Universidade Federal de Minas Gerais

Trabalho Prático: Oficina de Eletrodomésticos
Engenharia de Software

Guilherme Nascimento
Marcelle Yitse
Lucas Augusto
Willer

16 de novembro de 2013

1 Introdução

O trabalho a seguir consiste na implementação de um software de gerenciamento de uma oficina de eletrodomésticos. Para a realização desse trabalho foi necessário a aplicação dos diversos conceitos aprendidos na disciplina de Engenharia de Software tais como a elaboração dos requisitos, definição do modelo de processo, arquitetura e diagramas para a compreensão do projeto tanto para o cliente como para os desenvolvedores. Por fim, todo o programa é implementado na linguagem Java e por fim submetida a testes de validação para ser entrega na data final estipulada.

O sistema de gerenciamento aqui descrito suporta atividades CRUD: Cadastro, remoção e atualização para clientes, eletrodomésticos, peças, contatos de fabricantes e os próprios funcionários utilizadores do sistema.

Este documento está dividido em 7 seções das quais temos: contextualização, modelo de processo e cronograma, especificação de requisitos, projeto arquitetural, detalhes de projeto, comentarios da implementação e por fim, conclusão.

2 Modelo de Processo

2.1 Modelo Cascata

O modelo cascata foi escolhido pela equipe devido ao nível de complexidade do sistema a ser criado não ser grande e por todos os requisitos serem bem definidos. O sistema constitui numa aplicação desktop, logo o cascata favorece nesse sentido, pois caso haja falhas no processo de elicitação de requisitos, o que acarretará uma mudança no sistema já criado, implicará na criação de uma nova versão e com isso uma nova visita ao cliente para realizar a implantação do sistema.

Como o cascata possui seus requisitos bem definidos na documentação e essa parte sendo a principal, ele garantirá a minimização de falhas durante o processo

de desenvolvimento e seguirá de forma rigorosa a documentação para atender ao máximo o objetivo do cliente.

Caso haja alguma mudança a ser feita, será realizada uma nova fase de documentação, estudo de viabilidade e análise de requisitos para depois ser feita uma nova versão e logo em seguida a sua implantação no ambiente do cliente.

2.2 Responsabilidades e Cronograma

O cronograma abaixo aborda todo o processo necessário para o desenvolvimento do software de gerenciamento, o tempo total demandado foi de 7 semanas, onde todos elementos da equipe participaram do procedimento, composto por etapas de análise de requisitos, projeto arquitetural, documentação, criação de diagramas UML e de cenários de uso, Implementação e testes unitários, fase esta onde todos elementos da equipe participam, testes de integração, documentação final e manutenção, período este que perdura enquanto o software for utilizado.

3 Especificação de Requisitos

Para fazer a especificação de requisitos foi imaginado um cenário de comunicação entre o cliente e o grupo a fim de dar detalhes do sistema que gerenciará a oficina de eletrodomésticos.

3.1 Contextualização

Com o atual auxílio da informática, automatizar sistemas e processos arcaicos significa otimizar gastos, tempo e manter uma organização sobre o seu trabalho. Uma oficina de eletrodomésticos possui muito a ganhar ao automatizar seu sistema de gerenciamento, status de produtos na oficina, controle de notas fiscais, controle das finanças da empresa e também dos funcionários facilita a vida do gerente.

A presença de um sistema que possa agregar todas essas funcionalidades citadas acima poderá aumentar a qualidade dos serviços ofertados pela oficina, otimizar o tempo gasto com tarefas burocráticas, manter organizado todo ecossistema da mesma e garantir que o empresário dono da oficina atinja e atenda mais clientes, de uma maneira otimizada e segura.

3.2 Análise de viabilidade

Com as condição estipuladas, os desenvolvedores e projetistas definiu o projeto como viável, sendo necessário não mais que 4 desenvolvedores para a implementação do software sendo que as tecnologias utilizadas serão de total liberdade de escolha dos desenvolvedores sendo comentadas aqui posteriormente.

3.3 Elicitação de Requisitos

(Conversas aqui - usar frames)

Analisando as necessidades descritas pelo cliente, foram levantados os seguintes requisitos funcionais para o sistema:

- Cadastro de clientes;

- Cadastro de funcionários;
- Cadastro de peças;
- Cadastro de fabricantes;
- Geração de orçamento;
- Geração de nota fiscal;
- Emissão de relatórios:
  - Receita
  - Peças
  - Consertos

Dados os requisitos funcionais acima descritos, foram levantados também os seguintes requisitos não-funcionais:

- Desempenho
  - A emissão dos relatórios de vendas, receita, peças e conserto não devem demorar mais do que 15s;
- Facilidade de uso:
  - As telas devem ser intuitiva sendo necessário não mais que um dia para o aprendizado de todas as funções oferecidas pelo programa;
  - Um documento de ajuda deve ser oferecido em um dos menus do software
- Robustez:
  - Reinicialização do sistema em caso de falha de no máximo 3 minutos.
  - Permitir a execução de backups periodicos a fim de não se perder o conteúdo presente no banco de dados;
- Segurança:
  - Permitir um sistema de login, em que, dependendo do tipo de conta, haverá usuários comuns e administradores.

3.4 Especificação de Requisitos

3.4.1 Cenários

(cenários aqui)

3.4.2 Casos de uso

3.5 Validação de Requisitos

Os requisitos definidos pelo cliente foram aprovados pela equipe de desenvolvimento dando então prosseguimento ao desenvolvimento do projeto.

4 Projeto Arquitetural

A arquitetura utilizado para a implementação do sistema será a de três camadas por ser a mais utilizado no conceito CRUD. Nota-se que que este tipo de arquitetura proporciona uma maior possibilidade de mudanças em camadas superiores sem afetar de fato as camadas mais abaixo. Para um sistema de cadastro, o fluxo de dados não é grande o que não compromete o desempenho do sistema.

A primeira camada do sistema é a de Apresentação, que é onde toda a parte visual de interação do usuário com o sistema. É a partir dela que o funcionário ou o administrador do sistema tem acesso às funções definidas pelo cliente em alto nível.

A segunda camada, a de controle, é a responsável por toda encapsulação de informação como também o fornecimento de operações feitas pelo sistema tais como validação de usuário, geração de relatórios e orçamentos, calculo de valores entre outros. Nesta camada teremos a transição dos dados fornecidos pelo usuário para a camada de dados, descrita a seguir.

Assim, temos a terceira camada, a de Dados que é a responsável por todo o mapeamento do modelo de dados para a aplicação. Com isso temos funções de farão todo o gerenciamento do banco de dados como também a recuperação dos dados. Abaixo é mostrado como a seguinte arquitetura de comporta:

Abaixo o segue o diagrama de componentes que constituem o modelo arquitetural de 3 camadas:

5 Detalhes do Projeto

5.1 Tecnologias utilizadas

Ambiente: toda a implementação do sistema foi feita com a IDE Netbeans 7.1.1 com o Java JDK 1.7.

Banco de dados: Para o armazenamento de dados foi utilizado o sistema de bancos de dados MySQL versão 14.14 que provê uma facilidade na comunicação entre os pacotes Java e por ser um software gratuito.

Interface Gráfica: para a construção das telas do sistema, foi utilizado o pacote Swing por ser um pacote nativo da linguagem e pela vantagem do drag and drop para a contsrução da interface.

Persistência: para a persistência de dados foi utilizado o framework Hibernate. Ele possibilita toda a comunicação dos objetos da camada de controle com o banco de dados. Para a configuração do banco, é utilizado um arquivo XML que define as flags e variáveis de inicialização com o MySQL. ¡ver aqui se precisa de classe repositório¿

Testes: para a realização de testes, foi usado a biblioteca jUnit que constitui de funções que possibilitam usar todas as funções implementadas na camada de controle e dados sem necessária dependência de uma interface pronta. Uma vez que temos toda a camada de controle e de dados funcionais fica mais fácil a implementação de uma camada de Visualização que cubra todas os requisitos exigidos pelo cliente

5.2 Diagrama de Classes

Abaixo seguem dois diagramas de classes. O primeiro é uma versão simplificada de relacionamento entre os objetos e a segunda uma versão mais detalhada com classes utilizadas pelo Hibernate na camada de dados.

5.3 Diagrama de Sequência

Abaixo seguem os principais diagramas de sequência para as principais operações de cadastro, remoção e atualização. Para abstrair de forma mais genéricas as classes envolvidas, o uso do nome Subject foi usada já que as mesmas operações são feitas para todas as classes.

5.4 Diagrama de pacotes

Abaixo segue os pacotes utilizados na implementação, tais como aqueles criados para encapsular as classes utilizadas e das bibliotecas que farão a comunicação com o banco de dados e a criação da interface.

5.5 Diagrama de atividades

Abaixo os diagramas de atividades para as principais atividades realizadas. Nota-se que foram destacadas apenas as atividades mais críticas.

6 Implementação:

Toda a implementação da parte lógica foi concluída para as funções CRUD como as de cadastro, remoção e atualização de cliente, peça, funcionário e fabricante. Funções da interface gráfica não foram totalmente completadas, porém foram detalhadas nessa documentação para fins didáticos. A conclusão de uma interface gráfica demanda de várias horas assim como a constante avaliação do cliente o que pode levar tempo para a total implementação. Para cobrir os testes foi usado a biblioteca jUnit. Desse modo podemos cobrir todos os testes necessários sem a presença de uma interface completamente pronta. Isso nos dá uma abstração dos possíveis casos de uso do sistema.

7 Conclusão

O trabalho desenvolvido para a disciplina de Engenharia de Software constituiu na implementação de um sistema para uma oficina de eletrodomésticos. Nele foram abordados todos os processos que normalmente são realizados pelas empresas para a implementação de um software e que foram aprendidos no decorrer da matéria.

A elaboração de um documento completo, com a construção de diagramas UML foi de suma importância, já que na maior parte das matérias do curso de Ciência da Computação temos noções apenas da parte da programação não nos dando um detalhamento das etapas de um sistema mais complexo como a disciplina de Engenharia de Software nos dá.

Parte principal de um software é a sua especificação junto com a análise de requisitos, pois nela que se consegue determinar o quanto será necessário se esforçar para conseguir terminar o projeto e obter um software funcionando com qualidade, aspecto hoje que é totalmente negligenciado pelas empresas. Demandas inconsequentes, clientes que não sabem o que querem junto com desenvolvedores que não trabalham a parte de especificação de um software somam um conjunto precário que resultará em um sistema ruim.

Estimar quanto se gasta e quanto vai custar um software também é um dos pontos mais complicados hoje em dia, a falta de organização das empresas e de planejamento, faz com que mensurar quanto é gasto e quanto será necessário para executar um projeto seja uma tarefa complicada.

Engenharia de software mostrou o quanto é importante o planejamento, tanto para desenvolvedores quanto para clientes, onde um bom planejamento vai gerar um bom projeto, com deadlines viáveis e um sistema bem estruturado livre de falhas e garantirá ao cliente um sistema que atenda a sua demanda, solucione o seu problema e o principal um software de qualidade.

Portanto o trabalho cumpriu satisfatoriamente seus objetivos no desenvolvimento do sistema proposto e proporcionou bastante experiência aos envolvidos.

8 Referências