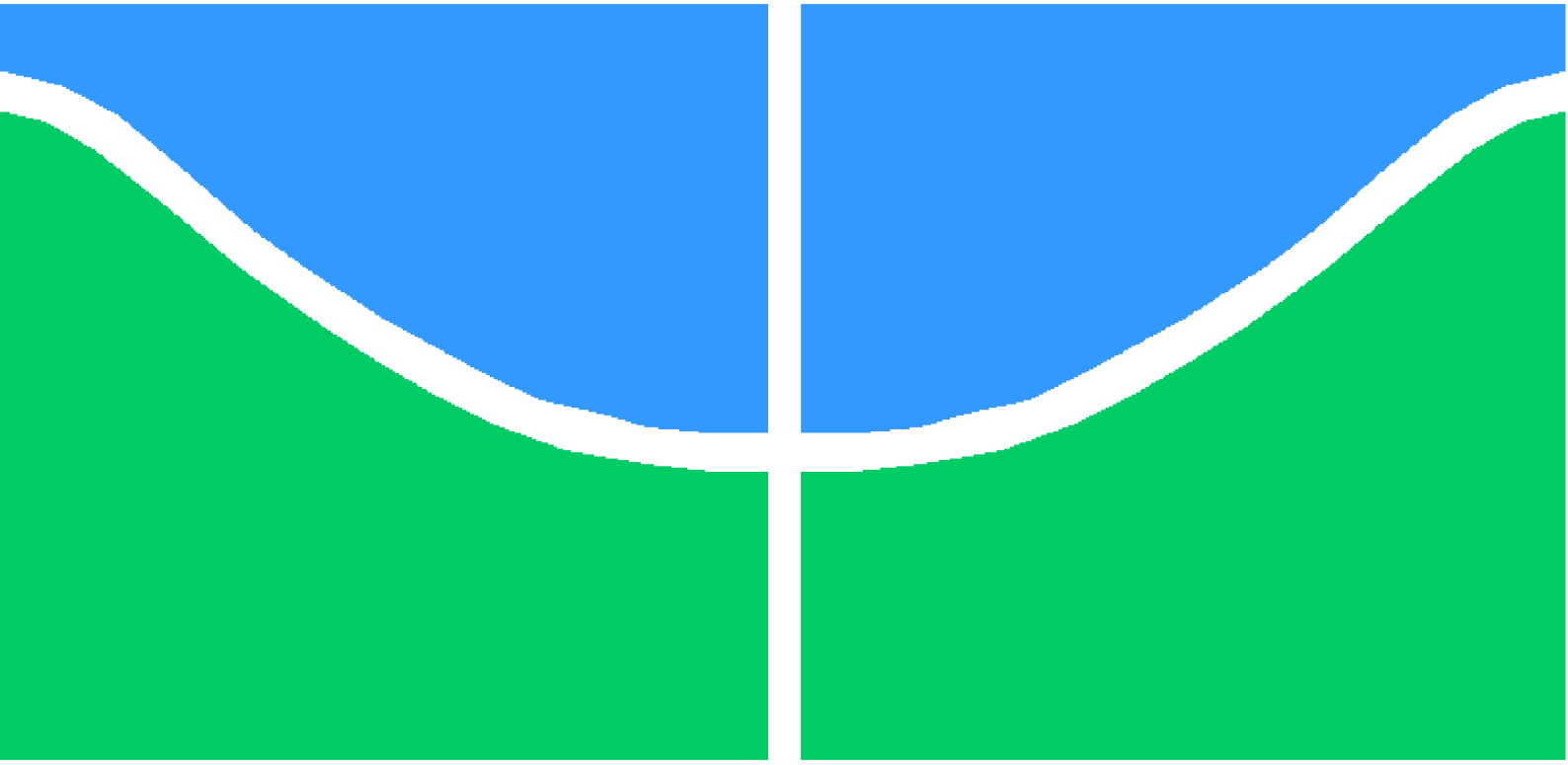

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Engenharia de Software

Trabalho Final - Verificação e Validação de Software

Autores: Karine Valença

Murilo Duarte

Tiago Assunção

Wilton Rodrigues

Professor: Ricardo Ajax

Brasília, DF

2016

Karine Valença
Murilo Duarte
Tiago Assunção
Wilton Rodrigues

Trabalho Final - Verificação e Validação de Software

Atividade submetida ao curso de graduação em (Engenharia de Software) da Universidade de Brasília, como requisito parcial para obtenção da aprovação em Verificação e Validacção de Software.

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Orientador: Ricardo Ajax

Brasília, DF

2016

Lista de abreviaturas e siglas

FS	Fábrica de *Software*
GQM	*Goal Question Metric*
UnB	Universidade de Brasília
EVeV	Equipe de Verificação e Validação
VeV	Verificação e Validação

Sumário

1 Introdução

O TEM-DF (Tranparência de Escalas Médicas do DF) é um software que visa apresentar as escalas dos médicos da rede pública de saúde do Distrito Federal, facilitando encontrar algum médico da especialidade adequada e na região desejada, além de permitir aos usuários realizarem reclamações ou elogios aos médicos.

O projeto foi desenvolvido por alunos da Faculdade do Gama - UnB, nas disciplinas de GPP e MDS e uma das restrições de qualidade do projeto é cobertura de testes acima de 90%. Porém, por ter sido desenvolvido por programadores iniciantes, boa parte dos testes, mesmo os que passaram, não eram de boa qualidade, ou seja os casos não estão contemplando o particionamento de equivalência e os valores limites. Além de conter apenas testes no nível unitário e funcional. Outro problema, é que durante o desenvolvimento, muito código era enviado ao repositório oficial sem testes e esse código não foi barrado ao ser enviado para as branchs principais do projeto.

Esse software pode ser acessado no seguinte endereço[1]

E sua documentação pode ser acessada neste endereço[2]

1.1 Problema

- Falta de Verificação dos Testes unitários no momento da integração da build.
- Baixa confiabilidade na qualidade dos testes unitários realizados.

1.2 Objetivos Gerais

- Verificar a integridade, analisando se todos os testes de unidade e aceitação passaram, de uma build de software a ser adicionada
- Melhorar a confiabilidade dos testes unitários realizados no software.

1.3 Objetivos Específicos

- Aplicar uma ferramenta de Integração contínua

[1] https://github.com/EscalaSaudeDF/TEM-DF

[2] http://lappis.unb.br/redmine/projects/grupo-1-tem-transparencia-das-escalas-dos-medicos-do-df/wiki

- Analisar os testes unitários no momento da integração

1.4 Questões de Pesquisa

- Como pode ser validado um novo incremento de software?
- Como garantir melhor qualidade dos testes unitários realizados?

1.5 Estratégia de Pesquisa

1.5.1 Revisão Sistemática de Literatura

- Verificação e validação de software
- Integração contínua de software
- Utilização da ferramenta travis para integração contínua de software
- Padrões de qualidade em testes unitários de software

1.5.2 Metodologia

- Definição de palavras-chave
- Gerar de uma string de busca
- Aplicação de string na base scopus
- Filtro de susca(análise do título, análise do abstract, referência)
- Snowball para trás

1.6 Resultados Esperados

1.6.1 Abordagem

- Qualitativas, tendo a confiabilidade de que os incrementos de software estão sendo testados corretamente.
- Quantitativa, o projeto deverá ter pelo menos 90% de cobertura de código de testes.

1.6.2 Métodos

- Aplicação de uma política de avaliação de qualidade estática de código contínua para validação de cada submissão de incremento de software.

1.6.3 Técnicas

- Integração Contínua

1.6.4 Ferramentas

Ferramentas a serem utilizadas e formas de estudos empíricos (Estudos de caso, questionários, entrevistas, estudos estatísticos, dentre vários outros possíveis).

- Travis[3]
- Estudo de Caso

[3] https://docs.travis-ci.com

2 Referencial Teórico

2.1 Verificação e validação

2.2 Qualidade de Testes

Quando se trata de manufatura ou produção de componentes físicos, os testes so bre a confiabilidade do que foi produzido é feito a partir de uma porcentagem do lote retirado do todo. Caso a quantidade returada do lote passe nos testes, ele é aprovado, caso não, todas peças são rejeitadas.

Ao se tratar de software, a maneira de executar os testes é bastante diferente. Cada software tem a sua peculiariedade e é desenvolvido de acordo com as capacidades dos desenvolvedores participantes do projeto. (Aditya), em seu estudo, afirma que antigamente eram realizados os testes com base nos testes de hardware. E, que esses, por sua vez, não eram realísticos.

Os softwares são desenvolvidos e uma bateria de testes primeiramente é definida no documento de casos de teste. Vários testes são executados de acordo com o sistema, podendo ser entrem eles testes de:

- Testes Unitários
- Testes de Integração
- Testes de Sistema
- Testes de Aceitação
- Testes de Instalação

Mas algumas perguntas que sempre entrigou os programadores responsáveis pelos testes são:

- O quanto eu devo testas de maneira que o software esteja suficientemente confiável?
- Quais são os pontos necessários a ponderar para entender o conceito de confiabilidade dos meus testes, por consequência, do meu software?

(Firmino) produziu um estudo que informa várias formas de se verificar a atestar a qualidade dos testes de software executados. E toda a sua maneira de identicar estes pontos é partindo de dois princípios:

- Avaliação do plano de testes, verificando se todos os requisitos foram descritos no caso de testes e se todos os testes e seus múltiplos casos foram contemplados na implementação.
- Associação dos testes cobrindo tudo o que foi implementado como requisito do sistema.

2.3 Integração Contínua

3 Desenvolvimento

TO DO

4 Resultados Obtidos

TO DO

5 Conclusão e Trabalhos Futuros

TO DO