¿Dónde invierto?

Segunda Parte (Segundo Cuatrimestre)

2.1 Cuarta Entrega - Persistencia

Hasta el momento persistíamos los datos en archivos. El objetivo de esta entrega consiste en adaptar esta arquitectura para centralizar el almacenamiento. Es decir, necesitamos permitir almacenar los elementos del dominio en un medio relacional, no siendo necesario mantener la persistencia en archivos. La única excepción serán las cuentas, que por ahora seguiremos cargando tanto desde una base de datos como desde un archivo.

Alternativas: algunos motores de bases de datos recomendados son los siguientes:

- MySQL
- PostgreSQL

Dado que son de código abierto, maduros y cuentan con excelente soporte a lo largo de una gran variedad de lenguajes, frameworks y bibliotecas. Otros motores maduros pero de código propietario son:

- SQLServer
- Oracle

Además, no queremos acoplarnos a una base de datos particular, y para poder reutilizar el modelo de objetos ya construido, se pide el uso de un ORM. Si bien parece fácil, ¡cuidado!, puede que haya que realizar modificaciones al modelo original.

Sugerencia: De utilizar Java 8 o posterior, recomendamos utilizar JPA/Hibernate (es decir, Hibernate a través de la interfaz estándar de JPA). Un buen punto de partida para adaptar el proyecto es el siguiente: https://github.com/dds-utn/jpa-proof-of-concept-template.

Se pide:

1. Introducir en la solución un motor de base de datos relacional
2. Introducir en la solución un motor ORM
3. Persistir en dicha base de datos las siguientes entidades:
    a. Cuentas
    b. Indicadores
    c. Metodologías
    d. Empresas
4. Lograr que la aplicación siga funcionando sin cambios funcionales
5. Lograr que la aplicación pueda seguir cargando cuentas desde archivos. No es necesario desarrollar nuevas pantallas, sino sólo continuar soportando esta carga programáticamente.

2.2 Quinta Entrega - UI Web

¡Llegó el turno de implementar las nuevas vistas! Para esta entrega, nuestro requerimiento es simple: necesitaremos implementar las pantallas Web necesarias para todas las funcionalidades existentes:

- Visualización de cuentas
- Creación de indicadores
- Evaluación de indicadores
- Creación de metodologías
- Evaluación de metodologías

Sugerencia: Spark (http://sparkjava.com/) es un framework muy simple para desarrollar aplicaciones Web en Java, que si bien no presenta funcionalidades avanzadas, resulta suficiente para esta entrega.

Por último, hay que tener en cuenta que ahora que el almacenamiento estará centralizado y la aplicación será accedida por múltiples analistas, las metodologías e indicadores serán propias de cada usuario. Por ejemplo:

- si Hernán crea un indicador A y B;
- y Anabel crea indicadores C y D;

Entonces Hernán sólo verá los indicadores A y B, y Anabel, C y D.

Sugerencia: En el marco de la presentación Web, para construir componentes visuales consistentes es necesario un grado considerable de experiencia en maquetado con HTML y CSS que excede los tiempos acotados del TP Anual. Por eso, recomendamos ampliamente el uso de frameworks como Bootstrap (http://getbootstrap.com/), que proveen componentes visuales y sistemas disposición de componentes que cubren los requerimientos planteados.

Se pide:

1. Reimplementar las vistas empleando una interfaz Web. No es necesario mantener compatibilidad con las interfaces de escritorio originales, que se podrán descartar.
2. Realizar los cambios arquitecturales necesarios para que la aplicación pueda ser servida a través a través de HTTP
3. Incorporar a la aplicación el concepto de login. **No** es necesario implementar:
    a. un flujo de registración, sino que bastará con que sea posible cargar los usuarios a través de la base de datos;
    b. una política segura de control de contraseñas
    c. login con redes sociales
4. Describir mediante diagramas y prosa la arquitectura física y lógica del nuevo sistema, haciendo foco en los siguientes aspectos:
    a. Nodos de red desplegados
    b. Motores de persistencia
    c. Interfaces con sistemas internos y externos
    d. Protocolos de red utilizados
    e. Principales componentes lógicos de alto nivel

    Indicar además qué acciones son necesarias para escalar horizontalmente a la nueva arquitectura.