Trivial4a

Escuela de Ingeniería Informática - Universidad de Oviedo

Grado en Ingeniería Informática del Software

Índice de contenidos

[Hoja de identificación 2](#_Toc418498965)

[Introducción 3](#_Toc418498966)

[Planteamiento del problema 3](#_Toc418498967)

[Metodología usada 4](#_Toc418498968)

[Metodología de trabajo 4](#_Toc418498969)

[Identificación de Interesados 4](#_Toc418498970)

[Descripción de los interesados 4](#_Toc418498971)

[Usuarios 4](#_Toc418498972)

[Administradores 4](#_Toc418498973)

[Responsables de NoGame 4](#_Toc418498974)

[Desarrolladores 4](#_Toc418498975)

[Roles e historias de usuario 5](#_Toc418498976)

[Roles 5](#_Toc418498977)

[Historias de usuarios 5](#_Toc418498978)

[Identificación de atributos de calidad 5](#_Toc418498979)

[Lista de atributos de calidad e interesados 6](#_Toc418498980)

[Solución planteada 7](#_Toc418498981)

[Módulo 1 – Extract 7](#_Toc418498982)

[Módulo 2 – Core 8](#_Toc418498983)

[Módulo 3 – Trivial web 8](#_Toc418498984)

[Escenarios de calidad 9](#_Toc418498985)

[Vistas 10](#_Toc418498986)

[Vistas de sistema 10](#_Toc418498987)

[Diagrama de componentes 10](#_Toc418498988)

[Diagrama de paquetes 10](#_Toc418498989)

[Anexos 10](#_Toc418498990)

[Diagrama BPMN 10](#_Toc418498991)

[Índice de figuras 10](#_Toc418498992)

[Índice de tablas 10](#_Toc418498993)

# Hoja de identificación

|  |  |
| --- | --- |
| Nombre del trabajo: | Trivial4a |
| Fecha: | Mayo 2015 |
| Autores: | |  |  | | --- | --- | | Nombre | UO | | Jorge Sierra Miranda | 226566 | | David Sariego Fernández | 223935 | | Santiago Marqués Rivera | 225086 | | Lucía Sánchez Díaz | 205169 | | Cristina Montes Rubio | 226259 | | Ángel Rodríguez Bobes | 214459 | |  |  | |  |  | |

# Introducción

La arquitectura descrita en este documento pretende dar cabida al desarrollo de una plataforma de juego. Esta plataforma pretende imitar al juego “Trivial” además de ofrecer funcionamiento adicional.

En el documento se describirá la arquitectura del sistema comparándolo con los patrones arquitectónicos conocidos resaltando aquellas características que se hayan intentado obtener con dichos patrones.

# Planteamiento del problema

NoGame, una empresa del sector de los videojuegos está interesada en la creación de una aplicación con un funcionamiento similar al juego de mesa “Trivial”.

La aplicación no será sólo una plataforma de juego, sino que deberá además guardar registros de las actividades de los jugadores.

Debido a la naturaleza del juego las preguntas deberán poder añadirse, quitarse, y cambiarse; permitiendo jugar cada vez con un set de preguntas distinto.

Adaptándose a la las nuevas tecnologías el sistema estará disponible para jugar en aplicaciones de escritorio o a través de la web.

Por último, los datos recopilados por la información deber estar disponibles para los administradores del sistema. Esta funcionalidad permitirá la reutilización de la plataforma en entornos profesionales o educacionales para la realización de controles o estadísticas de personal.



Ilustración 1 - Diagrama de contexto

# Metodología usada

El estudio de la arquitectura se realiza siguiendo el método de ADD (Atribute-Driven Design) (Bass, Clements, & Kazman, Software Architecture in Practice, Second Edition, 2003).

## Metodología de trabajo

La metodología para el desarrollo de la aplicación se basa en el reparto de tareas.

Las tareas repartidas se podrán repartir a más de una persona creando la posibilidad de dividir la tarea en subtareas o de realizar la tarea conjuntamente con técnicas como *pair porgramming*.

La comunicación del equipo se realizará presencialmente en el centro, a partir de mensajería instantánea en Skype y a través de Github, aprovechando todas las herramientas disponibles en este.

El estilo de desarrollo del software ha sido estilo incremental (*Incremental piecemeal*) por la manera en la que se nos ha entregado a los alumnos los enunciados.

# Identificación de Interesados

Los interesados en la aplicación, también llamados *stakeholders* son los siguientes:

* Usuarios
* Administradores
* Desarrolladores

## Descripción de los interesados

Se tendrán en cuenta las distintas características de los *stakeholders* para el desarrollo del sistema. Cada tipo de *stakeholder* tendrá unos objetivos distintos en la aplicación.

### Usuarios

Son los que utilizarán el sistema participando en juegos y respondiendo preguntas.

Deberán estar registrados en el sistema.

Se guardarán los datos de cómo estos usuarios interactúan con el sistema.

### Administradores

Son los usuarios encargados del sistema que tendrán acceso a los datos recopilados por la aplicación.

### Responsables de NoGame

NoGame como empresa contratante es un interesado en la aplicación.

No participarán directamente en la aplicación pero sus necesidades son aquellas que la aplicación intenta cubrir.

### Desarrolladores

Son aquellas personas designadas al desarrollo de la aplicación.

Interactuarán con el sistema en la fase de desarrollo y tendrán como objetivo crear un sistema que sea fácil de mantener, con un coste reducido y abierto para la extensión de su funcionalidad.

# Roles e historias de usuario

## Roles

Los roles en el sistema se corresponden con dos tipos de interesados.

* Usuarios
* Administradores

## Historias de usuarios

Tabla 1 - Historias de usuario

|  |
| --- |
| Como administrador quiero poder añadir preguntas a la aplicación desde ficheros GIFT. |
| Como administrador quiero poder ver las estadísticas de todos los usuarios. |
| Como administrados quiero poder modificar los perfiles de otros usuarios. |
| Como usuario quiero poder identificarme para utilizar la plataforma. |
| Como usuario quiero poder cerrar sesión. |
| Como usuario quiero poder ver mis estadísticas. |
| Como usuario quiero poder empezar una partida. |
| Como usuario quiero poder cambiar mis datos en la aplicación. |

# Identificación de atributos de calidad

Los atributos de calidad identificados para este sistema son:

#### Disponibilidad

El sistema debe poder estar disponible las 24 del día.

#### Modificabilidad

El sistema tiene que poder sufrir cambios como nuevos ficheros fuente, nuevos formatos de representación y posibles cambios de base de datos sin que estos resulten costosos.

#### Escalabilidad

El incremento del número de usuarios y de las preguntas almacenadas no debe suponer cambio alguno para la aplicación.

#### Rendimiento

Las acciones interactivas realizadas sobre el sistema deben ser instantáneas a la vista de los usuarios.

#### Seguridad

Los accesos a la aplicación deben ser controlados con cuentas de usuario y contraseña.

Se debe asegurar la integridad de los datos almacenados.

#### Testeabilidad

El funcionamiento de la aplicación se debe probar con facilidad.

#### Usabilidad

La aplicación debe ser amigable con el usuario, es decir, fácil de usar y entender.

# Lista de atributos de calidad e interesados

Tabla 2 - Stakeholders

|  |  |  |
| --- | --- | --- |
| CÓDIGO | STAKEHOLDER | INTERESES |
| S1 | Usuario | * Facilidad de uso * Rapidez en la interacción |
| S2 | Administrador | * Facilidad de uso * Seguridad de acceso |
| S3 | Responsable de NoGame | * Bajo coste de desarrollo * Aplicación resultante correcta * Entrega dentro del plazo * Fácilmente ampliable |
| S4 | Desarrollador | * Aplicación resultante correcta * Facilidad para mantener el sistema |

Tabla 3 - Atributos de calidad

|  |  |  |
| --- | --- | --- |
| CÓDIGO | DESCRIPCIÓN | TIPO DE ATRIBUTO |
| A1 | El sistema debe poder estar disponible las 24 del día | Disponibilidad |
| A2 | El sistema tiene que poder sufrir cambios como nuevos ficheros fuente, nuevos formatos de representación y posibles cambios de base de datos sin que estos resulten costosos. | Modificabilidad |
| A3 | El incremento del número de usuarios y de las preguntas almacenadas no debe suponer cambio alguno para la aplicación. | Escalabilidad |
| A4 | Las acciones interactivas realizadas sobre el sistema deben ser instantáneas a la vista de los usuarios. | Rendimiento |
| A5 | Los accesos a la aplicación deben ser controlados con cuentas de usuario y contraseña | Seguridad |
| A6 | Se debe asegurar la integridad de los datos almacenados. | Seguridad |
| A7 | El funcionamiento de la aplicación se debe probar con facilidad | Testeabilidad |
| A8 | La aplicación debe ser amigable con el usuario, es decir, fácil de usar y entender | Usabilidad |

Tabla 4 - Relacion Steakeholder/Atributo de calidad

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | S1 | S2 | S3 | S4 |
| A1 | √ | √ | √ |  |
| A2 |  |  | √ | √ |
| A3 |  | √ | √ | √ |
| A4 | √ | √ |  |  |
| A5 |  | √ | √ |  |
| A6 |  | √ |  | √ |
| A7 |  |  |  | √ |
| A8 | √ |  |  |  |

# Solución planteada

La solución planteada para los requisitos del sistema es la realización de una aplicación basada en tres módulos relacionados.

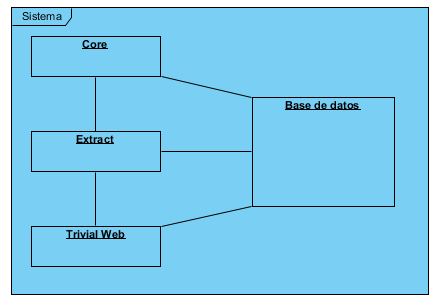


Ilustración 2 - Solución planteada

Los módulos comparten una única base de datos de forma similar a estilo arquitectónico de **base de datos compartida**, obteniendo de esta manera algunas de las ventajas de dicho estilo. Sin embargo los módulos de este sistema no son independientes lo que desvirtúa el patrón.

## Módulo 1 – Extract

Este módulo es el inicial del sistema. Desde este módulo se añaden preguntas a la base de datos desde ficheros en formato GIFT. Es decir, es un parser que lee ficheros y actualiza la base de datos. También se puede usar para escribir ficheros intermedios para el sistema.

El diseño de software interno consiste en la separación de responsabilidades. Independientemente de su diseño interno, este módulo tiene el comportamiento de una aplicación secuencial o batch. Por las características de este módulo y las necesidades que debe cubrir este comportamiento es el óptimo.

## Módulo 2 – Core

Este módulo consiste en una aplicación de escritorio dónde uno o varios usuarios pueden identificarse para participar en un juego. Al finalizar la partida los resultados de la partida son almacenados en la base de datos.

Este módulo se corresponde con el patrón *modelo vista controlador* (**MVC)**.Este patrón cubre todas las necesidades de este módulo, adaptándose al modelo de dominio, permitiendo distintas representaciones (vistas) del juego a la vez que controla el desarrollo de la partida.

## Módulo 3 – Trivial web

Este módulo es más interesante de la aplicación y el que más se explotará una vez se desarrolle la plataforma.

Este módulo permite jugar partidas personales a partir de un navegador web. Además dispone de vistas para los datos que almacena la aplicación para que los usuarios vean sus propios datos y los administradores controlen los datos de todos.

Para cubrir todas las necesidades y requisitos del sistema este módulo se desarrolla con el Framework Play. Este framework permite crear aplicaciones web basadas en el **MVC**. Al basarse en **MVC** tiene todas las ventajas que este patrón ofrece. Además al ser una aplicación web puede instalarse en un servidor, permitiendo el estilo de integración Cliente-Servidor. Cualquier cliente puede conectarse a un servidor que tenga la plataforma Trivial4a instalada y participar en juegos o revisar los datos.

# Escenarios de calidad

| Nº | Fuente de estímulo | Estímulo | Entorno | Artefacto | Respuesta | Medición de la respuesta | Atributo de Calidad afectado |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | Desarrollador | Arranque del sistema | Desarrollo | Sistema | Tiempo para que el sistema esté funcionando razonablemente corto | Tiempo de arranque de la aplicación < 30 seg | AT001 |
| 2 | Desarrollador | Nuevo formato de preguntas | Desarrollo | Sistema | Implementar la nueva funcionalidad en un tiempo razonablemente corto | Tiempo de desarrollo < 2 días | AT002 |
| 3 | Desarrollador | Aumento del repositorio de preguntas | Desarrollo | Sistema/Base de datos | Adición de preguntas a la base de datos del sistema | Tiempo para la inclusión de las preguntas de un fichero < 2 min | AT003 |
| 4 | Usuario | Intento de acceso a datos privados del sistema | Explotación | Base de datos | Negación del sistema a realizar la operación pertinente | Nº de intrusiones y accesos no permitidos =0 | AT004 |
| 5 | Usuario | Almacenamiento de datos en el sistema | Explotación | Base de datos | Almacenamiento correcto de los datos del usuario en el sistema. | Sin pérdida de información | AT005 |
| 6 | Desarrollador | Pruebas del sistema | Desarrollo | Sistema | Ejecución de pruebas del sistema | Superación de las pruebas definidas para la aplicación | AT006 |
| 7 | Usuario | Utilización del sistema | Explotación | Sistema | El sistema será simple e intuitivo | Tiempo de aprendizaje para la utilización del sistema < 5 min | AT007 |

# Vistas

## Vistas de sistema



Ilustración .1 - Diagrama del Sistema (I)



Ilustración .2 - Diagrama del Sistema (II)

|  |  |
| --- | --- |
| Elemento del  sistema | Descripción |
| Usuario | Usuarios de la aplicación (jugadores) que interactuarán constantemente con el Servidor Trivial4a para registrarse y acceder a sus cuentas |
| Servidor Trivial4a | Subsistema encargado de la interacción con el Usuario y el Servidor Trivial4a. |
| MongoDB | Base de datos en la que se almacenará toda la información que será utilizada por la aplicación de trivial (cuentas, preguntas y estadísticas) o que será obtenida a través de la misma (registro). |

## Diagrama de componentes

[[Diagrama de componentes]]

## Diagrama de paquetes

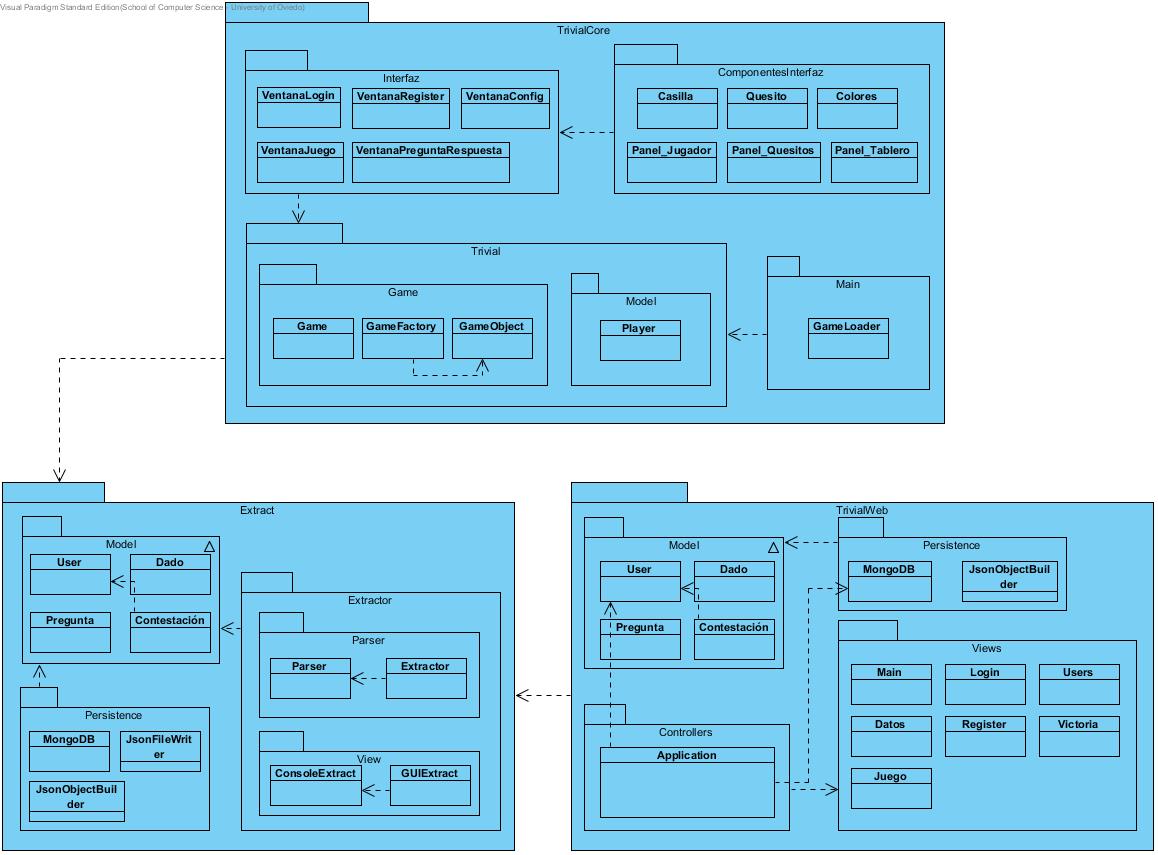


Ilustración 3.4- Diagrama de Paquetes

|  |  |
| --- | --- |
| Elemento | Descripción |
| Extract | Añade preguntas a la base de datos. Se encarga de leer/parsear ficheros y actualizar la base de datos. |
| TrivialCore | Juego. |
| TrivialWeb | Juego en versión web |

|  |  |
| --- | --- |
| Elemento EXTRACT | Descripción |
| Model | Modelo de dominio. |
| Extractor | Paquete para la aplicación que extrae las preguntas. |
| Persistence | Paquete para la gestión de la base de datos. |

|  |  |
| --- | --- |
| Elemento Extract/Model |  |
| User | Representación del usuario de la aplicación. Incluye sus datos y datos relativos a las partidas. |
| Dado | Representación del dado. |
| Pregunta | Representación de la pregunta del juego. |
| Contestación | Representa la respuesta de un usuario a una pregunta del juego. |
| Elemento Extract/Extractor | |
| Parser | Paquete con el parser de ficheros. |
| View | Paquete con la parte visual del extractor. |
| Elemento Extract/Extractor/Parser | |
| Extractor | Parsea un fichero de preguntas |
| Parser |  |
| Elemento Extract/Extractor/View | |
| ConsoleExtract | Aplicación del extractor de preguntas. |
| GUIExtract | Arranca el programa extractor de preguntas |
| Elemento Extract/Persistence | |
| MongoDB | Operaciones de gestión de datos de usuarios y preguntas. |
| JsonFileWriter | Escribe el fichero JSON |
| JsonObjectBuilder | Creación de preguntas y/o usuarios a partir de un fichero JSON |

|  |  |
| --- | --- |
| Elemento TRIVIALCORE | Descripción |
| Interfaz | Paquete con la parte gráfica del juego. |
| ComponentesInterfaz | Paquete con componentes gráficos del juego. |
| Trivial | Lógica del juego. |
| Main | Arranque de la aplicación. |

|  |  |
| --- | --- |
| Elemento TrivialCore/Interfaz |  |
| VentanaLogin | Ventana para que el usuario entre en sesión. |
| VentanaRegister | Ventana para que un usuario nuevo se registre en la aplicación. |
| VentanaConfig | Ventana donde el usuario configure el juego |
| VentanaJuego | Ventana principal donde el usuario juega. |
| VentanaPreguntaRespuesta | Ventana donde el jugador responde a la pregunta que se le muestra. |
| Elemento TrivialCore/ComponentesInterfaz | |
| Casilla | Representación de las casillas del tablero. |
| Quesito | Respresenta un quesito. |
| Colores | Asocia un color con una categoría. |
| Panel\_Jugador | Panel del jugador correspondiente |
| Panel\_Quesitos | Para visualizar los quesitos del jugador correspondiente |
| Panel\_Tablero | Visualización del tablero. |
| Elemento TrivialCore/Trivial | |
| Game | Paquete para la creación de partidas. |
| Model | Modelo para partidas. |
| Elemento TrivialCore/Trivial/Game | |
| Game | Interfaz con las operaciones propias de la partida. |
| GameFactory | Crea la partida |
| GameObject | Representa una partida. |
| Elemento TrivialCore/Trivial/Model | |
| Player | Implementa las operaciones del jugador en una partida. |
| Elemento TrivialCore/Main |  |
| GameLoader | Carga las preguntas y crea la partida. |

|  |  |
| --- | --- |
| Elemento TrivialWeb | Descripción |
| Model | Modelo de dominio. |
| Controllers | Controlador de la aplicación. |
| Persistence | Paquete para la gestión de la base de datos. |
| Views | Paquete con las vistas. |

|  |  |
| --- | --- |
| Elemento TrivialWeb/Model | |
| User | Representación del usuario de la aplicación. Incluye sus datos y datos relativos a las partidas. |
| Dado | Representación del dado. |
| Pregunta | Representación de la pregunta del juego. |
| Contestación | Representa la respuesta de un usuario a una pregunta del juego. |
| Elemento TrivialWeb/Controllers | |
| Application | Controlador de la aplicación. |
| Elemento TrivialWeb/Persistence | |
| MongoDB | Operaciones de gestión de datos de usuarios y preguntas. |
| JsonObjectBuilder | Creación de preguntas y/o usuarios a partir de un fichero JSON |
| Elemento TrivialWeb/Views | |
| Main | Pantalla de principal del usuario en sesión. |
| Datos | Formulario para visualizar y cambiar datos propios. |
| Juego | Pantalla principal del juego. |
| Login | Formulario de inicio de sesión. |
| Register | Formulario de registro en la aplicación. |
| Users | Lista para la gestión de usuarios. |
| Victoria | Pantalla para partida ganada. |

# Anexos

## Diagrama BPMN

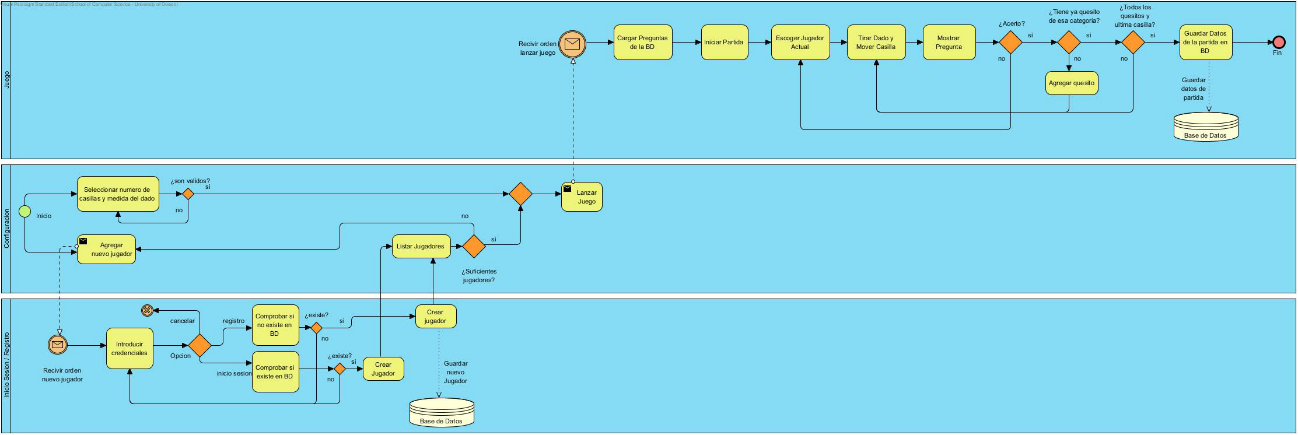


Ilustración 4 - Diagrama BPMN

Un diagrama a tamaño real se puede encontrar en la siguiente dirección:

<http://alniarez.host56.com/ASW/Trivial4a/BPMN.png>

## Índice de figuras

## Índice de tablas