Enunciado dos Requisitos

LAPR4
Semana/Iteração 1

(2012/13)

atb@isep.ipp.pt
v1.2.1 (2013-05-19)

Para todas as áreas funcionais / trilhos

Documentos técnicos necessários (que devem ser incluídos no relatório técnico)

- Casos de uso ou "user stories" (http://en.wikipedia.org/wiki/User_story);
- Diagramas de sequência ilustrando o "setup" da funcionalidade (i.e., extensão);
- Diagramas de sequência para ilustrar os *use case realization* de "análise"
- Diagramas de sequência para ilustrar de que forma os casos de uso/*user stories* vão ser realizados (design);
- Diagramas de classes ilustrando as novas classes e como elas "integram" com as classes existentes.
- Documentação sobre as classes (javadoc)
- Documentação sobre testes unitários (javadoc).

Nota sobre a solução: As soluções devem seguir a arquitectura e padrões iniciais da aplicação. Em particular devem-se usar os ***pontos de extensão*** que já existem para acrescentar funcionalidades. Alterações estruturais devem ser devidamente justificadas na documentação técnica.

Notas sobre a avaliação:

- A nota de um aluno é maximizada pelo número diferente de áreas funcionais em que trabalhou. Assim, o ideal é que um aluno trabalhe em 3 áreas funcionais diferentes, uma por cada semana. Caso não seja possível deve tentar trabalhar em 3 trilhos diferentes.
- Deve ser clara a associação entre trilho/iteração e issue (um issue deve corresponder a apenas um trilho/iteração, mas pode haver mais do que um issue para um trilho/iteração).

Notas sobre os requisitos:

- Os requisitos são apresentados divididos em trilhos. Cada aluno deve escolher o trilho que será da sua responsabilidade nessa semana. Essa tarefa faz parte do planeamento (que devem fazer no início da semana com o apoio dos docentes de Gestão). Como resultado do planeamento espera-se que o grupo registe no Bitbucket os issues relativos ao planeamento semanal.
- Os requisitos podem ser, "naturalmente", incompletos e vagos. Para além disso os requisitos vão ser apresentados iterativamente, no início de cada semana. Os alunos devem usar os fóruns do moodle para tirarem dúvidas assim com podem consultar o regente (atb@isep.ipp.pt) nas suas aulas de LAPR4.

Área funcional: Persistência e I/O (EAPLI)

Trilho 1: *"Permitir importar e exportar dados, em particular de bases de dados, para facilitar a integração com aplicações terceiras."*

Iteração/Semana 1

Pretende-se que seja possível exportar o conteúdo de uma folha para uma tabela de uma base de dados.

De preferência deve existir uma nova opção de menu para efectuar esta funcionalidade. O utilizador deve indicar a área da folha que pretende exportar. Deve seleccionar a base de dados destino e, para essa base de dados, qual o nome da tabela que deve ser criada para receber os dados. A primeira linha de células a exportar indica o nome das colunas da nova tabela.

A aplicação deve suportar os mais variados drivers de JDBC de forma a ser possível exportar para diversos SGBDs relacionais. Em particular existe a necessidade imediata de suportar o HSQLDB (http://hsqldb.org/) que deve ser o SGBD sugerido por omissão.

Não deve ser usado nenhum mecanismo de ORM (Object-Relaional Mapping) para este requisito.

Por uma questão de desempenho o processo de exportação dos dados deve ser executado paralelamente usando para isso uma nova *thread*.

Trilho 2: *"Permitir novos formatos de persistência e armazenamento de dados como, por exemplo, o XML".*

Iteração/Semana 1

Definir um formato XML para persistência das folhas.

A persistência no novo formato XML deve ser tão completa quanto a persistência do formato original CLS. Por exemplo, pretende-se que este formato consiga armazenar também as propriedades que as extensões acrescentam às células.

Deve existir um *schema* (xsd) para descrever a estrutura do xml. Este deve ser usado para validar a abertura de folhas de cálculo no formato XML.

Devem ser usadas apenas as API de xml incluídas na plataforma Java 2 (javax.xml.*, org.w3c.dom.* e org.xml.sax.*). Não devem usar classes que gerem a representação dos objectos java em xml de forma automática (como por exemplo a classe XMLEnconder).

Área Funcional: Linguagens (LPROG)

Trilho 3: *"Incorporar novas funcionalidades na linguagem das fórmulas. Poderá ser necessário definir novas linguagens de fórmulas"*

Iteração/Semana 1

Pretende-se desenvolver uma linguagem nova de fórmulas.

As expressões (fórmulas) actuais que se podem executar numa célula começam pelo caractere '=' e os seus nomes são em inglês. Pretende-se implementar uma nova linguagem de fórmulas que inicie pelo caractere "#". A nova linguagem deve suportar as mesmas "funcionalidades" da linguagem actual.

Para além disso pretende-se para já a seguinte nova funcionalidade (para a nova linguagem):
- Pretende-se possibilitar a expressão de atribuição de valores a células. Por exemplo:
A2:=sum(A3:A10)
deve fazer o "sum" e atribuir o resultado à célula A2 sendo que o valor deve ser também o resultado da expressão (que é normalmente atribuído à célula actual). A sequência ':=' identifica uma atribuição.
- A nova linguagem deve ter um ficheiro de gramática próprio.

Deve-se seguir a abordagem actual do cleansheets. O código fonte da célula é "compilado" gerando uma árvore de parse (AST). A árvore de parse é depois percorrida resultando numa instância de uma classe que implementa a interface Expression. A interface Expression contempla o método "evaluate" que é executado para avaliar a "fórmula" e obter o seu resultado.

Nota:

- O cleansheets utiliza o ANTLR para gerar *parsers*. As gramáticas são descritas em ficheiros .g que dão origem a código java que implementa o parser.

Trilho 4: *"Facilitar a customização do cleansheets pelos utilizadores finais. Uma linguagem de scripting pode ser necessária."*

Iteração/Semana 1

Deve-se iniciar o suporte a uma linguagem de macros.

A ideia é que as macros tenham um nome e sejam constituídas por uma sequência de instruções. Neste momento as instruções devem ser baseadas nas expressões que são possíveis de usar nas formulas das células.

Deve existir uma opção de menu que permite aceder a uma janela de macros. Essa janela deve permitir editar e executar macros. Por uma questão de desempenho o processo de execução de macro deve ser executado paralelamente (usando para isso uma nova *thread*).

Para já pretende-se que as macros suportem:

- as expressões que são possíveis de usar nas fórmulas das células;
- atribuir o resultado das expressões a células.
- Exemplo de macro:

```
A1:=sum(A1:A11)
A2:=if(A1>10; "grande"; "pequeno")
```

Deve-se reutilizar o código já existente para as fórmulas mas expandido para ter a noção de "programa". Neste momento um programa é uma sequência de expressões (tipicamente uma por linha).

A nova funcionalidade de macros deve ser implementada numa nova extensão. Essa extensão deve acrescentar um "side bar" que inclua um mini editor de texto para o código da macro e um botão para executar a macro.

Área Funcional: Partilha de Folha (SCOMP e RCOMP)
Trilho 5: *"Permitir a edição partilhada de folhas de cálculo. A ideia é que instâncias diferentes do cleansheets possam partilhar parte do seu conteúdo"*

Iteração/Semana 1
Deve ser possível definir áreas de folhas de cálculo para partilha por outras instâncias do cleansheets.

A ideia é que uma instância do cleansheets funcione como origem da partilha e outra instância funcione como destino da partilha.

Na instância que funciona como origem da partilha o processo passa por definir uma porta para o serviço de partilha e qual a área da folha a partilhar.

Na instância que funciona como destino da partilha o processo implica introduzir o endereço e a porta da origem e indicar o canto superior esquerdo a partir do qual se deseja receber o conteúdo partilhado. Após conexão bem sucedida a instância destino recebe o conteúdo da área partilhada na origem e "apresenta" esse conteúdo a partir da célula que foi indicada.

A partilha de folha deve ser disponibilizada através do mecanismo de extensões do cleansheets. Deve existir uma "sidebar" (janela) com uma área que permita especificar os dados da partilha e outra área que permita especificar os dados da conexão a uma partilha.

A solução deve ser baseada numa arquitetura peer-to-peer de sockets em java. Devem também ser usadas threads para garantir o processamento paralelo das "tarefas" de partilha.