

Universidade do Minho

Mestrado Integrado em Engenharia Informática

3º Ano, 1º Semestre

Unidade Curricular de

Desenvolvimento de Sistemas de Software

Ano Letivo de 2018/2019

Trabalho Prático de DSS

(Desenvolvimento de uma aplicação de gestão de encomendas envolvendo stands e fábricas)

Luís Filipe da Costa Cunha (a83099) Henrique José Carvalho Faria (a82200)

André Gonçalo Castro Peixoto (a82260) Miguel Ângelo Moreira Ramos Brandão (a82349)

Abril, 2018

Índice

**Não foi encontrada nenhuma entrada de índice.**

1. Introdução

O objetivo deste trabalho é analisar e modelar uma aplicação desktop de gestão e comunicação entre um stand e uma fábrica e, posteriormente, codificar e executar esta mesma aplicação. Para isto, o software deve:

* permitir que um funcionário de stand, junto do cliente, consiga configurar e enviar uma encomenda de um novo carro para a fábrica;
* ter pacotes de configuração predefinidos e com desconto;
* propor a melhor configuração possível perante um limite máximo a gastar;
* verificar se um componente obriga a colocação de mais componentes;
* permitir guardar e retomar uma seleção de componentes pendente;
* indicar os carros a serem produzidos por ordem de chegada de configurações;
* permitir ao funcionário de fábrica atualizar o stock de componentes;

Além disto, o software não deve:

* permitir que componentes incompatíveis sejam incluidos na mesma encomenda;
* permitir que uma encomenda seja enviada sem possuir todos os componentes obrigatórios;
* permitir que uma encomenda seja enviada quando não existe um número de componentes suficiente em stock para a satisfazer;

Este relatório tem como objetivo descrever a primeira fase do nosso trabalho: a análise e modelação do sistema. Para tal, tomámos em consideração os requisitos enunciados acima e apresentamos os nossos modelos na secção seguinte.

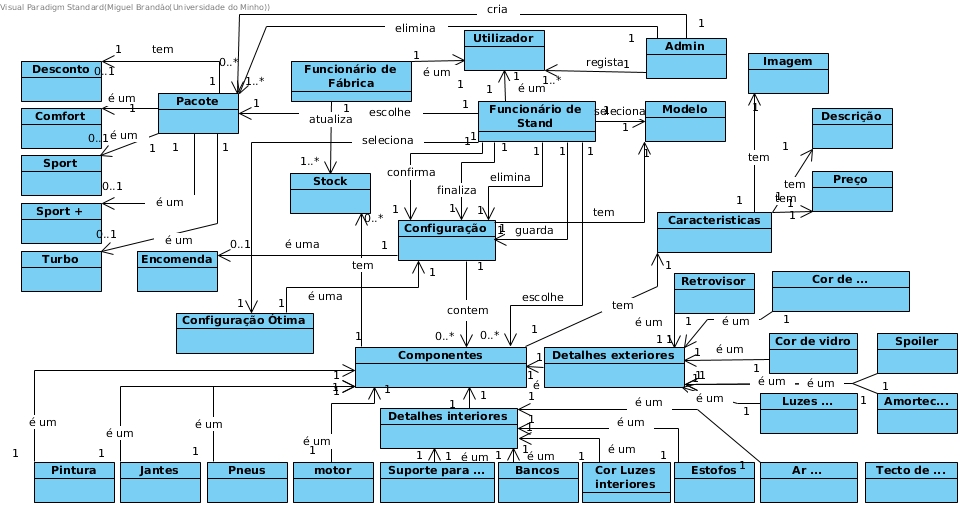
1. Análise de requisitos
   1. Modelo de Domínio

Figura 1: Modelo de Domínio

Neste trabalho temos 2 classes principais: o funcionário de fábrica e o funcionário de stand. Estes fazem ambos parte da class “utilizador”, que por sua vez são registados pelo administrador do sistema sempre que necessário adicionar um novo empregado.

O funcionário de stand trata de escolher os componentes das configurações, assim como as confirmações necessárias (se existe stock suficiente) e finalizações (envio de encomendas). Também pode fazer um “save” ou “delete” nas configurações que pretender, desde que estas não tenham sido finalizadas. Adicionalmente, o funcionário seleciona o modelo do carro a encomendar sempre que iniciar uma nova configuração, pois os componentes diferem de modelo para modelo. Para além da seleção de pacotes inteiros (com desconto), tem a possibilidade de selecionar uma “configuração ótima” que preenche, também automaticamente, os componentes que fazem o melhor uso ao limite de dinheiro do respetivo cliente.

Aos funcionários de fábrica, a aplicação permite modificar o stock de cada componente, que atualiza automaticamente no sistema para configurações futuras.

Por fim, para além de puder registar funcionários, o administrador do sistema pode também criar mais pacotes e eliminar os que já existem, conforme as necessidades da fábrica.

* 1. Modelo de Use Case

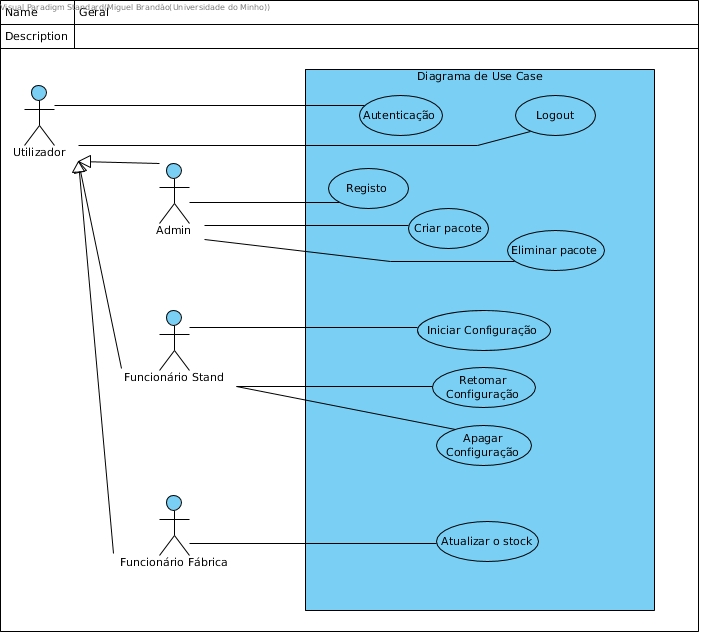


Figura 2: Diagrama de Use Case (1/2)

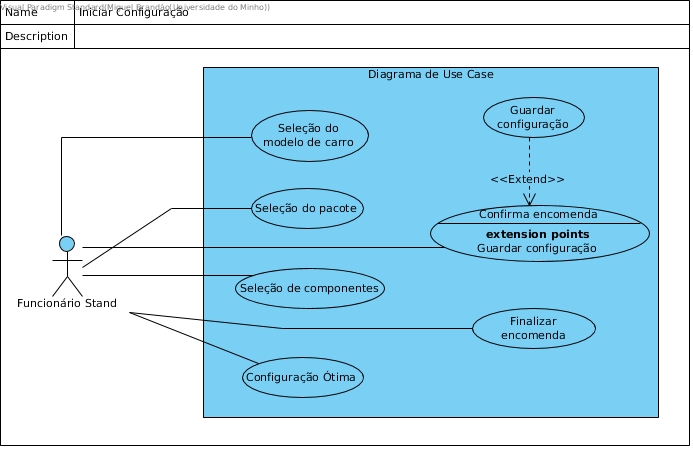


Figura 3: Diagrama de Use Case (2/2)

Com o diagrama de use cases acima, podemos observar que temos 15 use cases no total, sendo apenas 2 deles comuns a todos os utilizadores (autenticação e logout). Estes use cases têm as seguintes especificações:

* Autenticação: o utilizador entra no sistema;
* Logout: o utilizador sai do sistema;
* Registo: o administrador regista outro utilizador (quer seja funcionário de fábrica, funcionário de stand ou outro administrador);
* Criar Pacote: o administrador cria um pacote de configuração que ainda não existe no sistema;
* Eliminar Pacote: o administrador elimina um pacote de configuração existente no sistema;
* Retomar Configuração: o funcionário de stand continua a configuração que tinha guardado anteriormente;
* Apagar Configuração: o funcionário de stand apaga uma configuração guardada (se existir);
* Seleção do Modelo de Carro: o funcionário de stand seleciona o tipo de carro que o cliente pretende encomendar;
* Seleção do Pacote: o funcionário de stand seleciona um pacote de configuração disponível a encomendar;
* Configuração Ótima: o funcionário de stand insere o montante limite que o cliente está disposto a pagar e o sistema preenche automaticamente a melhor escolha;
* Seleção de Componentes: o funcionário de stand preenche manualmente cada componente para uma configuração personalizada;
* Guardar Configuração: o funcionário de stand guarda a configuração do cliente no sistema para ser retomada no futuro;
* Confirmar Encomenda: o funcionário de stand confirma que existe stock na fábrica para satisfazer a configuração do cliente;
* Finalizar Encomenda: o funcionário de stand finaliza a encomenda depois de ter sido confirmada (é enviada para a queue de encomendas na fábrica);
* Atualizar Stock: o funcionário de fábrica adiciona ou remove componentes manualmente no sistema;

Para uma análise de use cases mais aprofundada, incluímos um ficheiro Excel, onde se encontra uma observação detalhada de cada um.

* 1. Interface Gráfica

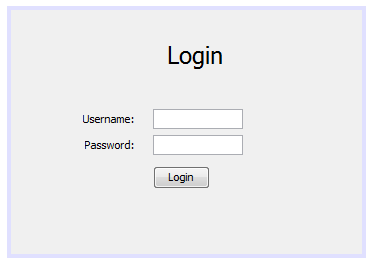
Nesta secção encontra-se a nossa proposta de interface para todo o sistema. Incluímos também os ficheiros de Netbeans fora do relatório, para possibilitar uma exposição mais direta.

Figura 4: Janela de Login



Figura 5: Lobby do Administrador



Figura 6: Registo (feito pelo admin)



Figura 7: Lobby de Funcionário de Stand

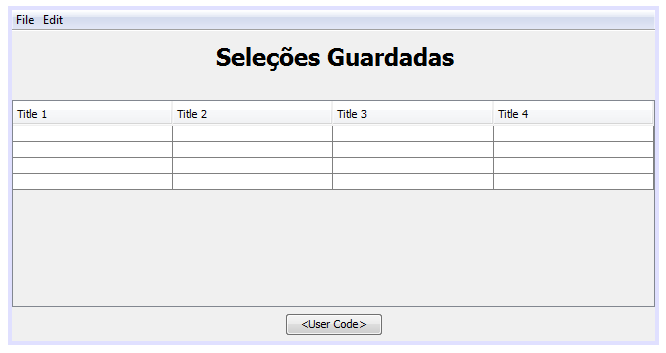


Figura 8: Janela das configurações guardadas

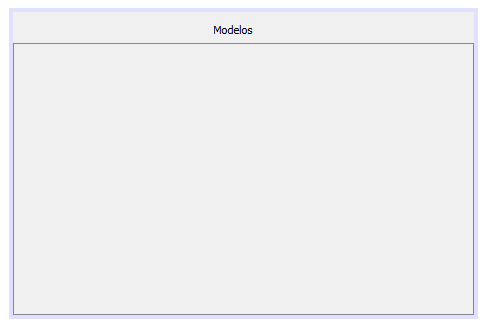


Figura 9: Primeira janela do processo de configuração (escolher modelo)

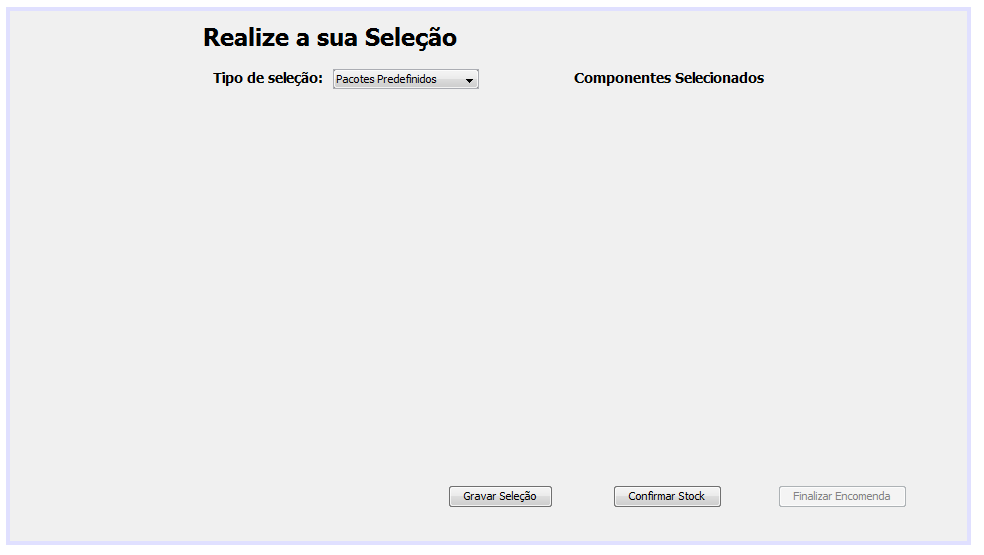
Figura 10: Segunda janela do processo de configuração (escolher o tipo de seleção/gravar/confirmar stock/finalizar encomenda)



Figura 11: Seleção de componentes manual



Figura 12: Configuração ótima automática

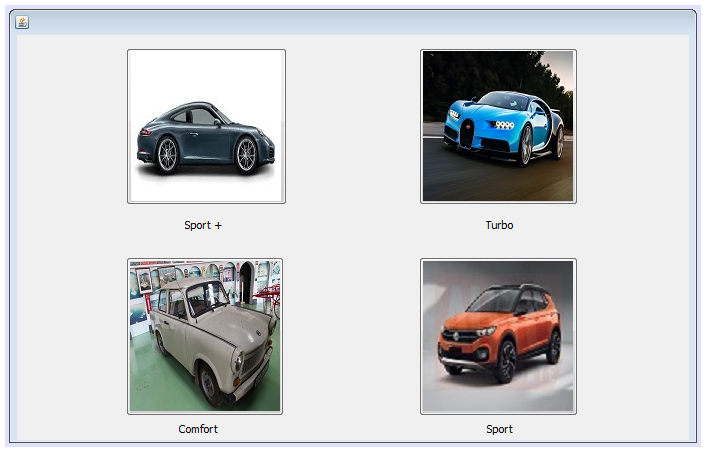


Figura 13: Seleção de pacote

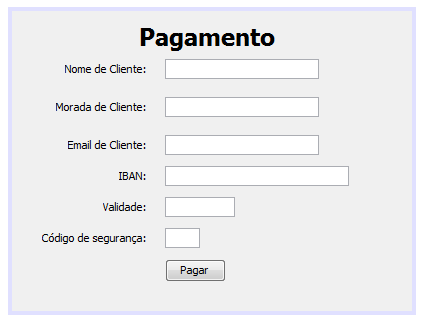


Figura 14: Finalizar configuração

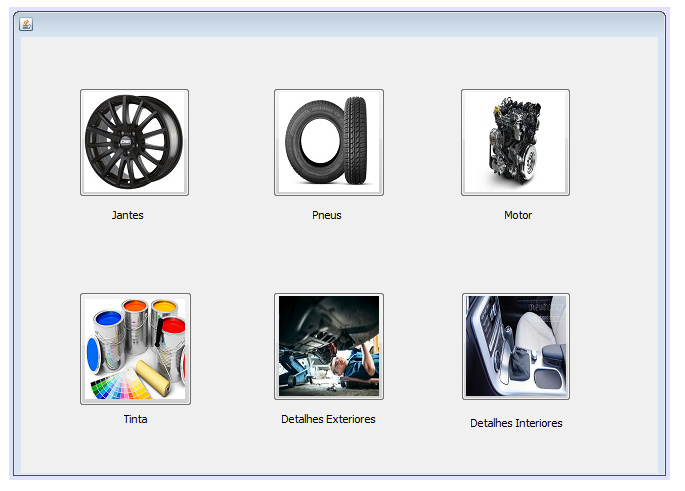


Figura 15: Lobby do Funcionário de Fábrica (escolher o que quer atualizar)

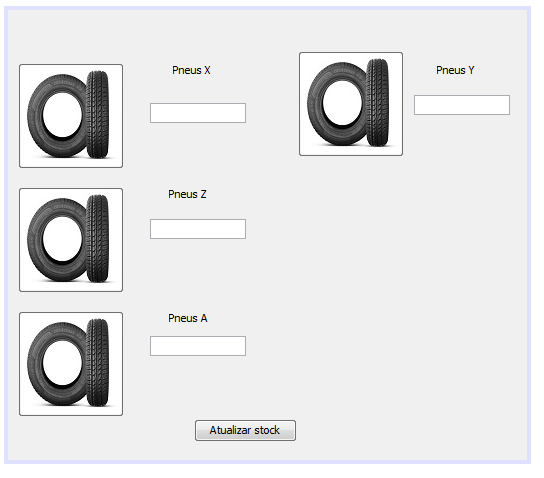
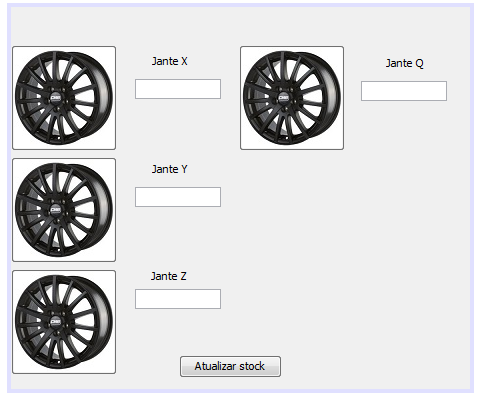
Figura 16: Atualizar stock de pneus

Figura 17: Atualizar stock de jantes

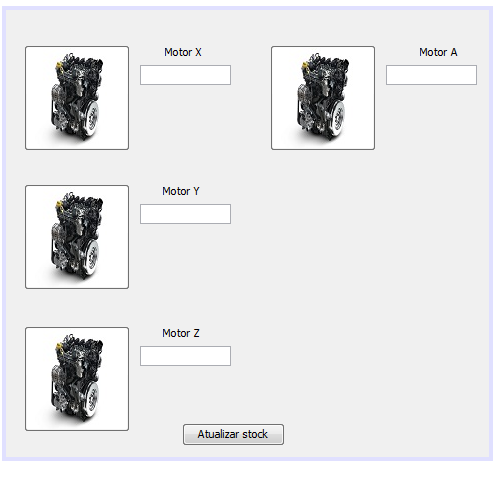
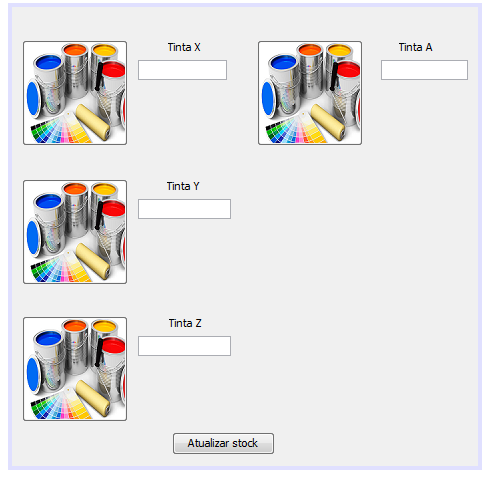
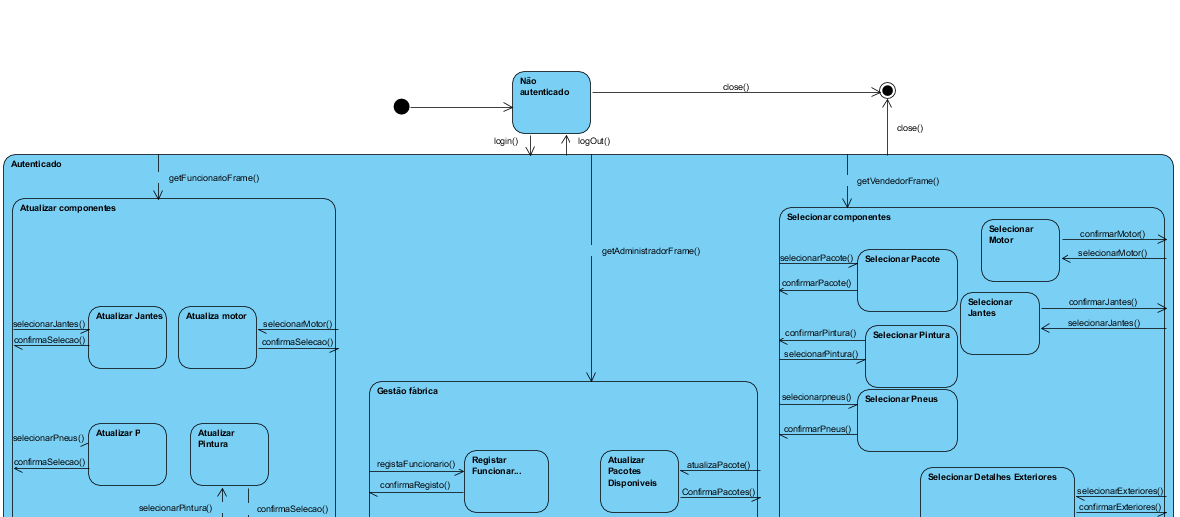
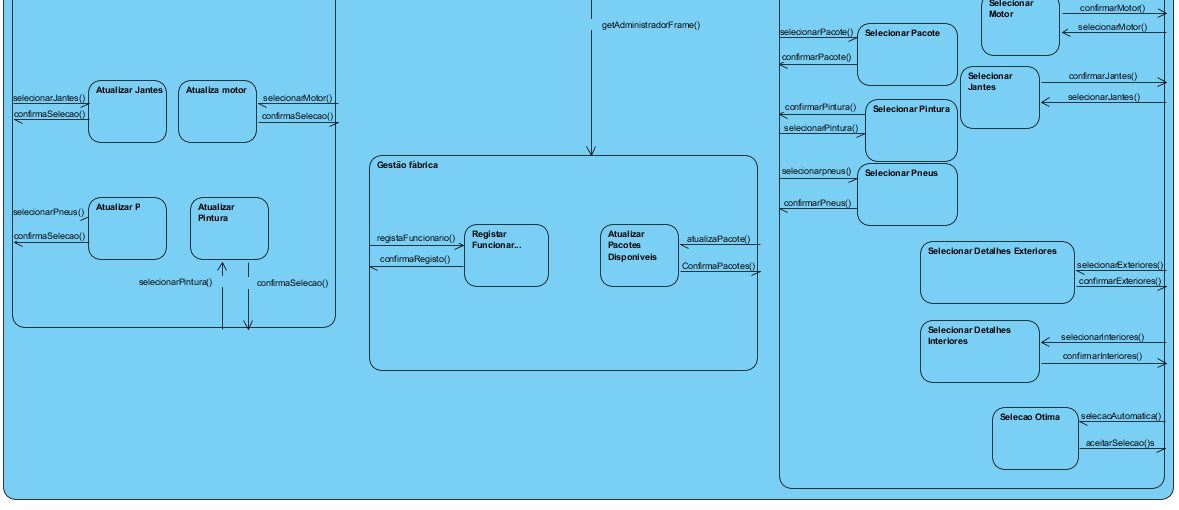
 Figura 18: Atualizar stock de motores

Figura 19: Atualizar stock de tintas

* 1. Diagrama de Estado

Por fim, apresentamos o diagrama de estado do nosso sistema. Também incluímos o ficheiro do Visual Paradigm fora do relatório, para uma visualização mais interativa.

Neste diagrama, nós admitimos que existe um superestado chamado “autenticado”, que surge quando se efetua o login. Este estado direciona o utilizador para diferentes sub-estados, dependendo da sua “role” no sistema. Se o utilizador for um funcionário de fábrica, este tem acesso aos métodos representados no lado esquerdo do modelo acima, que correspondem à atualização do stock (jantes, motores, pintura e pneus). Se o utilizador for um administrador, só tem acesso aos métodos que lhe permitem registar funcionários e criar/eliminar pacotes do sistema. Por último, todos os funcionários de stand têm acesso às funcionalidades faladas já nas secções anteriores, que estão representadas no lado direito do superestado do modelo acima.

1. Conclusão

Fazendo uma apreciação geral do resultado final, podemos concluir que, como todo o input necessário se encontra devidamente anotado, a modelação e análise do sistema (domínio, use cases e estado) facilita imenso a criação da interface da aplicação e, posteriormente, a escrita do código backend. Isto permitirá desenvolver o software da maneira mais eficaz, porque reduzimos o número de erros que poderão aparecer, o que nos diminui bastante o tempo necessário ao desenvolvimento.

Além disto, é da nossa opinião que conseguimos refletir o que era pedido no enunciado na primeira fase deste projeto. Dito isto, é de notar que esta análise e modelação pode sempre sofrer pequenas alterações no futuro, conforme o projeto avança – o que faz parte do desenvolvimento de um sistema de software. Porém, ficamos satisfeitos com a versão atual, que de momento nos parece ser a mais correta.