UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE CIÊNCIAS E TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
COORDENAÇÃO DE ESTÁGIOS DE ENGENHARIA ELÉTRICA

RELATÓRIO FINAL DE ESTÁGIO

Análise de Testes em Aparelhos Celulares de Tecnologia GSM

Estagiário: Eliezer Farrant Braz
Matrícula: 29711116

Empresa: CIn-BTC
Orientador: Prof. Dr. Antônio Marcus Nogueira Lima

Campina Grande – PB
Junho de 2005

Biblioteca Setorial do CDSA. Fevereiro de 2021.

Sumé - PB

Agradecimentos

Gostaria de agradecer a todos os participantes do projeto CIn-BTC pelo apoio e amizade demonstrados durante todo o período de realização do estágio. Em especial aos membros da gerência que me deram a grande oportunidade de ter estagiado no projeto.

Gostaria de agradecer ao prof. Antônio Marcus pela orientação e revisão para escrever este relatório de estágio.

Gostaria de agradecer a Coordenação de Estágios do curso de Engenharia Elétrica da UFCG por todo apoio que foi oferecido durante a realização do estágio.

Gostaria de agradecer as secretárias da Coordenação do curso de Engenharia Elétrica, Adail e Rosilda, por toda ajuda e apoio que nos foi fornecida.

Gostaria de agradecer especialmente minha família e a minha namorada por todo amor e apoio.

Resumo

O Projeto CIn-BTC (Centro de Informática – *Brazilian Test Center*) é um programa de testes de software mantido e desenvolvido pelo CIn-UFPE (Centro de Informática da Universidade Federal de Pernambuco) através de uma parceria com a Motorola Industrial Ltda com os incentivos e benefícios previstos na Lei de Informática. Dentro deste projeto há o Curso Seqüencial de Formação Complementar que visa formar profissionais especializados em Engenharia de Software, especificamente na área de testes. Os alunos têm não só uma visão geral das diversas fases do processo de desenvolvimento de software, mas também oportunidade de vivenciá-las em laboratório, proporcionando contato direto com atividades de testes para desenvolvimento de software para celulares da tecnologia GSM.

Esse projeto foi inspirado no programa de residência médica, no qual alunos têm oportunidade de obter uma formação teórica na Universidade e desempenhar as atividades práticas.

As atividades realizadas durante o estágio, como participante do Curso Seqüencial de Formação Complementar consistiram basicamente de realizar testes em protótipos de celulares Motorola, bem como sugerir melhorias ao processo utilizado pelo CIn-BTC através de uma monografia.

SUMÁRIO

LISTA DE FIGURAS

1. INTRODUÇÃO

Neste relatório, são apresentadas as atividades realizadas pelo estagiário como integrante do Programa de Imersão Tecnológica. Este programa é resultado de uma parceria entre o Centro de Informática da Universidade Federal de Pernambuco (UFPE) e a Motorola Industrial Ltda, tendo como objetivo principal "incentivar a formação de recursos humanos com alto grau de especialização em *software* embarcado para testes e aplicações em computação móvel, com os incentivos e benefícios previstos na Lei de Informática" [7].

Dentro do programa, há o Curso Seqüencial de Formação Complementar, que visa capacitar os seus integrantes na área de Engenharia de Software, mais especificamente em análise de testes. Este curso é baseado no sistema de residência médica, sendo dividido entre atividades teóricas e atividades práticas.

As atividades teóricas consistem de disciplinas que abordam conceitos básicos e avançados na área de Engenharia de Software, além de um projeto que deverá ser desenvolvido pelos integrantes do curso e cujo tema deve abordar um dos conceitos visto durante o curso.

As atividades práticas variam de acordo com a área na qual o participante do curso foi alocado, podendo este ser alocado para a área de desenvolvimento de programas ou rotinas de testes, ou para a área de execuções de testes em diversos tipos de protótipos de telefone celulares da Motorola - tanto manuais como automáticos - e análise e reescrita de casos de testes. No caso específico do deste estagiário, a área na qual ele foi alocado corresponde a de execução de testes em celulares Motorola.

1.1 A Empresa

O projeto CIn-BTC (Centro de Informática – *Brazilian Test Center*) é um programa de testes de *software* mantido e desenvolvido pelo CIn-UFPE (Centro de Informática da Universidade Federal de Pernambuco) através de uma parceria com a Motorola Industrial Ltda, com os incentivos e benefícios previstos na Lei de Informática. O organograma do CIn-BTC é apresentado na Figura 1.

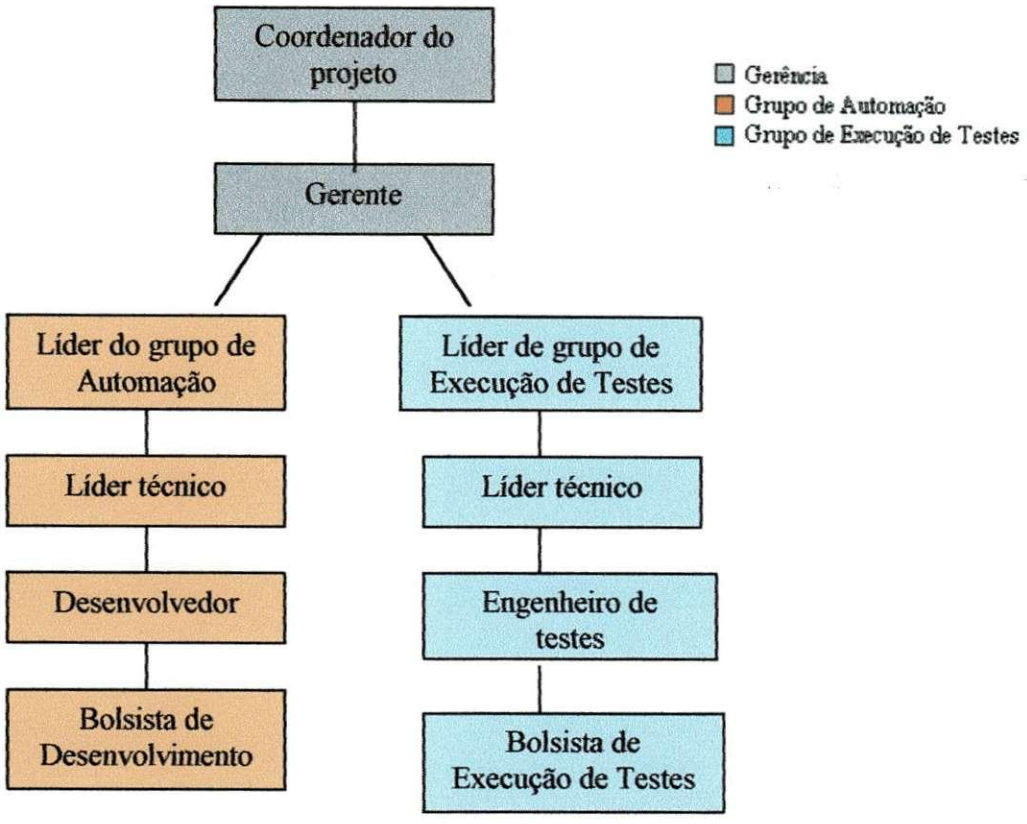


Figura 1 – Organograma do projeto CIn-BTC

A seguir, apresenta-se uma breve descrição da Motorola Industrial Ltda e do CIn-UFPE, de modo uma melhor compreensão do texto.

1.1.1 Motorola Industrial Ltda

A Motorola Industrial Ltda é líder mundial em soluções integradas de comunicação e de eletrônica. Seus investimentos no Brasil começaram em 1995, com o início da construção de um campus industrial e tecnológico localizado na cidade de Jaguariúna no estado de São Paulo. Atualmente, lá também estão localizados um centro de pesquisa e desenvolvimento de terminais celulares em hardware, *software*, mecânica e desenho industrial, um Centro de Tecnologia de Semicondutores, além de uma base da Motorola University, voltada ao treinamento de funcionários e à consultoria do programa Six Sigma para o mercado.

No Brasil, a empresa está ainda fortemente presente na comercialização de semicondutores, acesso à Internet e TV por banda larga, cable modems, sistemas

automotivos e soluções de telemática, além de desenvolver muitas outras soluções para os mercados corporativos e de comunicação pessoal.

1.1.2 CIn-UFPE

O Centro de Informática da Universidade Federal de Pernambuco (CIn-UFPE) é um dos mais conceituados centros acadêmicos de informática da América Latina. Oferece curso de bacharelado em ciência da computação, disponibilizando 100 vagas por ano. Também funciona no Centro o curso de engenharia da computação com 50 novas vagas a cada ano.

Hoje atuam no corpo docente do CIn 46 doutores. Estão matriculados no Centro cerca de 600 alunos de graduação, 123 de mestrado, 65 alunos de doutorado e 165 de especialização. A infra-estrutura do CIn conta com mais de 500 postos de trabalho interligados em Rede, distribuídos nos 15 laboratórios, nas salas de aulas e dos professores.

2. FUNDAMENTAÇÃO TÉORICA

2.1 Metodologia de Desenvolvimento de Projetos de *Software*

IBM *Rational Unified Process*®, ou RUP®, é uma plataforma de processos de desenvolvimento de *software* configurável que oferece melhores práticas comprovadas e uma arquitetura configurável. [6]

Possui as seguintes características principais:

- Iterativo incremental: o sistema é construído de forma incremental;
- Orientado a objetos;
- Guiado por casos de uso;
- A arquitetura do *software* tem papel central;
- Planejado por riscos,

Possui quatro fases, sendo elas Concepção, Elaboração, Construção e Transição. Na Figura 2, mostra-se um gráfico conhecido como *gráfico das baleias*, no qual pode ser vista a relação entre as fases e as atividades (disciplinas) do RUP.

Na fase de Concepção, um estudo de viabilidade do projeto é realizado, no qual o escopo e os objetivos do projeto são definidos. Nesta fase o projeto começa a ser modelado e os requisitos do sistema começam a ser levantados, como pode ser visto na Figura 2.

Na fase de Elaboração, a maioria dos requisitos do projeto é capturada e a arquitetura do sistema é definida e construída. Durante esta fase, protótipos são construídos para se eliminar riscos. Como pode ser visto na Figura 2, as atividades mais importantes nessa fase são as de requisitos e a de análise e projeto.

Na fase de Construção, os casos de uso criados durante a fase de Elaboração são implementados. Nesta fase, são desenvolvidas versões alfa e beta do produto (*software*) para que este possa ser testado em condições reais de operação. Como pode ser visto na Figura 2, implementação e testes são as atividades principais desta fase.

Na fase de Transição, o sistema é implantado no ambiente do cliente. Tem como atividades principais distribuição, suporte e entrega do produto, como pode ser observado na Figura 2.

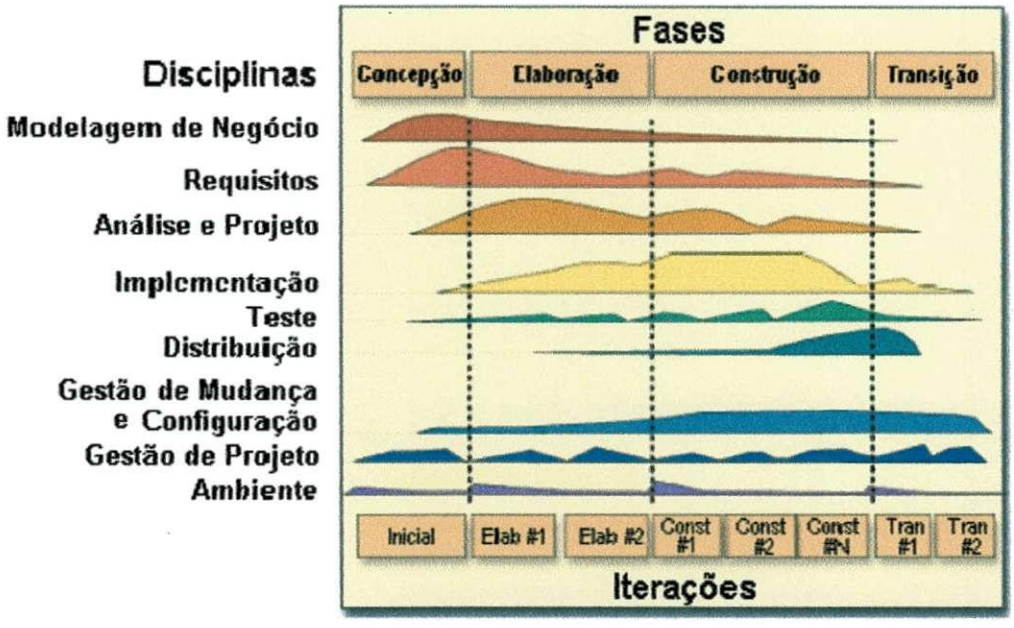


Figura 2 – Gráfico das Baleias

O RUP consiste também de nove fluxos de projetos, os quais são citados abaixo:

- Modelagem de negócios: descreve a estrutura e dinâmica da organização;
- Requisitos: descreve o método baseado em casos de uso para a elicitação de requisitos;
- Análise e projeto: descreve as múltiplas visões arquiteturais;
- Implementação: leva em conta o desenvolvimento de *softwares* com testes de unidade e integração;
- Testes: descrevem os casos de testes, procedimentos e métodos para coleta de métricas;
- Entrega: cobre as configurações de entrega para o sistema;
- Gerência de configuração: controla as mudanças e mantêm a integridade dos artefatos do projeto;

- Gerência do projeto: descreve as várias estratégias de trabalho de um projeto iterativo;
- Ambiente: cobre a infra-estrutura necessária para desenvolver o sistema.

Existe um conjunto de artefatos e atividades correspondente a cada fluxo de processos. Um artefato é um documento, relatório ou arquivo executável que é produzido, manipulado ou consumido. Uma atividade descreve as tarefas realizadas pelos executores para criar ou modificar artefatos, juntos com as técnicas e guias para gerenciar as tarefas, possivelmente incluindo o uso de ferramentas que ajudarão a automatizar algumas das tarefas.

2.2 Testes de *software*

Das disciplinas do RUP a que foi mais importante no decorrer do período de estágio foi a disciplina de testes. Desta forma, vamos discutir resumidamente sobre um processo de teste.

Em um processo de desenvolvimento de *software*, há uma grande possibilidade de injeção de falhas humanas no produto. Quanto mais cedo essas falhas são descobertas, menor é o custo de repará-la, como mostrado no gráfico da Figura 3.

Na Figura 3, pode-se observar o comportamento do custo de reparar uma falha durante o processo de desenvolvimento de *software*. Pode-se observar que o custo cresce exponencialmente e isto indica que é e relativamente barato se consertar um erro no inicio do processo, mas extremamente caro de consertar uma falha perto ou depois do lançamento do produto no mercado.

Outro aspecto importante é que no caso de um produto entrar no mercado com uma falha grave, há um comprometimento na imagem da empresa, o que poderá afetar consideravelmente seus negócios.

Assim, pode-se perceber a importância de se adotar um processo de testes no processo de desenvolvimento de um *software*.

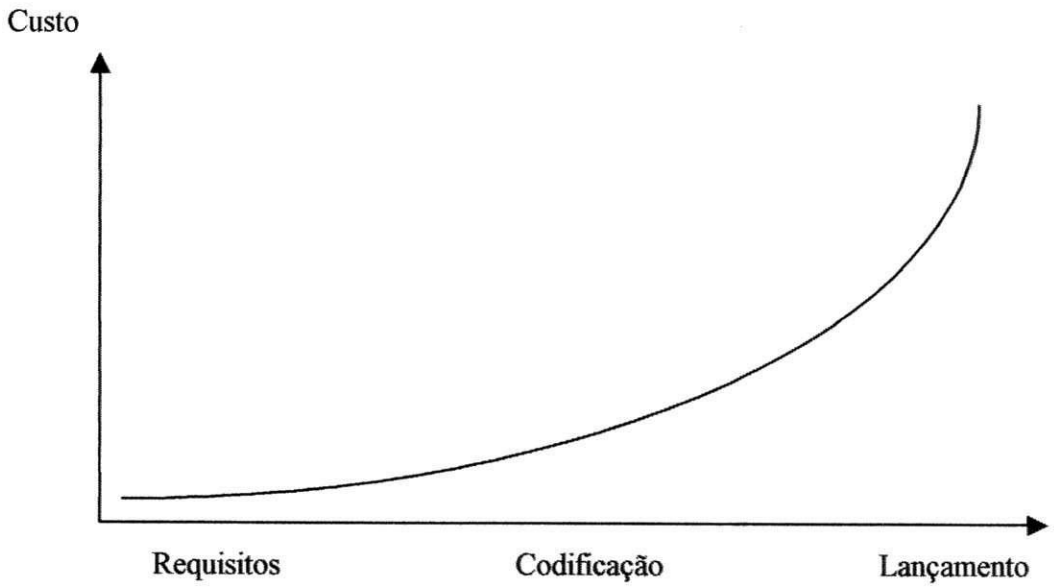


Figura 3 – Custo de reparo de falhas versus período de desenvolvimento

Contudo, há alguns problemas em se implementar um processo de testes. O primeiro é a impossibilidade de se encontrar todas as falhas em um *software* não-trivial [2]. O segundo problema é que um processo de testes pode consumir uma parcela considerável dos recursos totais do projeto, podendo consumir até 50% do esforço total gasto no desenvolvimento do produto. Desta maneira, um projeto de testes deve ser cuidadosamente projetado e implantado, de modo que o maior número de falhas possa ser descoberto gastando o menor montante de recursos possível.

De um modo geral, um processo de testes tem as seguintes finalidades:

- Verificar se todos os requisitos do sistema foram corretamente implementados;
- Assegurar a qualidade do produto e a corretude do *software* produzido, na medida do possível;
- Reduzir os custos de manutenção corretiva e retrabalho
- Assegurar a satisfação do cliente com o produto desenvolvido.

Um processo formal de teste incorpora as seguintes fases:

- Planejamento dos testes
- Projeto de testes
- Implementação dos testes
- Execução dos testes
- Avaliação dos testes

As duas principais abordagens de testes são a abordagem funcional e a abordagem estrutural.

Na abordagem funcional, também denominada de "caixa-preta", os testes são gerados a partir de uma análise dos relacionamentos entre os dados de entrada e saída, com base nos requisitos levantados com o usuário. Tem como objetivos encontrar erros de interface, na estrutura de dados ou de acesso ao banco de dados e de desempenho.

Na abordagem estrutural, também conhecida como "caixa-branca", os testes são gerados a partir de uma análise dos caminhos lógicos possíveis de serem executados, de modo a conhecer o funcionamento interno dos componentes do *software*.

2.2.1 Estágio de testes

Os estágios de um processo de teste são os seguintes:

- Testes de unidade: componentes individuais são testados para assegurar que os mesmos operam de forma correta.
- Testes de integração: a interface entre as unidades integradas é testada.
- Testes de sistema: os elementos do *software* integrados com o ambiente operacional são testados como um todo.
- Testes de aceitação: o *software* é testado pelo usuário final e envolve treinamento, documentação e empacotamento.

Os estágios de testes se integram ao RUP como mostrado na Tabela 1.

Tabela 1 - Fases do RUP e os estágios de testes

Fase do RUP	Estágios de Teste
Concepção	- Planejamento inicial dos testes
Elaboração	- Testes de unidade - Testes de integração
Construção	- Testes de unidade - Testes de integração - Testes de sistema
Transição	- Testes de sistema - Testes de aceitação

2.2.3 Tipos de testes

Testes podem ser divididos nos seguintes tipos:

- Teste funcional: a funcionalidade geral do sistema em termos de regras de negócio (fluxo de trabalho) é testada. Exemplo: em um *software* para gerenciamento de empréstimos de livros para uma biblioteca, um teste funcional seria testar se é possível realizar um empréstimo de um livro.
- Teste de recuperação de falhas: o *software* é forçado a falhar de diversas maneiras para que seja verificado o seu comportamento, bem como a adequação dos procedimentos de recuperação.
- Teste de integridade de dados: verifica a corretude dos métodos de acesso à base de dados e a garantia das informações armazenadas. Por exemplo, acessar dados contidos em um banco de dados.
- Teste de performance: verifica os tempos de resposta e processamento do *software*.
- Teste de volume (carga): verifica se o *software* suporta altos volumes de dados em uma única transação, além do número de terminais, modems e bytes de memória que é possível gerenciar. Por exemplo, transferir uma grande quantidade de dados, de uma única vez, para um banco de dados.

- Teste de stress: verifica a funcionalidade do sistema em situações limites. Por exemplo, executar e finalizar a execução do programa repetidas vezes e verificar se ao final dessas execuções o programa ainda funciona adequadamente.
- Teste de configuração ou portabilidade: verifica a correta instalação e desinstalação do sistema para diferentes plataformas de hardware/*software* e opções de instalação. Por exemplo, testar se o sistema é corretamente instalado no Windows e no Linux.
- Teste de interface com o usuário: verifica a aparência e comportamento da interface. Este teste é um dos mais complicados de se executar, pois tem forte relação com outras disciplinas, como marketing, por exemplo.
- Teste de regressão: re-execução de testes feitos após uma manutenção corretiva ou evolutiva. Uma manutenção pode alterar o funcionamento do sistema em uma parte do sistema diferente daquela na qual foi realizada a manutenção. Assim, deve-se testar o sistema como um todo, para verificar se o sistema ainda funciona adequadamente após a manutenção.
- Cobertura de testes: é uma medida de qualidade dos testes, medindo o quão completo é um conjunto de testes com relação a um critério adotado. Assim, tem como objetivo avaliar a qualidade dos testes que são executados em um determinado código e não o código em si. Denominando-se a variável CT como sendo a cobertura dos testes, tem-se a seguinte expressão:

$$CT = \frac{\text{nº de elementos requeridos exercitados}}{\text{total de elementos requeridos}}$$

3. DISCIPLINAS CURSADAS

Embora não façam parte da atividade de estágio propriamente dito, por terem sido importantes durante o período de duração do estágio e para que haja uma melhor descrição de todas as atividades realizadas durante a atividade de estágio, decidimos colocar neste relatório uma breve descrição de cada uma das disciplinas assistidas durante o Programa de Imersão Tecnológica.

Houve uma avaliação ao final de cada uma das disciplinas. Na avaliação, os participantes do curso tinham que obter nota igual ou superior a sete. Caso o participante obtivesse nota inferior a sete, era-lhe oferecido outra oportunidade, na qual ele faria outra avaliação. Se houvesse novamente não obtivesse nota superior a sete, o participante era excluído do curso.

Na grande maioria das disciplinas, a avaliação consistiu de uma prova. Entretanto, nas disciplinas Planejamento e Gerenciamento de Projetos de Software além da prova escrita houve também apresentação de trabalhos enquanto que na disciplina Projeto de Testes a avaliação consistiu apenas de apresentação de trabalhos.

A seguir, são apresentadas as disciplinas cursadas. Cada apresentação consiste de uma breve descrição e da ementa (quando possível).

3.1 Orientação a Objetos com Java / J2ME

Nesta disciplina, os conceitos básicos de programação orientada a objetos e da linguagem de programação Java foram apresentados e discutidos. Os conceitos vistos correspondem aos módulos apresentados acima. Também foram apresentadas noções sobre componentes Swing e AWT. No final do curso houve um treinamento em J2ME, versão de Java para componentes embarcados. Neste treinamento foram abordados temas como a construção de telas de interface, armazenamento de informações e criação de canais de comunicação de um sistema embarcado, no caso telefones celulares, para servidores Web. Utilizaram-se simuladores neste treinamento.

Ementa

- Introdução
- JSDK
- Sintaxe Básica
- Classes e Objetos; Atributos e Métodos
- Tipos de Referência, Strings e arrays
- Herança, Polimorfismo e Ligação Dinânmica
- Modelagem em Camadas
- Classes Abstratas e Interfaces
- Exceções
- Pacotes
- Treinamento J2ME

3.2 Fundamentos do Processo Unificado

Nessa disciplina, os conceitos básicos do *Rational Unified Process* (RUP) foram apresentados e discutidos. Como o RUP é um conjunto de processos que modelam todos os passos de ciclo de vida de um projeto de desenvolvimento de *software*, esta disciplina também atuou como uma apresentação para as demais disciplinas do curso.

Ementa

- Introdução
- Casos de Uso
- Arquitetura de Software
- Modelos de Ciclo de Vida
- Visão Geral da Metodologia
- Arquitetura Baseada em Componentes
- Gerência de Requisitos
- Desenvolvimento Iterativo
- Modelagem Visual
- Verificação de Qualidade
- Controle de Mudanças
- Concepção
- Elaboração
- Construção
- Transição

Bibliografia

JACOBSON, Ivar; BOOCH, Grady; RUMBAUGH, James. The Unified Software Development Process. Addison-Wesley, 1998.

KRUCHTEN, Phillipe. The Rational Unified Process: An Introduction.
http:/www.rational.com/rup

3.3 Planejamento e Gerenciamento de Projetos de Software

Nesta disciplina, uma metodologia de desenvolvimento de *software* interativa e incremental foi apresentada e discutida. Além disso, foram apresentadas e discutidas as atividades de planejamento e projetos sob a ótica de um processo interativo e incremental, bem como as dúvidas mais comuns relacionadas ao planejamento de projetos. Vale salientar, também, que esta disciplina teve uma atividade prática, que fez parte da avaliação.

Ementa

- Introdução
- Modelos de Ciclo de Vida
- Estimativas de Custo
- Casos de Uso
- Visão Geral da Metodologia
- Fluxo de Atividades
- Riscos
- Planejamento
- Organização de Equipes
- Boas Práticas
- Implementando o Processo

Bibliografia

JACOBSON, Ivar; BOOCH, Grady; RUMBAUGH, James. The Unified Software Development Process. Addison-Wesley, 1998.
ROYCE, Walker. Software Project Management. Addison-Wesley, 1998.
CANTOR, Murray. Object-Oriented Project Management with UML. : John Wiley & Sons. 1998.
HALL, Elaine M. Managing Risks: Methods for Software Systems Development. Addison-Wesley, 1998.
BOOCH, Grady. Object Solutions: Managing the Object-Oriented Project (Addison-Wesley Object Technology Series). Addison-Wesley, 1999.

3.4 Engenharia de Requisitos

Nesta disciplina, os conceitos básicos relativos a Engenharia de Requisitos foram apresentados e discutidos. O processo padrão da Engenharia de requisitos foi mostrado, bem como técnicas para levantar, especificar e estruturar os casos de uso do sistema. No final, as principais técnicas de gerenciamento de requisitos foram discutidas.

Ementa

- Motivação
- O Processo Padrão de Engenharia de Requisitos
- Fluxo de Requisitos
  - ✓ Conceitos Básicos
  - ✓ Levantamento de Casos de Uso
  - ✓ Especificação detalhada dos casos de uso
  - ✓ Estruturação dos casos de uso
  - ✓ Revisão dos Requisitos
- Gerenciamento de Requisitos

3.5 Testes de Software

Nesta disciplina, inicialmente foi observado que o processo de teste incorpora as seguintes etapas: planejamento, projeto, implementação, execução e avaliação dos testes. Em seguida foram analisados os tipos de testes existentes, bem como os estágios de um processo de testes. Cada um desses estágios foi detalhado e foram apresentados os principais tipos de testes realizados em cada estágio. O curso teve continuidade com a relação entre os testes, as fases de desenvolvimento de SW e os fluxos de atividades.

Ementa

- Conceitos chaves
- Relação com fases e outros fluxos
- Fluxo de testes
- Ferramentas de testes
- Apresentar os diferentes tipos de ferramentas que podem ser utilizadas no processo de testes

Bibliografia

KANER, Cem; FALK, Jack; NGYEN, Hung Quoc.. Testing Computer Software, Second Edition. New York: John Wiley & Sons. 1999.
NGYEN, Hung Quoc. Testing Applications on the Web. John Wiley & Sons.

3.6 Ferramentas de Testes

Neste curso algumas ferramentas utilizadas para se realizar testes em um *software* foram estudadas em laboratório. As ferramentas utilizadas foram as seguintes: Rational TestManager, Rational Robot, JUnit, Rational Test RealTime, Canoo, Jmeter e Ant.

3.7 Análise e Projetos de Software com UML

Nesta disciplina, os conceitos básicos do fluxo de análise e projeto de um sistema foram apresentados. Além disso, apresentou-se a linguagem de modelagem UML e como modelar um sistema computacional utilizando-a. Por fim, apresentou-se um padrão de arquitetura em camadas, que no caso específico desta disciplina, foi o padrão utilizado pela Qualiti e pelo CESAR.

Ementa

- Introdução
- Casos de Uso
- Fluxo de Requisitos
- Levantamento de Casos de Uso
- Especificação Detalhada dos Casos de Uso
- Diagrama de Atividades
- Requisitos Não-Funcionais
- Estrutura do Modelo de Casos de Uso
- Qualificação do Modelo de Casos de Uso
- Orientação a Objetos com UML
- Fluxo de Análise e Projeto
- Analisar Caso de Uso
- Projetar Arquitetura, Distribuição, Casos de Uso, Subsistema, Cápsulas, Classes e Base de Dados
- Considerações Finais

Bibliografia

JACOBSON, Ivar; BOOCH, Grady; RUMBAUGH, James. The Unified Software Development Process. Addison-Wesley, 1998.
RUMBAUGH, James; JACOBSON, Ivar; BOOCH, Grady. The Unified Modeling Language Reference Manual. Addison-Wesley, 1999.
DOUGLASS, Bruce Powel. Real-Time UML – Developing Efficient Objects. Addison-Wesley, 1998.

3.8 Arquitetura de Software para Celulares

Nesta disciplina, foram vistos os conceitos básicos relativos a sistemas embarcados, bem como a arquitetura básica, as tecnologias empregadas e as metodologias de projeto que são utilizadas no desenvolvimento de um sistema embarcado. Além disso, foi visto técnicas de programação com memória restrita e por fim, foi mostrada a arquitetura da plataforma utilizada em celulares da Motorola Ltda.

Ementa

- Introdução aos sistemas embarcados
- Arquiteturas de software para celulares
- A arquitetura Plataforma 2000 da Motorola Industrial Ltda
- Gerenciamento de memória limitada

Bibliografia

http:/www.lcmi.ufsc.br/grt/livro/principal.htm
http:/www.cs.bu.edu/pub/ieee-rst
http:/www.timesys.com

3.9 Projeto de Testes

Nesta disciplina foi visto aspectos relativos à elaboração, implementação e implantação de um projeto de teste em um processo de desenvolvimento de *software*.

3.10 Gerência de Configuração

Nesta disciplina, os conceitos básicos relativos a Gerência de Configuração foram apresentados e discutidos. As ferramentas CVS, Ant e Bugzilla, que auxiliam na Gerência de Configuração foram apresentadas e seu funcionamento discutido através de atividades práticas.

Ementa

- Introdução
- Adoção gradual
- Padrões
- CVS
- *Builds* com ant
- Bugzilla

- Gerência de mudanças
- Fluxo de gerência de configuração de ambiente

3.11 Qualidade de Software

Esta disciplina abordou de forma resumida todos os aspectos que envolvem a adoção de processo de qualidade por empresas desenvolvedoras de *software*. Os modelos ISO, CMM e CMMI foram abordados e comentados, tendo suas principais características discutidas.

Ementa

- Qualidade de software
- Qualidade de produto de software
- Modelos ISO
- Norma ISO 9000-3
- Modelo SEI/SW-CMM
- Modelo SEI/SW-CMM Nível 2
- Modelo SEI/SW-CMM Nível 3
- CMMI
- Aspectos Práticos

Bibliografia

http:/www.iso-9000.co.uk/index
http:/www.isonet.com
http:/www.iso.ch
http:/www.abnt.org
http:/www.sei.cmu.edu/cmmi/
http:/www.sei.cmu.edu
http:/www.cin.ufpe.br/~qualisoft
http:/www.cin.ufpe.br/~processos/TAES3

Além destas disciplinas, houve mais uma disciplina que teve caráter meramente informativo. Esta disciplina foi a de XML Básico, e consistiu de um treinamento nos conceitos básicos de XML.

3.12 Aproveitamento nas disciplinas

O aproveitamento nas disciplinas cursadas foi muito bom. Os assuntos abordados nas disciplinas tiveram uma boa assimilação e em geral, todos os participantes do curso não tiveram dificuldades durante cada uma das disciplinas.

Uma única ressalva é o fato de que as disciplinas poderiam ter tido uma carga maior de exercícios práticos, o que melhoraria ainda mais o rendimento da turma.

4. ATIVIDADES DE ESTÁGIO

O projeto CIn-BTC funciona como um prestador de serviços à Motorola Industrial Ltda, sendo que o principal serviço prestado é a execução de testes, nos estágios de testes de integração e de testes de sistema, em celulares Motorola, sendo estes testes tanto manuais como automáticos.

O fluxograma das atividades realizadas durante o estágio pode ser visto no Anexo A. Essas atividades correspondem àquelas desempenhadas por um bolsista do grupo de execução de testes (vide Figura 1). Desta forma, as atividades realizadas durante o período de estágio consistiram de:

- Execução de testes manuais;
- Contribuição com a coleta de métricas;
- Sugestões para a melhoria do processo adotado pelo projeto CIn-BTC.

Infelizmente, nem todas as descrições dos procedimentos, processos e mesmo do trabalho realizado pelo estagiário enquanto participante do Programa de Imersão Tecnológica poderão ser detalhadas, porque como os celulares testados correspondem em sua maioria a protótipos que ainda não foram lançados no mercado, mundial ou nacional, todos os participantes do projeto assinam contratos de sigilo com a Motorola. Todavia, daremos uma visão superficial que será suficiente para descrever as atividades realizadas durante a realização do estágio.

No projeto CIn-BTC há dois grupos principais: o grupo de Execução de Testes e o grupo de Automação. Como o próprio nome já diz, o grupo de Execução de Testes é encarregado da execução de testes manuais e automáticos, enquanto o grupo de Automação é encarregado de desenvolver scripts de testes automáticos que serão executados e pelo desenvolvimento e/ou aperfeiçoamento de ferramentas de testes desenvolvidas pelo próprio CIn-BTC.

O grupo de Execução de Testes, por sua vez, possui algumas subdivisões, cada uma responsável por testar uma determinada *feature* do celular. Testa-se tanto toda a

parte relativa à comunicação entre celulares como envio e recebimento de mensagens e *e-mail*, bate-papo etc., como as funções de multimídia presentes nos celulares. Essas funções de multimídia correspondem a obter, armazenar, realizar o download e remover arquivos de multimídia como fotos, figuras, arquivos de áudio e vídeo.

4.1 Execução de Testes

Um grande problema com um processo de desenvolvimento de software de grande porte é que há uma grande e constante mudança nos requisitos e geralmente os testes realizados durante as fases de implementação do código não são suficientes para assegurar a qualidade do produto por serem superficiais. Por isso, é necessária a execução de testes além daqueles que são executados pelos desenvolvedores.

A execução de testes tem dois objetivos principais:

- Realizar testes em telefones celulares Motorola na qualidade e tempo requeridos.
- Conter defeitos no *software* do celular

Durante a atividade de estágio, testamos basicamente duas *features* do telefone celular: MMA e EMS.

A *feature* de MMA (*Multimedia Application*) corresponde a toda funcionalidade de multimídia do telefone celular. Os tipos de arquivos que são testados nessa *feature* são arquivos de temas, arquivos de imagens, arquivos de áudio e arquivos de vídeo. Os arquivos de temas podem conter a tela de apresentação do telefone celular, tela de descanso e a música que é tocada quando o telefone recebe uma chamada. Testes que são geralmente realizados são executar *downloads* de arquivos dos tipos mencionados e instalá-los no telefone celular; executar um arquivo de áudio, vídeo ou imagem; criar listas de execução de algum tipo desses arquivos, entre outros testes. Em telefones que possuam câmera, alguns dos testes realizados consistiram em tirar e editar fotos, gravar e editar vídeos, entre outros.

A *feature* de EMS (*Enhanced Messaging Service*) é um sistema de mensagens aprimorado para telefones celulares com tecnologia GSM que permite o envio e o

recebimento de arquivos para serem utilizados como descanso de tela, papel de parede e toques do telefone. Os testes dessa *feature* consistem em se testar o envio e o recebimento de mensagens EMS com diferentes tipos de arquivos anexos, como também anexos com tamanhos diferentes. Um dos problemas em se realizar os testes de EMS é a dependência dos testes com as operadoras de telefonia celular. Isto pode ser uma dificuldade, pois problemas em uma operadora podem atrasar ou mesmo impossibilitar a execução do caso de teste utilizando-se determinada operadora. Para contornar este problema, já existem ferramentas que simulam o envio de mensagens para o telefone celular. Estas ferramentas são bastante úteis quando se executa testes de recebimento de mensagens.

Com relação aos tipos de testes que são realizados no CIn-BTC, pode-se dizer que os seguintes testes são realizados:

Testes funcionais: no caso do CIn-BTC, testes funcionais são realizados tanto nos testes de sanidade como nos testes de regressão. No caso dos testes de sanidade, são testadas apenas as funcionalidades básicas relativas a cada uma das diversas *features* do grupo de Execução de Testes. Nos testes de regressão, os testes funcionais são apenas uma parte dos testes que são realizados, mas um número maior de funcionalidades do celular são testadas, e o são mais extensivamente.

Testes de integridade de dados: foram testadas estruturas de dados presentes no software dos celulares, como listas de arquivos (áudio, vídeo e figuras), que possuíam uma dimensão pré-determinada e verificava-se se era possível adicionar o número máximo de elementos, se era possível adicionar um elemento além do número máximo de elementos, se os dados dessas estruturas permaneciam disponíveis e inalterados depois de operações com outros dados da estrutura de dados, entre outros testes.

Testes de volume: esses testes foram realizados principalmente no envio e recebimento de mensagens. Tinham como finalidade verificar se mensagens com arquivos anexo de diversos tamanhos podiam ser enviados e recebidos pelos telefones celulares. Eram realizados nas baterias de testes de regressão.

Testes de stress: vários tipos de testes de stress eram realizados nas duas *features* que foram testadas ao longo do estágio[1]. Na *feature* de MMA, corresponderam a tirar diversas fotos, gravar vários vídeos, executar um arquivo de áudio e o deixar tocando durante um longo período, entre outros. Na feature de EMS, corresponderam a enviar várias mensagens, com arquivos anexos ou não, em seqüência a partir do modelo que estava sendo testado; receber várias mensagens, com anexo ou não, no modelo que estava sendo testado, entre outros testes.

Testes de regressão: esses testes são um conjunto dos demais testes que são realizados pelo CIn-BTC, sendo uma grande bateria de testes, nas quais todas as funcionalidades de uma determinada *feature* são exaustivamente testadas, de modo a verificar que mudanças realizadas não alteraram o comportamento de outros componentes do software. Além disso, no CIn-BTC, os testes de regressão atuam como testes de unidade, pois as funcionalidades de cada componente são testadas, e testes de integração, pois funcionalidades diferentes são testadas de forma integrada.

No CIn-BTC, executam-se testes de sanidade e testes de regressão, sendo que nas baterias de testes de regressão todos os outros tipos de teste mencionados são executados. Não são executados testes de usabilidade e testes de configuração ou portabilidade, pois esses testes estão fora do escopo do projeto. Os testes de performance não são executados de forma direta, pois não há casos de testes específicos para isso, mas a partir de outros tipos de casos de teste, como casos de teste de volume e de stress entre outros, é possível realizar uma avaliação superficial do desempenho do sistema.

Antes de se testar um determinado protótipo de telefone celular, primeiramente é necessário configurá-lo. Essa configuração consiste em se injetar no telefone todos os arquivos que são necessários à execução dos testes, como a *build* e arquivos de configuração, por exemplo. Uma *build* representa uma versão ainda incompleta do sistema em desenvolvimento, mas com certa estabilidade[5]. Atualmente, utilizam-se programas específicos para se injetar esses arquivos e isto é feito conectando-se o

[1] Em todas as *features* há testes de stress, entretanto, não se sabe o que realmente se testava, pois não houve contato com essas *features*.

protótipo do celular a um computador por meio de um cabo USB (Universal Serial Bus). Este procedimento de injeção de arquivos no telefone celular deve ser feito com cuidado, pois uma falha pode acarretar em um dano permanente no protótipo.

Os possíveis resultados que podem ser atribuídos a um caso de teste são os seguintes:

- Passou: quando não há falhas
- Falhou: quando é encontrada uma falha. Quando esse resultado ocorre, é necessário criar uma requisição de mudança contendo a descrição da falha e o procedimento para repeti-la. Depois, deve-se submeter esta requisição de mudança à aprovação dos supervisores da área de testes da Motorola (existe um sítio especial na rede interna da Motorola para fazer a submissão). Se for confirmada a falha, a requisição de mudança segue para a equipe de desenvolvimento responsável pela área onde ocorreu a falha, caso contrário, a requisição de mudança é ignorada.
- Bloqueado: quando por alguma razão não é possível executar o teste, por exemplo, se um teste depende da conexão do celular com a internet, mas no momento do teste esta conexão não está disponível. Deve-se colocar no repositório central de testes o motivo pelo qual se bloqueou o teste.
- Investigar: quando há algum problema com a descrição do caso de teste ou o resultado do teste é ambíguo. Neste caso, também se deve colocar no repositório central o motivo do pedido de investigação.
- Excluir: quando o caso de teste não se aplica a um determinado telefone celular, por exemplo, um caso de teste que testa a função de tirar uma foto em um teste para um telefone celular que não possui uma câmera. Neste caso, também se deve colocar no repositório central o motivo do pedido de exclusão.

4.1.1 Execução de Testes Manuais

A execução de testes manuais foi a principal atividade realizada durante o período de estágio. Os testes manuais são testes com abordagem funcional e consistem em executar manualmente casos de testes previamente definidos e contidos no repositório central de testes da Motorola, de modo a verificar se o funcionamento do celular está de acordo com o esperado.

Há basicamente dois tipos de testes manuais que são realizados: os de sanidade e os de regressão.

Como já foi dito, os testes de sanidade são testes funcionais que têm como objetivo testar as funcionalidades básicas de um determinado modelo de celular. São executados para cada nova versão da *build* que é desenvolvida para um determinado modelo de celular.

Os testes de regressão têm como objetivo testar se as funcionalidades de um determinado modelo de celular continuam funcionando após serem realizadas modificações no *software* utilizado pelo mesmo. Normalmente são grandes baterias de testes que testam a funcionalidade do celular como um todo.

Os resultados tanto dos testes de sanidade como dos testes de regressão são colocados em um repositório central de teste, localizado na rede interna da Motorola. Nesse repositório estão todas as informações necessárias para que os testes possam ser executados, como *builds* que devem ser aplicadas, outros arquivos que devem ser injetados no celular, planilhas que contém todos os testes que deverão ser executados para um determinado modelo de celular, etc.

Ao final da execução da planilha de testes, deve-se elaborar um breve relatório indicando todos os resultados obtidos para a planilha de testes, e isso deve ser feito tanto para os testes de sanidade quanto para os testes de regressão. A elaboração desse relatório foi sempre realizada por um dos engenheiros de teste contratados pelo CIn-BTC.

Nesse repositório central de testes, também são armazenados os dados relativos a duração dos tempos de *setup* e execução de cada caso de teste. A coleta destes dados é muito importante, pois auxilia na verificação de problemas, na estimação de metas e no melhoramento do processo.

O ambiente de teste consiste em um computador e o protótipo do celular que deve ser testado. Se for necessária a injeção de arquivos no telefone celular, ferramentas de software são utilizadas para se injetar esses arquivos. Na Figura 4, exibe-se um esquema deste ambiente.

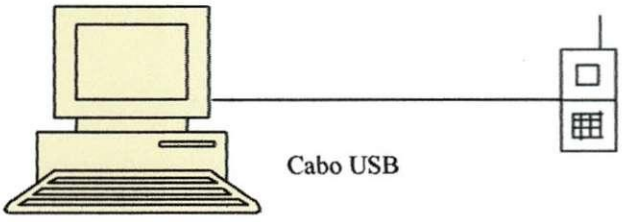


Figura 4 – Ambiente de teste

Dos telefones celulares testados, alguns já foram lançados no mercado internacional ou nacional. Na Figura 5, são exibidos alguns desses.

E1000 E398 RAZR V3 V600 E550

Figura 5 – Modelos testados já no mercado

4.1.2 Testes Automáticos

Os testes automáticos também são testes caixa-preta, mas são executados por um computador por meio de ferramentas de testes. Algumas dessas ferramentas foram desenvolvidas dentro do próprio CIn-BTC, outras foram desenvolvidas pela Motorola. A conexão do computador com o telefone celular se faz por meio de cabo USB, como mostrado na Figura 4.

Baseados nos casos de testes são escritos scripts de testes, nos quais estão contidos todos os passos que serão necessários para que uma determinada ferramenta possa executar um determinado caso de teste e verificar o resultado. Estes scripts são desenvolvidos pela equipe de automação do CIn-BTC.

Os testes automáticos consistem em configurar o modelo que será testado e a ferramenta que será utilizada para que ela possa executar os testes no modelo especificado. Em seguida, utilizando a ferramenta de testes, executam-se os casos de teste. O testador deve estar atento ao processo de testes para verificar os resultados obtidos e é preferível executar um caso de teste de cada vez, para possíveis erros possam ser identificados mais facilmente no relatório gerado pela ferramenta durante a execução dos testes.

Apesar de já se obter bons resultados com a automação de testes. Ainda existem alguns problemas com a execução de testes automáticos. O primeiro deles é que nem todos os casos de teste são automatizáveis, pois certos casos de teste necessitam da intervenção humana, como por exemplo, um teste que verifica o estado do telefone celular quando ele está realizando uma determinada função e se fecha o *flip*. Outro problema é que o tempo de execução da maioria dos scripts de testes realizados automaticamente é bem superior aos correspondentes tempos de execução de testes manuais e apesar da confiabilidade ter aumentado bastante, ainda há a necessidade de um ser humano para validar os resultados obtidos pelos testes automáticos.

Até o término do período de estágio, somente testes de sanidade tinham sido automatizados e mesmo assim, só os casos de testes de algumas *features*. Os motivos

para isso é que o processo de automação dos testes é relativamente recente e o processo de automação dos testes é relativamente lento, pois o processo possui etapas que devem ser compridas antes que o código do caso de teste automatizado seja validade e entre em funcionamento normal.

Há um esforço considerável por parte do CIn-BTC e da Motorola para que o maior número de casos de testes possível seja automatizado. Automatizar casos de testes é uma atividade importante, pois evita que os engenheiros de teste executem atividades muito repetitivas ou demoradas como testes de stress e de volume, podendo dedicar mais tempo na elaboração e melhoramento dos casos de teste.

4.2 Conversão de caso de testes de baixo nível para alto nível

Casos de teste de baixo nível são casos de teste cujo procedimento é bastante detalhado e tem como objetivo não apenas testar uma funcionalidade em si, mas como estão dispostos os comandos na tela do celular, o caminho para se chegar à determinada funcionalidade e como os objetos são exibidos na tela, entre outros.

Um problema com os testes de baixo nível é que eles são muito dependentes da plataforma do celular, assim, celulares com plataformas diferentes deverão ter casos de teste em baixo nível diferentes, pois a disposição dos comandos e a organização da tela nas duas plataformas poderão ser diferentes.

Desta forma, casos de teste em baixo nível não são muito interessantes quando se deseja automatizar os casos de teste, pois um dos objetivos é escrever um *software* que substitua os casos de teste manuais em vários telefones celulares diferentes.

Uma solução encontrada para automatizar os casos de teste é antes convertê-los para testes de alto nível antes de realizar a automação propriamente dita.

Casos de teste de alto nível são casos de teste que são escritos de tal maneira que a funcionalidade é testada independente de como foi feito para se testar tal funcionalidade.

Assim, não importa a disposição dos comandos da tela nem como as telas estão organizadas, desde que a funcionalidade possa ser testada.

O processo de conversão de testes de baixo nível para caso de testes em alto nível possui as seguintes etapas:

- Conversão dos casos de teste: os casos de teste em baixo nível são reescritos para se tornarem testes de alto nível;
- Inspeção: são reuniões formais cujo objetivo é se encontrar falhas no produto, que neste caso, são os casos de teste em alto nível;
- Re-trabalho: correção das falhas apontadas durante o a etapa de inspeção;
- Validação: validar os casos de teste, verificando se são compreensíveis e se estão de acordo com os objetivos iniciais do caso de teste em baixo nível.

É interessante que quem estiver incumbido de realizar a conversão de casos de teste de baixo nível para alto nível tenha uma certa familiaridade com os casos de testes que serão convertidos para diminuir o risco de haver alterações ns objetivos do caso de teste durante o processo de reescrita. Desta forma, geralmente um testador da *feature* que está sendo automatizada geralmente faz parte do processo de conversão. Foi desta maneira que surgiram as oportunidades de participar de processos de conversão de casos de teste de baixo nível para casos de teste de alto nível.

4.3 Monografia

Como já foi dito anteriormente, todos os alunos do curso têm que escrever uma monografia e apresentá-la ao final do curso. Esta monografia se assemelha à disciplina de Projeto do curso de graduação de Engenharia Elétrica do campus I da UFCG, sendo que ao invés de elaborar um projeto de engenharia, na monografia deve-se elaborar um projeto de melhoria do processo interno do CIn-BTC.

Apesar de ser parte integrante do curso, a monografia também pode ser vista como uma das atividades práticas realizadas durante o período de estágio, uma vez seu

objetivo é contribuir para o melhoramento das condições de trabalho do CIn- BTC. A monografia tem que ser escrita em inglês e deve ser apresentada a uma banca formada por dois professores do Centro de Informática da UFPE. Como nas disciplinas do curso, deve-se obter uma nota igual ou superior a sete na monografia. A monografia pode ser feita individualmente ou em grupo de no máximo três pessoas.

No caso deste estagiário em particular, a monografia foi realizada em parceria com outro participante do curso. Nela, foi realizada uma avaliação das vantagens e desvantagens de se adotar um processo de realização de testes de cobertura de código desenvolvido pela equipe de automação do CIn-BTC.

Os testes de cobertura de código, diferentemente dos demais tipos de testes, não têm como finalidade testar o código fonte em si, mas verificar se os testes unitários e/ou de integração estão testando completamente o código fonte. Assim, os testes de cobertura atuam como sendo o teste dos testes e são bastante úteis na identificação de partes do programa que não estão sendo executadas e assim, são utilizados na melhoria dos testes unitários e de integração que estão sendo executados. Possuem como desvantagem o aumento do consumo dos recursos do projeto em pessoal e tempo, principalmente, como ocorre nos demais processo de teste em *softwares*.

Infelizmente não é possível detalhar esse trabalho neste texto, pois utilizamos códigos que são propriedade da Motorola que não devem ser comentados por questão de sigilo. Contudo, pode-se dizer que o trabalho consistiu em realizar testes de cobertura nos testes unitários de uma das ferramentas mais utilizadas pela equipe de automação, analisar os dados coletados e com base nesses dados, verificar a utilidade de testes de cobertura para a melhoria do processo de automação de casos de testes.

Utilizou-se uma ferramenta de testes de cobertura disponível no mercado para auxiliar na execução dos testes, uma vez que realizar testes de cobertura em um código não trivial é uma tarefa bastante complexa, devido ao grande número de fluxos de execução que esses códigos podem apresentar. A ferramenta utilizada foi a *Clover*. Para realização dos testes, configurou-se a *Clover* para que ela fosse executada quando os

testes unitários da ferramenta sob teste fossem executados. Desta forma, foi possível verificar a porcentagem de cobertura de cada um dos componentes do código, bem como as partes do código que não foram exercitadas pelos testes unitários. As conclusões obtidas pela análise dos resultados foram favoráveis a adoção de testes de cobertura no processo de automação de testes do CIn-BTC.

5. CONCLUSÕES

O Curso de Formação Seqüencial em Análise de Testes contribui de forma bastante positiva para a formação profissional de um estagiário, uma vez que ele é inserido dentro de uma organização que possui um processo definido, no qual todos sabem o que deve ser feito. A cobrança de resultados e a supervisão constante das atividades desempenhadas desenvolvem o senso de responsabilidade do estagiário e o suporte que lhe é dado por parte dos profissionais lhe são superiores dentro da hierarquia da empresa é bastante satisfatório. Além disso, a comunicação entre os membros de uma equipe é estimulada, desenvolvendo a capacidade de trabalhar em grupo do estagiário.

Entretanto, houve alguns problemas durante o estágio. O primeiro deles, é que as atividades desempenhadas durante o estágio agregaram pouco conhecimento técnico da área de Engenharia Elétrica ao estagiário, embora a formação técnica em engenharia, com o desenvolvimento do raciocínio lógico entre outras coisas, tenha sido bastante útil no decorrer do estágio. Um segundo problema é que diferentemente do que havia sido pensado no inicio do curso, uma vez que foi dito que o curso se assemelhava à residência médica, não houve uma integração entre as disciplinas cursadas e as atividades de estágio. Desta forma, não foi possível obter o nível de conhecimento que poderia ter sido obtido caso a integração tivesse ocorrido de fato.

Entretanto, apesar dos problemas, avaliamos que a atividade de estágio foi bastante benéfica devido à experiência de se trabalhar em uma empresa com um processo bem definido, o aumento da capacidade de trabalhar em grupo e de comunicação que foram adquiridas. Além disso, o curso possibilitou um bom aprendizado em Engenharia de Software e em análise de testes, esta última sendo uma promissora área de atuação para profissionais qualificados.

6. REFERÊNCIAS BIBLIOGRÁFICAS

[1] BRAZ, Eliezer F.; SOARES, Elifrancis R. Code Coverage Analysis on TAF Utility Functions. Monografia Final do Programa de Imersão Tecnológica - Turma 4. Recife. CIn-BTC. 2005.

[2] KANER, Cem; FALK, Jack; NGUYEN, Hung Quoc.. Testing Computer Software, Second Edition. New York: John Wiley & Sons. 1999.

[3] LEWIS, William E.. Software Testing and Continuous Quality Improvement. Boca Raton: CRC Press. 2000

[4] PERRELLI, Hermano. Fundamentos do Processo Unificado. Apostila do Curso Seqüencial de Formação Complementar. Centro de Informática. Universidade Federal de Pernambuco. 2004.

[5] SANTOS, André. Gerência de Configuração. Apostila do Curso Seqüencial de Formação Complementar. Centro de Informática. Universidade Federal de Pernambuco. 2004.

[6] VASCONCELOS, Alexandre. Teste de Software. Apostila do Curso Seqüencial de Formação Complementar. Centro de Informática. Universidade Federal de Pernambuco. 2004.

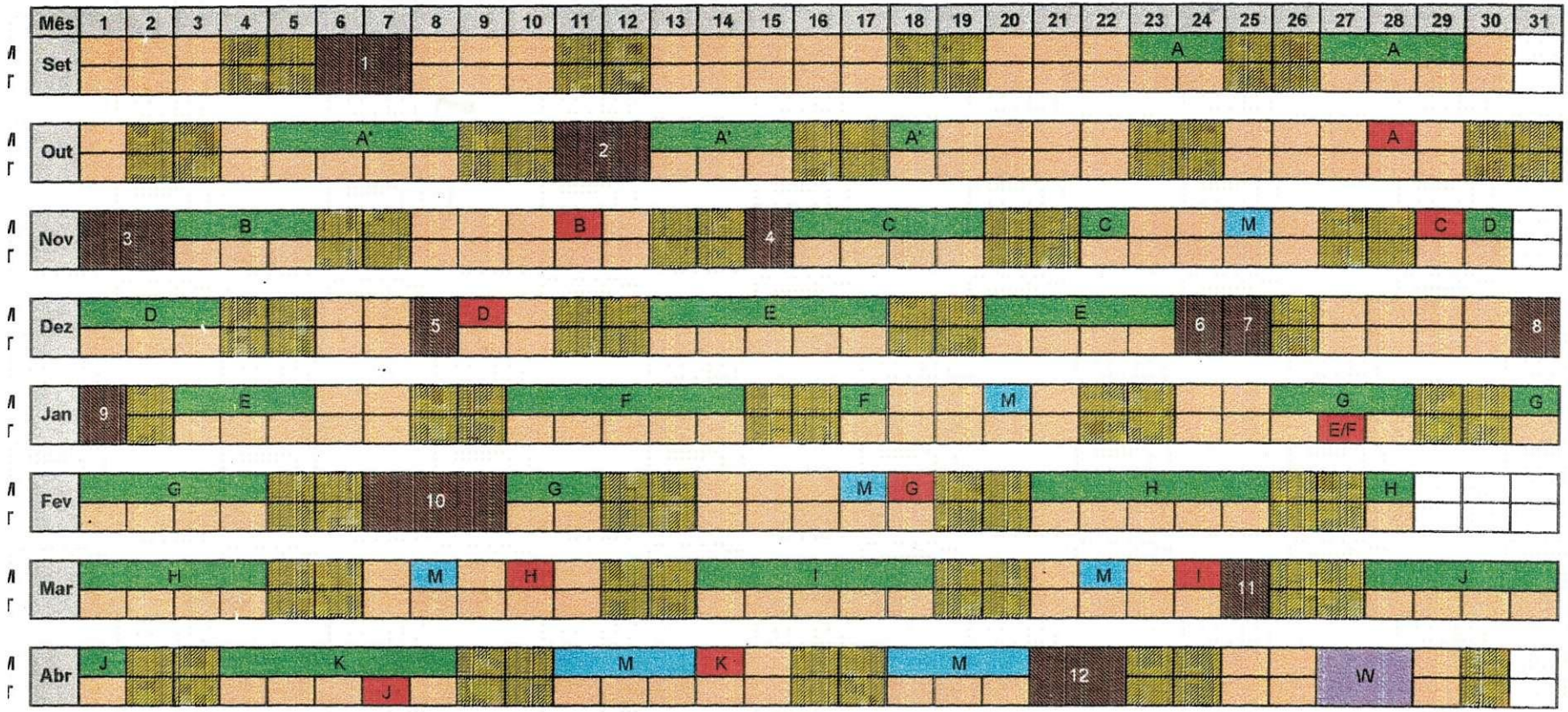
[7]http://www.cin.ufpe.br/servlets/newstorm.notitia.apresentacao.ServletDeNoticia?codigoDaNoticia=3662&dataDoJornal=1091826321000 (acessado em maio de 2005).

[8] http://www.ibm.com/br/products/software/rational/rup.phtml (acessado em maio de 2005).

[9] http://www-128.ibm.com/developerworks/rational/library/873.html (acessado em maio de 2005).

[10]http://tede.inatel.br/tede/tde_arquivos/1/TDE-2004-03-23T04:31:19Z-27/Publico/LPA.pdf (acessado em junho de 2005).

CURSO SEQÜENCIAL DE FORMAÇÃO COMPLEMENTAR EM ANÁLISE DE TESTES - TURMA 4
CALENDÁRIO DE AULAS E LABORATÓRIO



Cód	Disciplina	Professor	C.H.
A	Orientação a Objetos com Java / J2ME (Parte I)	Roberto Barros	20
A'	Orientação a Objetos com Java / J2ME (Parte II)	Paulo Borba	32
B	Fundamentos de um Processo Iterativo e Incremental	Hermano Perrelli	12
C	Planejamento e Gerenciamento de Projetos	Hermano Perrelli	20
D	Engenharia de Requisitos	Jaelson Castro	16
E	Testes de Software	Alexandre Vasconcelos	48
F	Ferramentas de Testes	Alexandre Vasconcelos	24
G	Análise e Projetos de Software com UML	Augusto Sampaio	40
H	Arquitetura de Software para Celulares	Sergio Cavalcante	40
I	Projeto de Testes	Alexandre Vasconcelos	20
J	Gerência de Configuração	André Santos	20
K	Qualidade de Software	Alexandre Vasconcelos	20
L	Monografia	Augusto Sampaio	240

FERIADOS

1-	Dia da Independência	7-	Natal
2-	Dia de N. Sra. da Aparecida	8-	Véspera de Ano Novo
3-	Finados	9-	Confraternização Universal
4-	Proclamação da República	10-	Carnaval
5-	Dia de N. Sra. da Conceição	11-	Semana Santa
6-	Véspera de Natal	12-	Tiradentes