Inspeção de código

- Resumo -

Tester: Sónia Araújo

Developer: Abdul Karim

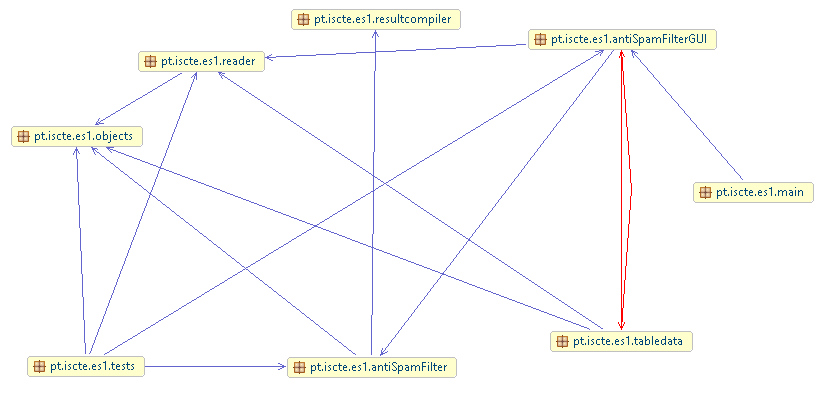
Developer: Pedro Carneiro

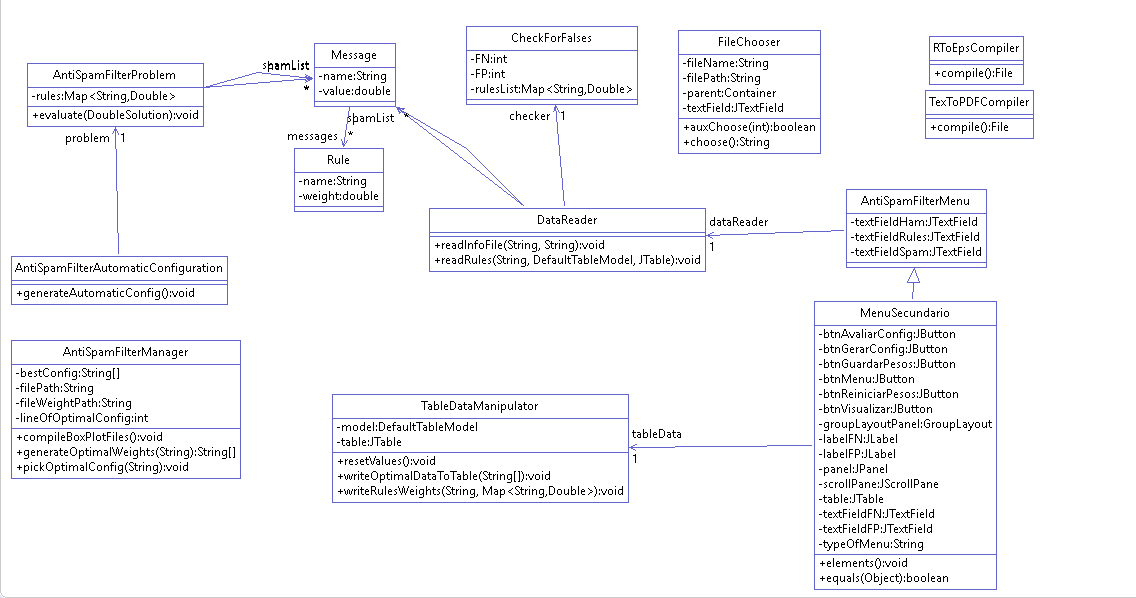
Scrum master: Mário Dantas José

Engenharia de Software I

21/12/2017

De forma a garantir um *software* que garantisse uma resposta adequada aos requisitos efetuou-se a verificação dos pacotes e classes presentes no mesmo. Esta verificação decorreu em diversos momentos, excluindo as classes *Rules, Message, RToEpsCompiler* e *TexToPDFCompiler*; nestas classes não foram aplicadas *checklists* na íntegra e de forma exaustiva por se tratarem de classes pequenas e simples. De realçar que nestas classes os comentários foram adicionados.





Verificou-se que, no inicio do projeto, a falta comentários e recurso a *JavaDoc* em todas as classes, contribuindo para uma dificuldade acrescida para os *developers* e *tester*. Após a abordagem desta mesma falha no primeiro *sprint review*, verificou-se um crescente cuidado a este nível na equipa.

A última coisa a ser feita, que embora nas primeiras CheckLists ainda não estava feito foi a cobertura do código do projeto feita através de testes unitários que neste momento já estão devidamente elaborados.

Relativamente às inspeções de código por classe:

**Package : pt.iscte.es1.antiSpamFilterGUI**

* *AntiSpamFilterMenu*

Na primeira versão:

* Ausência de comentários;
* Ausência de testes unitários;

Na versão seguinte:

* Comentários foram adicionados;
* Testes unitários foram realizados;
* *MenuSecundario*

Na primeira versão:

* Ausência de comentários;
* Ausência de testes unitários
* Métodos de elevada dimensão;
* Era a única classe com o nome em português (as restantes em inglês);

Na versão seguinte:

* Comentários foram adicionados;
* Testes unitários foram realizados;

**Package : pt.iscte.es1.tabledata**

* *TableDataManipulator*

Na primeira versão:

* Ausência de comentários;
* Ausência de testes unitários;

Na versão seguinte:

* Comentários foram adicionados
* Testes unitários foram realizados

**Package : pt.iscte.es1.reader**

* *CheckForFalses*

Na primeira versão:

* Ausência de comentários;
* Ausência de testes unitários;
* A estrutura de dados *rulesList* (*ArrayList*) não esta a ser utilizada a não ser no construtor da classe;

Na versão seguinte:

* Comentários foram adicionados;
* Testes unitários foram realizados;
* A estrutura de dados *rulesList* (*ArrayList*) foi removida;
* *DataReader*

Na primeira versão:

* Falta de comentários bem estruturados, tem comentários de reduzida dimensão;
* Ausência de testes unitários;
* As variáveis para contagem de FP e FN não estavam a ser utilizadas;
* Ausência de testes unitários;

Na versão seguinte:

* Comentários foram adicionados e melhorados;
* Testes unitários foram construídos;
* As variáveis inutilizadas foram removidas;
* *FileChooser*

Na primeira versão:

* Ausência de comentários;
* Ausência de testes unitários;

Na versão seguinte:

* Comentários foram adicionados;
* Testes unitários foram realizados;

**Package : pt.iscte.es1.antiSpamFilter**

* *AntiSpamFilterProblem*

Na primeira versão:

* Ausência de comentários;
* Bibliotecas Java bem utilizadas;
* Classe pequena bem estruturada;
* Ausência de testes unitários;

Na versão seguinte:

* Comentários foram adicionados
* *AntiSpamFilterManager*

Na primeira versão:

* Ausência de comentários;
* Bibliotecas Java bem utilizadas;
* Classe pequena bem estruturada;
* Ausência de testes unitários;

Na versão seguinte:

* Comentários foram adicionados;

**Package : pt.iscte.es1.objects**

* *Rules*
* *Message*

**Package : pt.iscte.es1.resultcompiler**

* *RToEpsCompiler*
* *TexToPDFCompiler*

Estas classes são de dimensão reduzida pelo que não foram aplicadas *checklists* de forma exaustiva, estando as classes estão em parte comentadas.

De uma forma geral, analisando a estrutura do programa, nomeadamente ao nível dos *packages*, considerou-se as ligações existentes entre as estruturas como sendo adequadas e que estas se encontram organizadas de uma forma pertinente. O mesmo se concluiu relativamente às classes em que a hierarquia e dependências são estabelecidas de forma ajustada. Os ciclos são bem-feitos e o programa não apresenta grandes demoras ou erros nas respostas.