

Trivial 4b

Arquitectura del software

04/03/2015

Grupo 4b

Tabla de contenido

| | |
|---|---|
| 1. Introducción | 3 |
| 1.1. Propósito..... | 3 |
| 1.2. Alcance..... | 3 |
| 1.3. Referencias | 3 |
| 1.4. Visión general..... | 4 |
| 2. Stakeholders | 4 |
| 2.1. Dueño de la empresa contratante..... | 4 |
| 2.2. Equipo de desarrollado..... | 4 |
| 2.3. Administrador de la base de datos..... | 4 |
| 2.4. Usuario de la aplicación | 4 |
| 3. Requisitos del proyecto..... | 5 |
| 3.1. Requisitos funcionales..... | 5 |
| 3.2. Requisitos no-funcionales | 5 |
| 4. Atributos de calidad | 5 |
| 5. Escenarios de calidad | 6 |
| 6. Vistas del Modelo de Diseño | 6 |
| 6.1. Diagrama de paquetes | 6 |
| 6.2. Diagrama de clases..... | 7 |
| 6.2.1. Resumen..... | 7 |
| 7. Manual de usuario..... | 8 |
| 7.1 Eclipse..... | 8 |
| 7.2 Línea de comandos..... | 8 |

Tabla de figuras

| | |
|-------------------------------|---|
| 1. Diagrama de paquetes | 6 |
| 2. Diagrama de clases..... | 7 |

1. Introducción

En este proyecto se intenta dar solución a las necesidades planteadas por la empresa NoGame. Dicha empresa se dedica a la creación de videojuegos de pregunta/respuestas y tiene como proyecto el desarrollo de una línea de juegos basados en una variante del Trivial.

Previamente al desarrollo del videojuego la empresa quiere hacerse con una base de preguntas y respuestas utilizando para ello un banco de preguntas ya existente. Dicho banco de preguntas distribuye las preguntas en ficheros con el formato GIFT. Debido a ello la empresa NoGame quiere poseer una aplicación que le permita convertir este formato de ficheros en otro formato.

La aplicación debe ejecutarse en dos etapas. Una primera etapa que analiza los ficheros con las preguntas y otra etapa en la que exportar las preguntas a un fichero de salida y las almacena en la base de datos. El formato que se desea obtener es JSON, el cual podrá ser usado más adelante para insertar su contenido en una base de datos, la cual en principio podría ser MongoDB aunque aún no ha sido confirmado.

La empresa NoGame quiere que la aplicación sea lo más autosuficiente posible, es decir, que se pueda automatizar en lo máximo posible el proceso de conversión entre formatos. No obstante esto no debe impedir que pueda llevar a cabo un proceso de revisión y depurado el cual muestre los errores que hayan podido ocurrir.

1.1. Propósito

Este documento proporciona una apreciación global y comprensible de la arquitectura del sistema usando diferentes puntos de vista para mostrar distintos aspectos del sistema. Intenta capturar y llegar a las decisiones de arquitectura críticas que han sido hechas en el sistema.

1.2. Alcance

En este documento se analizarán los requisitos de nuestra arquitectura tanto funcional como no funcional. Además se analizarán detalladamente cada uno de los stakeholders y sus respectivos intereses dentro del proyecto. Así como también la distribución arquitectónica del proyecto mediante los pertinentes diagramas. Y por último un manual de usuario, que servirá como ayuda para la implementación y uso del programa final. Todo ello será plasmado en el documento de manera clara y sencilla para su correcta comprensión evitando toda posible mal interpretación.

1.3. Referencias

Para la elaboración de este documento se usará como referencia el documento de la guía académica de la asignatura Arquitectura del Software.

1.4. Visión general

La organización de documento consistirá en diferentes apartados en los que se analizarán los Stakeholder y sus necesidades respecto al proyecto, las vistas del diseño arquitectónico, los requisitos de calidad y sus diferentes escenarios.

2. Stakeholders

En esta sección se analizaran los diferentes Stakeholders que tienen relación con nuestro proyecto.

2.1. Dueño de la empresa contratante

Esta persona exigirá un producto que sea estable que sea desarrollado en el menor tiempo posible ya que quiere poder lanzar al mercado cuanto antes su nueva línea de juegos, basados en la temática trivial, que constará de diferentes preguntas con multirespuesta.

2.2. Equipo de desarrollo

Este grupo de personas quieren tener un margen de tiempo amplio para poder desarrollar el programa de manera que sea fácilmente mantenible y extensible a nuevas funcionalidades.

2.3. Administrador de la base de datos

Esta persona quiere que nuestro programa disponga de configuración que permita modificar las opciones de conexión a la base de datos de una manera fácil y sencilla. También exigirá que nuestro programa sea capaz de conectarse y trabajar con cualquier tipo de base de datos que pueda necesitar en el futuro.

2.4. Usuario de la aplicación

Esta persona será la que va a utilizar potencialmente la aplicación, requerirá que la aplicación no tenga una gran dificultad a la hora de usarla y que cumpla con los objetivos que desea, en este caso, la conversión de un fichero con formato “.gift” o “.xml” a “JSON”.

3. Requisitos del proyecto

3.1. Requisitos funcionales

- El programa debe generar un fichero de salida en formato JSON con las preguntas y respuestas.
- Las preguntas deben ser obtenidas de una fuente externa, un fichero de formato GIFT.
- El programa debe ser lo menos interactivo posible. Tan solo debe solicitar unos simples datos, tales como, el nombre del fichero de entrada, el nombre del fichero de salida y la extensión del fichero en ambos casos.
- Se debe almacenar las preguntas y respuestas en una base de datos.
- El proceso debe poder ser automatizado.

3.2. Requisitos no-funcionales

- Poder agregar preguntas desde diferentes formatos de fichero.
- La base de datos puede ser SQL o NOQSL como por ejemplo MongoDB.
- Generar un log que permita analizar errores en el proceso de conversión.

4. Atributos de calidad

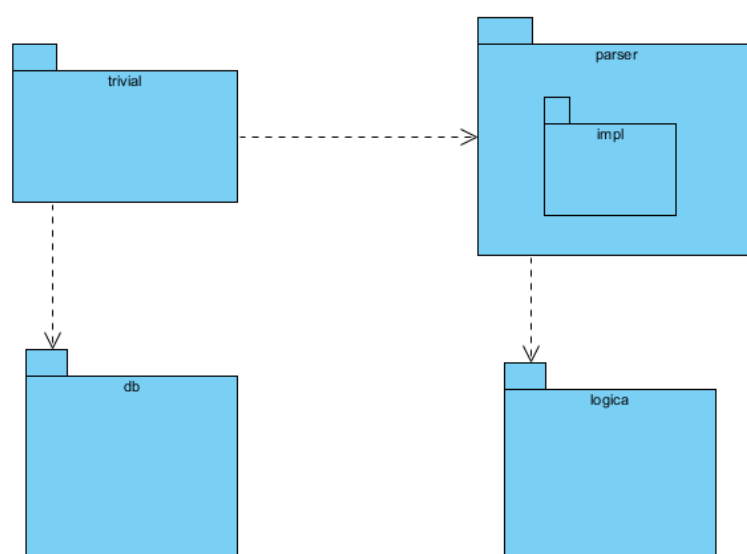
| Código | Atributo | Descripción |
|---------|---------------|--|
| AC – 01 | Adaptabilidad | Capacidad de procesar ficheros con distintas extensiones. |
| AC – 02 | Usabilidad | Sencilla comprensión y uso de la aplicación. |
| AC – 03 | Testabilidad | Facilidad para verificar el producto y poder analizar errores |
| AC – 04 | Conectividad | Posibilidad de conectarse a una base de datos externa al propio sistema. |

5. Escenarios de calidad

| Código | Fuente | Estímulo | Entorno | Artefacto | Respuesta | Medición | Atributo afectado |
|---------|-----------------------------------|--|-------------|------------------------------|---|--|-------------------|
| EC – 01 | Usuario | Iniciar recopilación de preguntas | Explotación | Lector de fichero de entrada | Se parsea el fichero de entrada | | AC – 02 |
| EC – 02 | Usuario | Guardar preguntas en la base de datos | Explotación | Conector a la base de datos | Las preguntas son guardadas en la base de datos | Comprobar la base de datos. | AC – 04 |
| EC – 03 | Usuario | Extraer en preguntas el fichero de salida | Explotación | Conversor | Obtener el fichero de salida con las preguntas | Comprobar el fichero de salida | AC – 02 |
| EC – 04 | Usuario | Analizar errores | Explotación | Sistema en conjunto | Se crea un log con errores | Se analiza el log para ver dónde están los errores | AC – 03 |
| EC – 05 | Administrador de la base de datos | Cambiar base de datos | Desarrollo | Conector a base de datos | Cambiar conector de conexión a base de datos | | AC – 04 |
| EC – 06 | NoGame | Añadir nuevos formatos de fichero de entrada | Desarrollo | Lector de fichero de entrada | Se programan nuevos filtros para los fichero de entrada | Comprobar que sean válidos los nuevos formatos | AC – 01 |
| EC – 07 | NoGame | Añadir nuevos formatos de fichero de salida | Desarrollo | Conversor | Se programan nuevos ficheros de salida | Comprobar que el fichero de salida es correcto | AC – 01 |

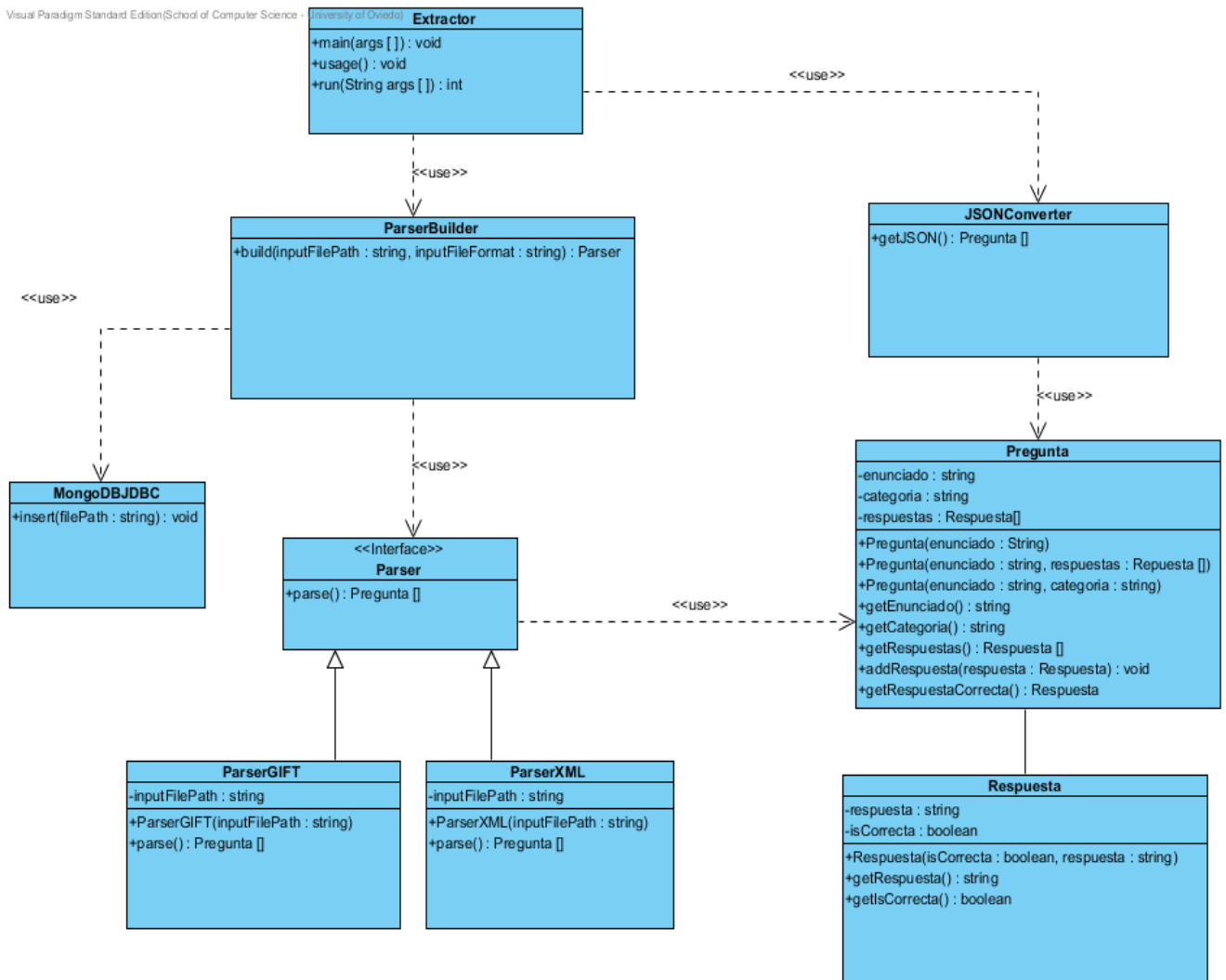
6. Vistas del Modelo de Diseño

6.1. Diagrama de paquetes









1. Diagrama de paquetes




6.2. Diagrama de clases



2. Diagrama de clases

6.2.1. Resumen

| Nombre | Documentación |
|--|---|
|  Extractor | Esta clase es la encargada del lanzamiento de la aplicación. |
|  JSONCon- verter | Esta clase se encarga de obtener una cadena con el JSON a partir de una lista de preguntas. |
|  Parser- Builder | Crea el parser que se pide por parámetro. Por defecto crea un Parser de tipo Gift |
|  Pregunta | Esta clase forma parte del modelo del dominio del problema. |
|  MongoD- BJDBC | Clase encargada de realizar la conexión e inserción de datos en la base de datos MongoDB. |
|  Parser | Interfaz de las clases ParserGIFT y ParserXML |

| | |
|--|---|
|  ParserGIFT | Esta clase es la encargada de leer un fichero con formato .GIFT y convertirlo en una lista de preguntas, que servirán como paso intermedio para obtener el JSON |
|  ParserXML | Esta clase es la encargada de leer un fichero con formato .XML y convertirlo en una lista de preguntas, que servirán como paso intermedio para obtener el JSON |
|  Respuesta | Esta clase sirve para utilizarla como parte del modelo del dominio del problema. |

7. Manual de usuario

Existen dos maneras de hacer funcionar la aplicación, mediante eclipse, o mediante la línea de comandos.

7.1 Eclipse

Mediante eclipse, el usuario deberá importar el proyecto en su directorio de trabajo. Una vez importado deberemos de hacer click derecho sobre la clase Extractor y pinchar en “run as/run configurations” seleccionar la pestaña de arguments de dicha clase e introducir en éste orden los siguientes parámetros:

- InputFilePath: Que será la ruta donde está el archivo que se desea convertir.
- InputFileFormat: Será el formato del archivo que se desea convertir, en este caso, GIFT o XML.
- OutputFilePath: Ruta del directorio donde queremos situar el archivo con formato JSON que hemos obtenido.

Si se quiere realizar la inserción en la base de datos, es necesario que el usuario tenga una instancia de mongodb activa en el momento del lanzamiento de la aplicación, además de la confirmación cuando el programa le pregunte si desea introducir los datos en la base de datos.

7.2 Línea de comandos

Una vez que se tiene el .jar correspondiente al proyecto, a través de la consola, el usuario solamente necesitará introducir la siguiente orden:

```
java -jar extractor [INPUT_FILE] [FORMAT_INPUT_FILE] [OUTPUT_FILE]
```

Haciendo corresponder los parámetros de la siguiente manera:

- **[INPUT_FILE]**: Que será la ruta donde está el archivo que se desea convertir.
- **[FORMAT_INPUT_FILE]**: Será el formato del archivo que se desea convertir, en este caso, GIFT o XML.
- **[OUTPUT_FILE]**: Ruta del directorio donde queremos situar el archivo con formato JSON que hemos obtenido.

Si se quiere realizar la inserción en la base de datos, es necesario que el usuario tenga una instancia de mongodb activa en el momento del lanzamiento de la aplicación, además de la confirmación cuando el programa le pregunte si desea introducir los datos en la base de datos.