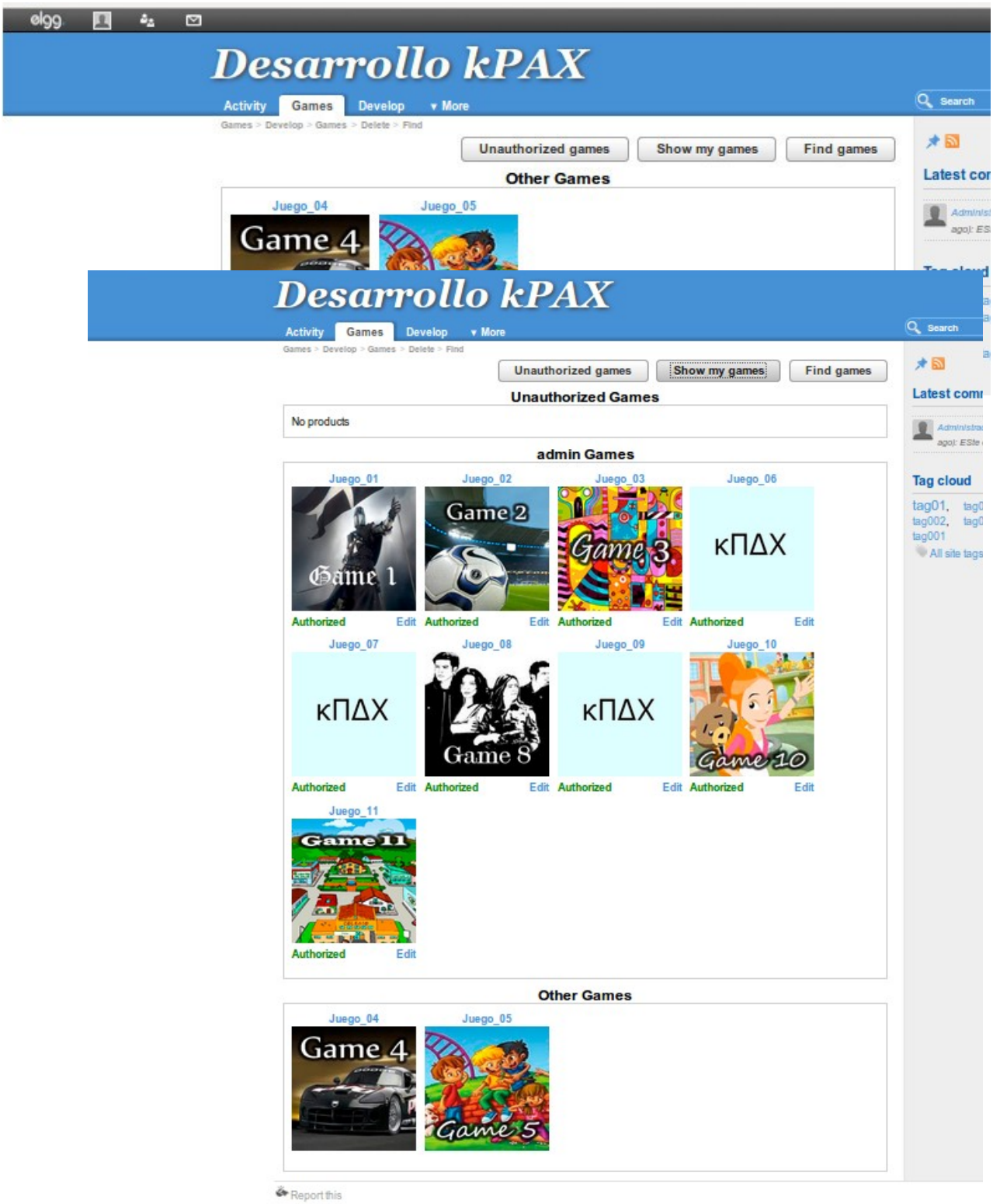

Figura 19: Vista de juegos

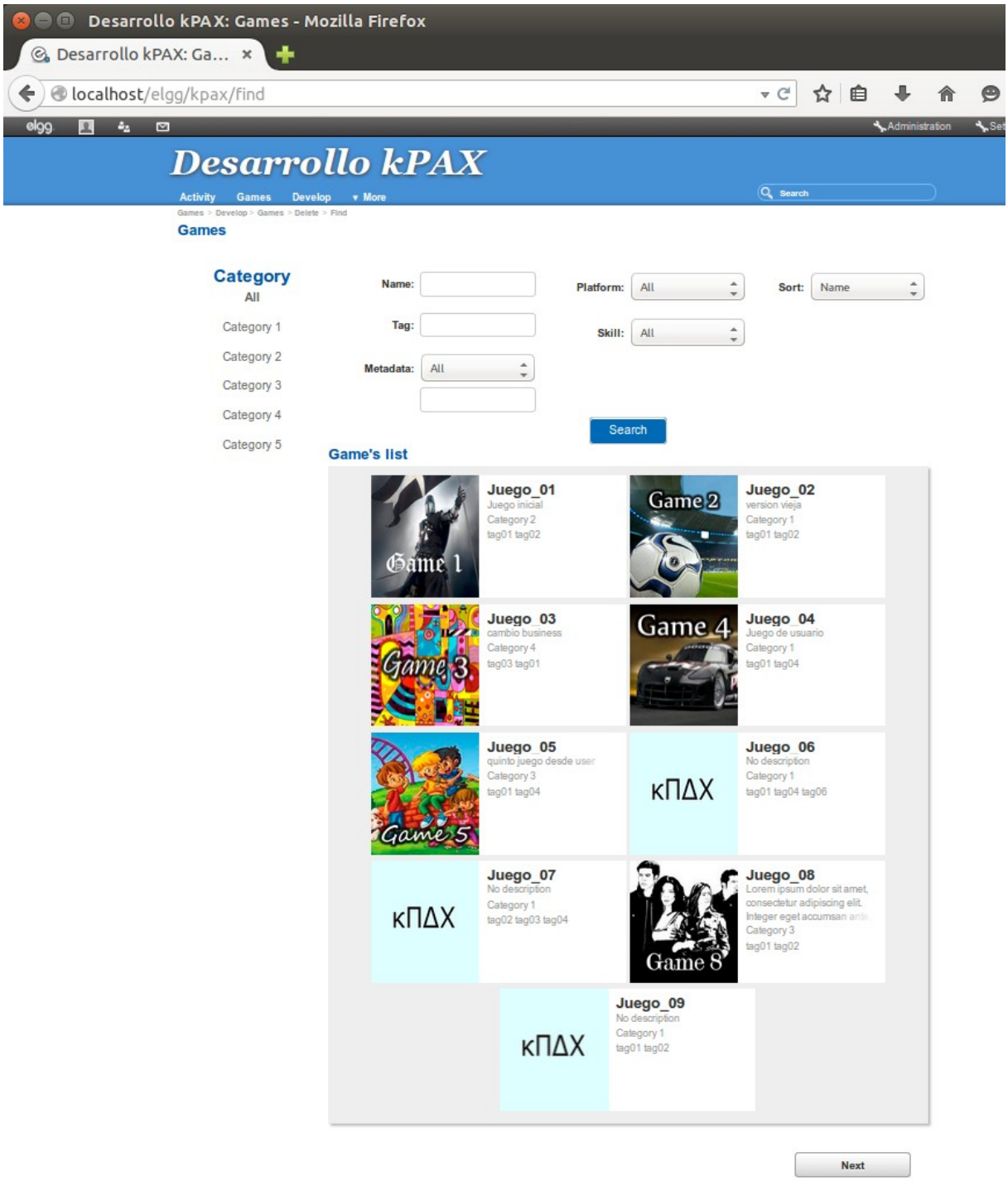

Figura 20: Búsqueda de juegos

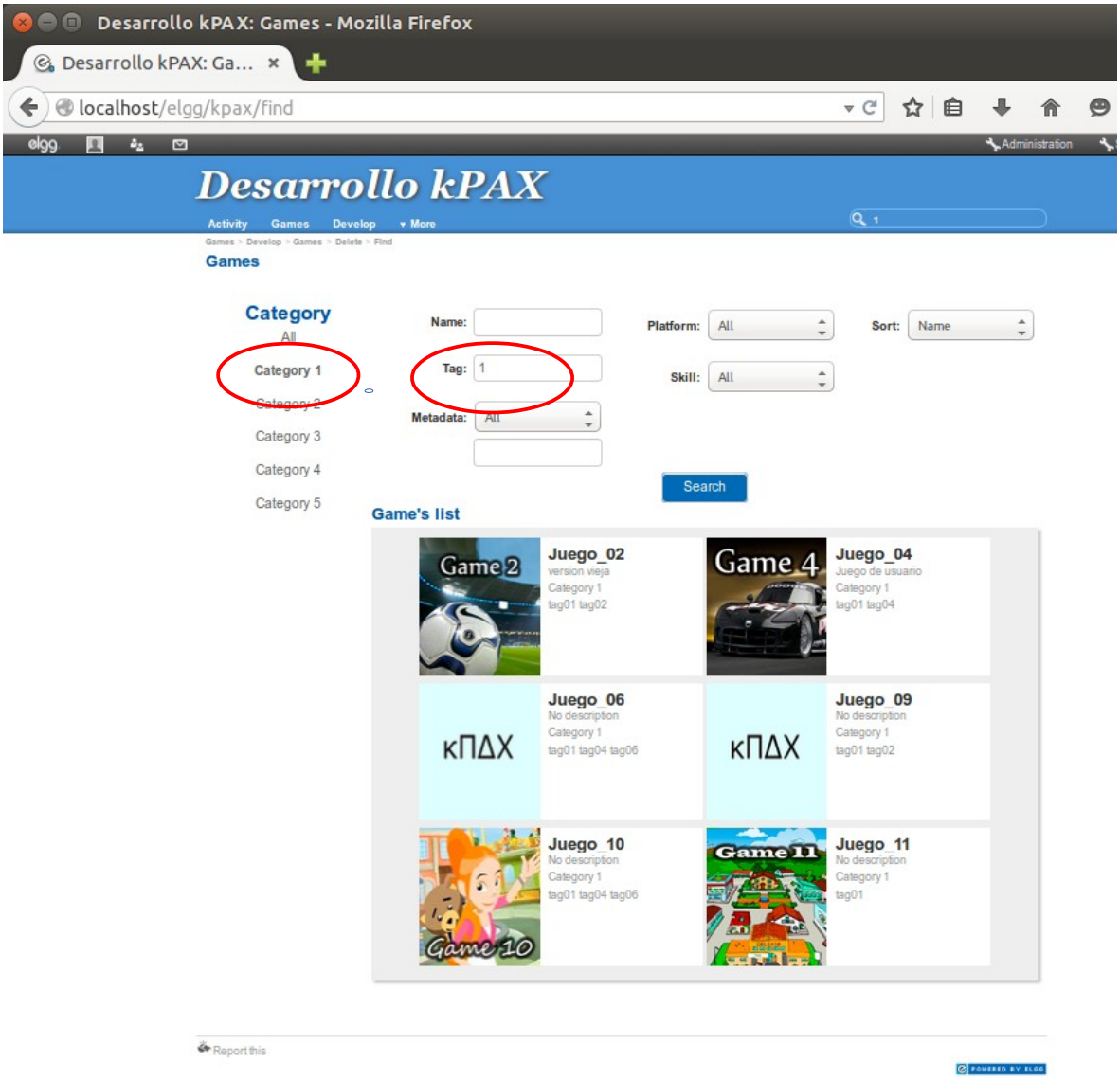

Figura 21: Filtrado en la búsqueda de juegos

La Figura 22 muestra la ficha de un juego y la Figura 23 la página de edición – es similar a la de carga – donde pueden observarse las modificaciones realizadas.

Sobre la primera de ellas se destaca que ahora no se visualiza el mismo producto en el sector de juegos similares.

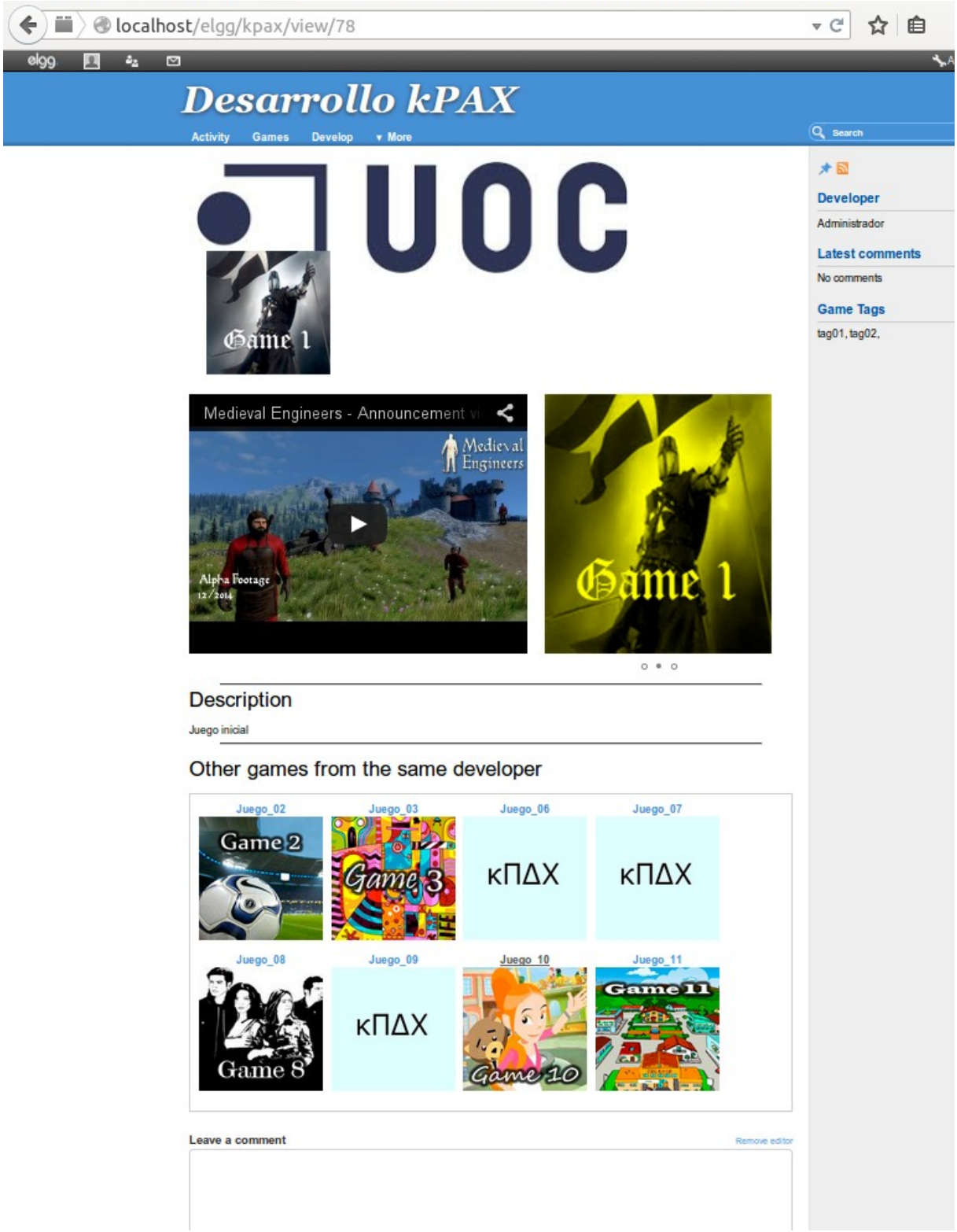

Figura 22: Descripción de un juego

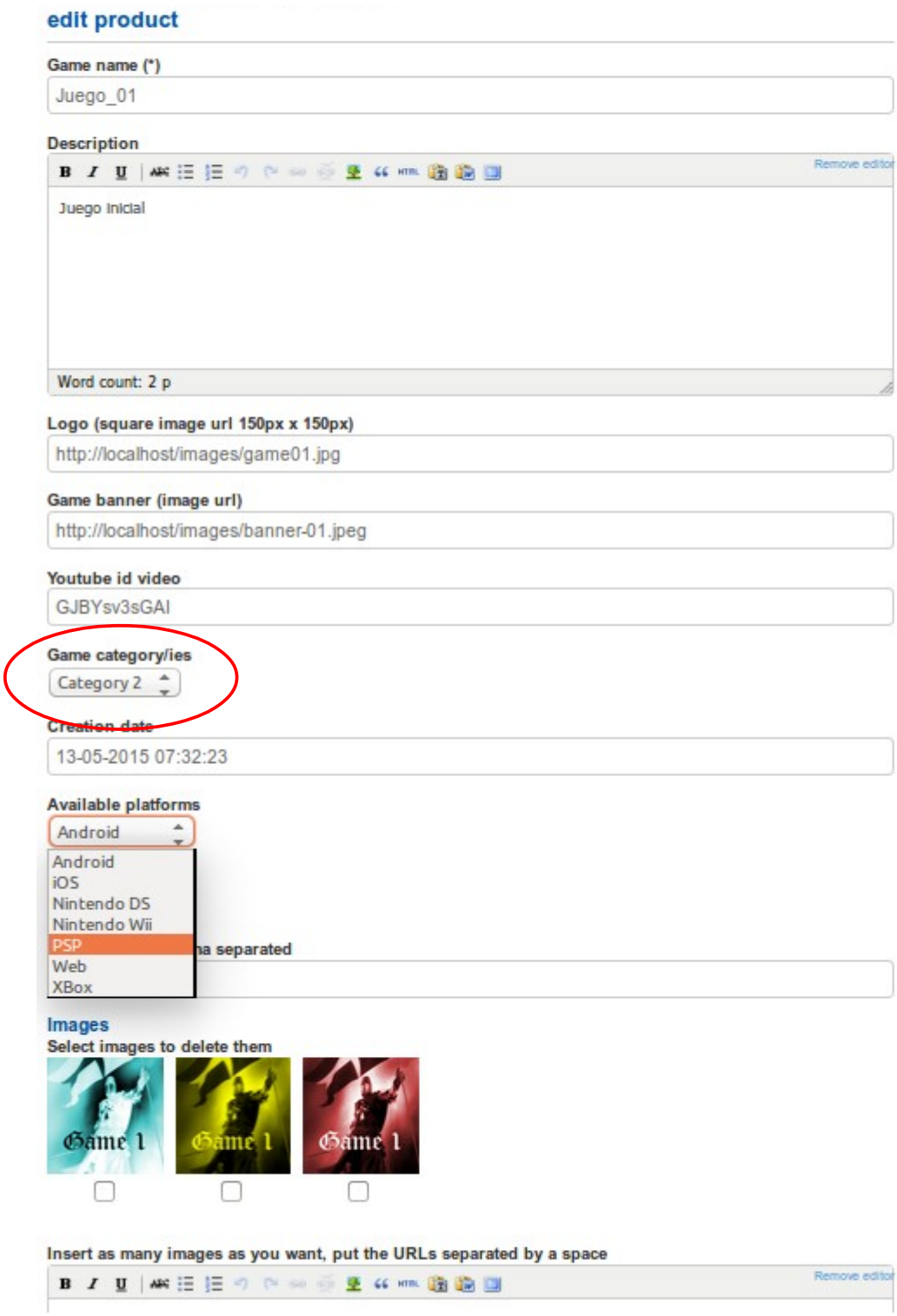

Figura 23: Edición de un juego

Finalmente, la Figura 24 exhibe la página *my_dev_games*, rediseñana en este trabajo.

Figura 24: Juegos del usuario

6. Conclusiones

El proyecto tenía como principal objetivo integrar en condiciones de funcionamiento sobre la plataforma *k-PAX,* distintas funcionalidades desarrolladas por terceros y que no estaban operativas en la versión inicial.

Se entiende que el propósito ha sido alcanzado, toda vez que ambos módulos se encuentran integrados y funcionales.

Para ello debieron modificarse la estructura de la base de datos, el núcleo de aplicación y el conector con *Elgg*.

Se intentó respetar al máximo el trabajo ajeno, introduciendo la menor cantidad de cambios al código original. Sin embargo, esto no siempre fue posible y algunas partes debieron ser reescritas.

También se incorporaron algunas mejoras a la vez que se corrigieron errores.

Como ejemplo puede mencionarse que ambos módulos integrados presentaban dificultades con la carga y edición de los juegos.

El primero de ellos no persistía todos los valores que se ingresaban desde el formulario de carga y en el segundo la acción directamente no funcionaba.

Una de las principales dificultades para realizar el trabajo se debió a que los desarrolladores originales enfrentaron los problemas de diseño de diferente manera y los resolvieron con criterios no siempre compatibles entre si.

Cabe recordar que el primero de los módulos no se recibió operativo, por lo que previamente debió lograrse la puesta en funcionamiento.

Por razones de tiempo no han podido realizarse algunos trabajos, tales como[46]:

- Solucionar los problemas de borrado de un juego y que aparentemente son de larga data.
- Modificar el diseño de la tabla *Game* de la base de datos y principalmente el código involucrado para hacer referencia al desarrollador mediante un identificador relacionado con la tabla *User.*
- Gestionar altas, bajas y modificaciones – desde el código de la aplicación – de las tablas asociadas con *Game* que a excepción de *Tag*, tampoco estaba previsto en los módulos originales.
- Minimizar la cantidad de accesos al servidor en cada consulta para reducir la latencia.

En otro orden, también pudo observase que *Elgg* es un proyecto en evolución y que con los sucesivos cambios algunas de sus funciones se modifican o desaparecen. Por la dependencia que tiene *k-PAX* de él, es muy probable que a futuro deba modificarse el código para contemplar esas variaciones.

Lamentablemente debido a que duración del estudio de la arquitectura de *k-PAX* – y *Elgg* – fue mayor a lo esperado y sumado al tiempo requerido para la reparación del primer módulo, no fue posible dedicar mayor cantidad de tiempo a la tarea de homo-

46 Tampoco estaban planificados, puesto que al inicio del trabajo los módulos se desaconocían.

geneizar del código y aplicar más profundamente el criterio *DRY*[47].

Ha resultado muy satisfactoria la forma de trabajo siguiendo una modalidad iterativa que ha permitido realizar refinaciones sucesivas, mejorando el código de la aplicación sobre la base del conocimiento adquirido.

La actividad ha sido muy provechosa, como también muy beneficioso el trabajo en equipo, el intercambio y el asesoramiento por parte del Consultor y del Tutor Externo.

El desarrollo realizado guarda estrecha relación con lo aprendido en diversas asignaturas de la maestría y la elección del tema facilitó profundizar sobre aquellos tópicos que no son estrictamente de dominio personal y profesional.

La experiencia posibilitó el contacto de primera mano con un proyecto de Software Libre muy amplio que utiliza diversas tecnologías, no todas con dominio previo.

Como se puede apreciar por lo expuesto, la envergadura del proyecto *k-PAX* es tal que, pese a los trabajos realizados, los faltantes siempre parecerán más.

Afortunadamente, al tratarse de un proyecto de Software Libre, otros desarrolladores lo continuarán.

47 Don’t Repeat Yourself – evitar código duplicado.

Bibliografía

[1] «Ludificación», *Wikipedia, la enciclopedia libre*.

[2] K. M. Kapp, *The gamification of learning and instruction: game-based methods and strategies for training and education*. San Francisco, CA: Pfeiffer, 2012.

[3] C. C. Abt, *Serious games*. Lanham, MD: University Press of America, 1987.

[4] «Escaparate de innovación - El lugar donde difundir los proyectos de innovación de la UOC - Programa d'Innovació de la UOC». [En línea]. Disponible en: http://www.innovauoc.org/showcase/?content=load_proyecto&id=112.

[5] Riera, Daniel et al, «kPAX. Plataforma d'aprenentatge en xarxa. Juga seriosament.» Universitat Oberta de Catalunya (UOC), 2011.

[6] «Elgg 1.9 Documentation — Elgg 1.9 documentation». [En línea]. Disponible en: http://learn.elgg.org/en/1.9/

[7] Sergio Melgar Gonzalez, «Proyecto final de carrera - Implementación de una red social para el aprendizaje basado en juegos - Módulo Developers». Universitat Autònoma de Barcelona.

[8] «Repositori institucional: Incorporació de funcionalitats al projecte kPAX». [En línea]. Disponible en: http://openaccess.uoc.edu/webapps/o2/handle/10609/35341.

[9] «Scrum Methodology & Agile Scrum Methodologies». [En línea]. Disponible en: http://scrummethodology.com/.

[10] «Representational state transfer - Wikipedia, the free encyclopedia». [En línea]. Disponible en: http://en.wikipedia.org/wiki/Representational_state_transfer#Guiding_principles_of_the_interface.

[11] «Create, read, update and delete», *Wikipedia, the free encyclopedia*.

[12] «Maven – Welcome to Apache Maven». [En línea]. Disponible en: http://maven.apache.org/.

[13] «About JBoss». [En línea]. Disponible en: https://docs.jboss.org/jbossas/docs/Server_Configuration_Guide/4/html/About_JBoss.html.

[14] «Jersey». [En línea]. Disponible en: https://jersey.java.net/.

[15] «Spring Framework Reference Documentation». [En línea]. Disponible en: http://docs.spring.io/spring/docs/current/spring-framework-reference/htmlsingle/.

[16] «Hibernate ORM - Hibernate ORM». [En línea]. Disponible en: http://hibernate.org/orm/.

[17] «jsanchezramos/mods-kpax · GitHub». [En línea]. Disponible en: https://github.-com/jsanchezramos/mods-kpax.

[18] «jsanchezramos (Juan Francisco Sánchez Ramos)». [En línea]. Disponible en: https://github.com/jsanchezramos.

[19] «Model View Controller». [En línea]. Disponible en: http://c2.com/cgi/wiki?Mo-delViewController.

[20] «Plugin skeleton — documentación de Elgg - master». [En línea]. Disponible en: http://learn.elgg.org/es/latest/guides/plugins/plugin-skeleton.html.

[21] «JAX-RS - Wikipedia, la enciclopedia libre». [En línea]. Disponible en: http://es.wikipedia.org/wiki/JAX-RS. [Accedido: 26-ene-2015].

[22] «Getting Started with Eclipse». [En línea]. Disponible en: https://www.eclip-se.org/users/.

[23] «Standards - W3C». [En línea]. Disponible en: http://www.w3.org/standards/.

[24] «Fotorama». [En línea]. Disponible en: http://fotorama.io/.

[25] «jQuery». [En línea]. Disponible en: https://jquery.com/.