1. Planteamiento del problema

La empresa NoGame dedicada a la creación de videojuegos va a crear una nueva línea de productos basados en juegos de preguntas/respuesta para diversas plataformas. En primer lugar se desea construir una variante del juego Trivial, aunque la idea es que puedan crearse más juegos de preguntas/respuesta similares en el futuro.

En una primera fase, desean hacerse con una base de preguntas suficiente para poder alimentar los diferentes juegos. Para ello, desean construir una sencilla aplicación que lea ficheros con los enunciados y respuestas de las preguntas, procese dichas preguntas indicando si hay errores o son correctas y las almacene en una base de datos.

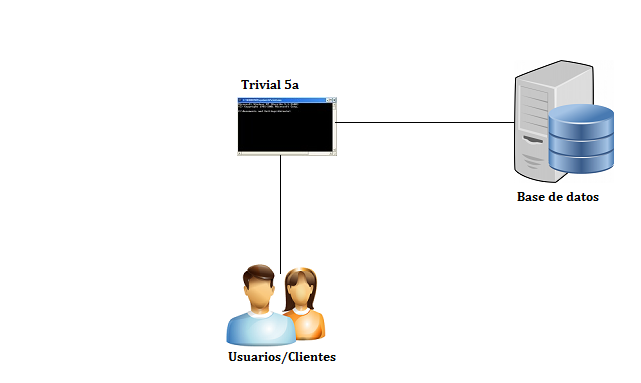
Los ficheros con los enunciados de las preguntas se tomarán de bancos de preguntas ya existentes. Uno de los bancos de preguntas utiliza el formato GIFT. La aplicación utilizará una representación interna de las preguntas en formato JSON que facilitará su posterior almacenamiento en la base de datos. La compañía está considerando utilizar MongoDb, aunque esta decisión todavía no se ha tomado.

Se desea que la aplicación pueda ejecutarse en dos etapas. Una primera etapa que analiza los ficheros con las preguntas y genera el formato intermedio y otra etapa que toma las preguntas en dicho formato intermedio y las almacena en la base de datos. La ejecución de estas 2 etapas será controlada por un operador de la compañía que podrá decidir cuándo ejecuta cada etapa e incluso podrá automatizar dicha ejecución para que se realice cada cierto tiempo.

Es importante garantizar que la conversión es correcta, facilitando la depuración del proceso de carga de preguntas. Por ese motivo, se buscará una solución que no requiera interfaz interactivo pero que permita varias opciones de entrada (elegir nombre del fichero a cargar, elegir formato de la entrada, elegir nombre del fichero de salida, elegir formato del fichero de salida, etc.) y que permita observar los resultados intermedios de la conversión para chequear que no hay errores.

Puesto que la conversión puede automatizarse para que se realice en cualquier momento, la compañía no requiere que las aplicaciones sean muy eficientes ni que el proceso de conversión se realice de forma interactiva. En la primera fase la compañía solamente requiere la conversión de preguntas en formato GIFT, pero se está pensando que la solución debe admitir otros formatos en el futuro. Por ejemplo, la compañía está pensando procesar también preguntas en formatos XML como el formato QTI.

1.1 Diagrama contextual

**

2. Metodología usada

Se va a realizar un estudio de arquitectura siguiendo el método de ADD.

ADD es un método de diseño de arquitectura de modelo Top-don, y por lo tanto se basa en la descomposición de procesos en los atributos de calidad que el software ha de cumplir.

Por tanto el primer paso es identificar todos los atributos de calidad del sistema afectados por los requisitos no funcionales y modelarlos como escenarios de calidad.

A partir de estos escenarios y siguiendo las iteraciones del proceso ABC, se va refinando la arquitectura hasta tener una solución madura. Estas iteraciones son:

* + Creación del modelo de negocio para el sistema
  + Comprensión de los requisitos
  + Creación o Selección de la arquitectura
  + Documentar y comunicar la arquitectura
  + Análisis o evaluación de la arquitectura
  + Implementación del sistema basado en la arquitectura
  + Aseguramiento de que la aplicación se ajusta a la arquitectura

Adicionalmente, y principalmente al principio del proceso, se pueden crear diagramas contextuales u otros elementos que servirán para comunicar con los stakeholders en primera instancia. Y otros diagramas que se consideren necesarios.

En un primer momento aparecerán las vistas estructurales (estáticas) y posteriormente se le añadirán vistas dinámicas para añadir información a las anteriores.

Los pasos a seguir por tanto serán:

1. Elegir un módulo a descomponer

2. Refinar el módulo

3. Repetir estos pasos hasta que no se necesario una descomposición más profunda

3. Stakeholders

* **Responsables de la empresa NoGame**

Se trata de los equipos directivos de la corporación, son responsables de los presupuestos y toman las decisiones que comprometen fondos de dicho presupuesto.

Objetivos:

- Bajo coste en el desarrollo de la aplicación

* **Equipo de desarrollo del proyecto**

Equipo responsable de desarrollar el sistema resultante de la arquitectura

Objetivos:

- Proyecto rentable, que permita ser desarrollado con el precio establecido y un grado de rentabilidad que haga atractivo el desarrollo.

- Proyecto portable, que permita el cambio de plataforma sin que esto conlleve la modificación de la mayoría del código.

* **Usuarios/Cliente de la aplicación**

Personas que utilizarán la aplicación en tiempo de explotación

Objetivos:

- Aplicación intuitiva, con una baja curva de aprendizaje

- Bajo coste en la adquisición del producto

3.1 Lista de los Stakeholders

Así pues, la lista de Stakeholders (interesados) queda:

|  |  |  |
| --- | --- | --- |
| **Código** | **Stakeholder** | **Intereses** |
| **ST-01** | Responsables de la empresa NoGame | Bajo coste en el desarrollo de la aplicación |
| **ST-02** | Equipo de desarrollo del proyecto | Proyecto rentable, que permita ser desarrollado con el precio establecido y un grado de rentabilidad que haga atractivo el desarrollo.  Proyecto portable, que permita el cambio de plataforma sin que esto conlleve la modificación de la mayoría del código |
| **ST-03** | Usuarios/Cliente de la aplicación | Aplicación intuitiva, con una baja curva de aprendizaje  Bajo coste en la adquisición del producto |

4. Requisitos Funcionales:

* Facilidad para añadir nuevos algoritmos al código, puede ser necesario en el futuro.
* Escalabilidad de la aplicación pues en el futuro puede ser necesario modificarla.
* Seguridad de acceso a los datos almacenados.
* Facilidad para probar el correcto funcionamiento del sistema.
* Facilidad de uso por parte del usuario.
* Tener un coste de desarrollo de la aplicación que no sea muy alto.
* Realizarla en un espacio de tiempo no demasiado extenso.
* Facilidad de la aplicación para adaptarse a los distintos SO.
* Buen rendimiento bajo una razonable carga de trabajo
* Rapidez a la hora de acceder a los datos.
* No mostrar las partes internas de la aplicación, el usuario solo tiene que poder ver lo que nosotros estimemos coherente y oportuno para la seguridad de la aplicación.

5. Requisitos No Funcionales (Atributos de calidad):

* Disponibilidad

Completa ya que el operador puede programar la ejecución de las dos etapas (analizar preguntas y guardarlas en la BBDD).

* **Modificabilidad**

Facilidad a la hora de añadir nuevas funcionalidades a la aplicación así como nuevos formatos a los que convertir el archivo GIFT o añadir nuevos algoritmos de procesado de datos.

Flexibilidad a la hora de admitir nuevos formatos de entrada.

Se prevé que la BBDD de preguntas se usara en varios juegos de la compañía.

* **Rendimiento**.

No es crítico, únicamente procesar las preguntas en tiempos razonables y sin necesidad de hardware potente.

* **Seguridad**.

Vital la correcta conversión de las preguntas.

Confianza en la integridad de la base de datos.

No hay login ni sistema de autenticación de momento.

* **Testabilidad**

Tiene que ser fácil probar el funcionamiento del sistema y que los resultados son los que deben ser.

Facilitar depuración del proceso de carga de preguntas.

Permitir la observación de los resultados intermedios.

* **Usabilidad**.

Facilidad para su uso por parte de los usuarios.

UI por consola, sencilla y fácil de usar por el operador.

5.1 Lista de atributos de calidad:

En función de los trabajos anteriores se identifica la siguiente lista de atributos de calidad, en la que aparecen nuevos atributos influenciados por los Responsables de CSAL (AT009 y AT010):

|  |  |  |
| --- | --- | --- |
| **Código** | **Descripción** | **Tipo de atributo** |
| **AT001** | Facilidad para admitir nuevos formatos de entrada | Modificabilidad |
| **AT002** | Modularización del sistema, se empleara en varios juegos de preguntas | Modificabilidad |
| **AT003** | Integridad de los datos | Seguridad |
| **AT004** | Correcta conversión de los ficheros de preguntas | Seguridad |
| **AT005** | Facilidad para comprobar la fiabilidad del sistema de conversión de preguntas | Testabilidad |
| **AT006** | Uso medido de recursos del sistema | Rendimiento |
| **AT007** | Facilidad de uso por el operador | Usabilidad |
| **AT008** | Etapas independientes y automatizables | Disponibilidad |
| **AT009** | Mantenible por los desarrolladores | Mantenibilidad |
| **AT010** | Tiempo de desarrollo muy bajo | Time to market |
| **AT011** | Coste de desarrollo muy bajo | Coste-Beneficio |
| **AT012** | Escalabilidad de la aplicación. | Modificabilidad |
| **AT013** | Facilidad para añadir nuevos algoritmos al código | Modificabilidad |
| **AT014** | Facilidad de uso por parte del usuario/cliente | Usabilidad |

5.2 Atributos de calidad e Interesados:

Los diferentes atributos de calidad son de interés para alguno de los Stakeholders. La siguiente tabla muestra la lista de intereses para el proyecto actual:

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributos vs interesados** | **ST-01** | **ST-02** | **ST-03** |
| **AT001** |  | X |  |
| **AT002** |  | X |  |
| **AT003** | X | X | X |
| **AT004** |  | X |  |
| **AT005** |  | X |  |
| **AT006** |  | X |  |
| **AT007** |  | X |  |
| **AT008** |  | X |  |
| **AT009** |  | X |  |
| **AT010** | X | X |  |
| **AT011** | X | X |  |
| **AT012** |  | X |  |
| **AT013** |  | X |  |
| **AT014** |  |  | X |

5.3 Escenarios de calidad:

Con toda la información anterior se procederá a definir los escenarios de calidad que influencian esta arquitectura.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Nº Escenario** | **Fuente de estimulo** | **Estimulo** | **Entorno** | **Artefacto** | **Respuesta** | **Medición de la respuesta** | **Atributo de calidad afectado** |
| **1** | Desarrollador | Nuevo formato de entrada | Desarrollo | Parser | Convertir del formato de entrada a JSON | Tiempo de desarrollo 1 dia | AT001 |
| **2** | Desarrollador | Nueva funcionalidad | Desarrollo | Sistema | Añadir la funcionalidad | No tener que reescribir código | AT002 |
| **3** | Usuario | Acceso a datos | Explotación | BBDD | Devolver datos | Sin fallo de integridad ni perdida de información | AT003 |
| **4** | Usuario | Etapa 1 | Explotación | Parser | Convertir fichero de preguntas a JSON | 0 fallos en la conversión de formato | AT004 |
| **5** | Desarrollador | Debug | Desarrollo | Sistema | Ejecutar pruebas | Pruebas unitarias correctas | AT005 |
| **6** | Aplicación | Uso estándar | Explotación | Equipo local | Funcionamiento fluido | Uso de CPU y RAM <50% del equipo | AT006 |
| **7** | Consola | Uso de la UI | Explotación | UI | El usuario no precisa de ayuda externa | Asistencia requerida: 0 | AT007 |
| **8** | Usuario |  | Explotación | Sistema | Ejecución independiente |  | AT008 |
| **9** | Desarrollador | Mantener aplicación | Desarrollo | Sistema | Código mantenible |  | AT009 |
| **10** | Desarrollador | Nuevo desarrollador | Desarrollo | Sistema | Código legible | Tiempo de adaptación <2 horas | AT009 |
| **11** | Sistema en desarrollo | Fin del desarrollo | Desarrollo | Sistema | Bajo tiempo de desarrollo | < 15 días | AT010 |
| **12** | Sistema en desarrollo | Fin del desarrollo | Desarrollo | Sistema | Bajo costes de desarrollo | 0€ | AT011 |