



Universidade de Brasília - UnB
Faculdade UnB Gama - FGA
Engenharia de Software

Criação de um Dashboard para Monitoramento de Perfis de Qualidade de Software

Autor: Levi Moraes dos Santos

Orientador: Prof. Dr. Maurício Serrano
Coorientadora: Profa. Dra. Milene Serrano

Brasília, DF
2016



Levi Moraes dos Santos

Criação de um Dashboard para Monitoramento de Perfis de Qualidade de Software

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software .

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Orientador: Prof. Dr. Maurício Serrano

Coorientador: Profa. Dra. Milene Serrano

Brasília, DF

2016

Levi Moraes dos Santos

Criação de um Dashboard para Monitoramento de Perfis de Qualidade de Software / Levi Moraes dos Santos. – Brasília, DF, 2016-

37 p. : il. ; 30 cm.

Orientador: Prof. Dr. Maurício Serrano

Trabalho de Conclusão de Curso – Universidade de Brasília - UnB
Faculdade UnB Gama - FGA , 2016.

1. Qualidade. 2. Dashboard. I. Prof. Dr. Maurício Serrano. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. Criação de um Dashboard para Monitoramento de Perfis de Qualidade de Software

CDU 02:141:005.6

Levi Moraes dos Santos

Criação de um Dashboard para Monitoramento de Perfis de Qualidade de Software

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software .

Trabalho aprovado. Brasília, DF, 01 de junho de 2013:

Prof. Dr. Maurício Serrano
Orientador

Profa. Dra. Milene Serrano
Coorientadora

s
Convidado 1

Convidado 2

Brasília, DF
2016

• *Dedicatória ao Sr 1,2,3 de Oliveira 4*

Agradecimentos

Agradeço inicialmente a mim mesmo, porém agradecerei a mais pessoas quando me formar

Resumo

Palavras-chaves: qualidade. dashboard. monitoramento de métricas. visualização de métricas.

Abstract

This is the english abstract.

Key-words: latex. abntex. text editoration.

Lista de ilustrações

Figura 1 – Relação entre as NBR ISO/IEC 9126 e NBR ISO/IEC 14598 .Fonte:	
[NBR ISO/IEC 9126-1 2016]	25
Figura 2 – Modelo de Qualidade para Qualidade Interna e Externa .Fonte: [NBR	
ISO/IEC 9126-1 2016]	26

Lista de tabelas

Lista de abreviaturas e siglas

MIT	Massachusetts Institute of Technology
-----	---------------------------------------

Sumário

1	INTRODUÇÃO	21
1.1	Contextualização	21
1.2	Problema de pesquisa	22
1.3	Justificativa	22
1.4	Objetivos	22
1.4.1	Objetivos Gerais	22
1.4.2	Objetivos Específicos	23
1.5	Resultados Esperados	23
1.6	Organização do trabalho	23
2	REFERENCIAL TEÓRICO	25
2.1	Qualidade	25
3	SUPORTE TECNOLÓGICO	27
3.1	Ferramentas para Programação	27
3.1.1	GIT	27
3.1.2	Github	27
3.1.3	Bonita	27
3.1.4	Mac OS X	28
3.1.5	LaTeX	28
3.1.6	Sublime Text 3	28
3.1.7	Zotero	28
4	PROPOSTA	29
5	METODOLOGIA	31
5.1	Metodologia	31
6	RESULTADOS PARCIAIS	33
7	CONSIDERAÇÕES FINAIS	35
	REFERÊNCIAS	37

1 Introdução

Neste primeiro capítulo é apresentado uma visão mais ampla do trabalho e que tem como objetivo introduzir a temática abordada. Este capítulo está dividido em 6 (seis) seções. Na primeira seção é abordado o contexto (Contextualização) em que se encontra este trabalho. Seguido da contextualização apresenta-se a problemática (Problema) na qual se deseja resolver com a solução apresentada. As justificativas (Justificativa) que levaram à realização deste trabalho. Após as justificativas são apresentados os objetivos (Objetivos) que servem como guia para solução proposta. Resultados esperados (Resultados Esperados) que como o próprio nome já sugere apresenta o que é esperado ao fim deste trabalho e por último organização do Trabalho (Organização do Trabalho) que descreverá o conteúdo apresentado durante todo o trabalho.

1.1 Contextualização

O conceito de engenharia de software foi proposto inicialmente durante uma conferência na década de 60 em Garmisch na Alemanha. Nesta conferência estavam presentes usuarios, fabricantes e pesquisadores que debatiam sobre os constantes problemas no desenvolvimento de software [?]. A qualidade na produção de software é uma área muito ampla e que abrange desde qualidade da arquitetura de software até qualidade no processo REFERENCIA. Este trabalho teve como objetivo central apresentar uma solução para visualização de métricas de qualidade dentro de alguns órgãos, tendo como os principais pilares três áreas comuns da engenharia de software, gerência de configuração, integração contínua e análise estática de código. Uma arquitetura próxima a essa vem sendo trabalhada dentro de alguns órgãos públicos do governo federal, entre eles o Tribunal de Contas da União o qual tem mostrado os melhores resultados nesta área. O problema encontrado atualmente está na falta de um acompanhamento na qualidade dos chamados softwares legados, softwares que são produzidos dentro/para o órgão e que passado um tempo ainda estão em atividade.

Dentro da área de Tecnologia as organizações tem encontrado um grande problema quando se trata de softwares legados. O trabalho feito por REFERENCIA prova que os softwares legados são esquecidos pelas organizações quando se fala sob uma perspectiva de manutenção. Contudo o estudo também revela que em grande parte das empresas esses mesmos softwares continuam rodando no ambiente de produção.

1.2 Problema de pesquisa

O principal produto da engenharia de software é o software, contudo o que tem se vivenciado na realidade brasileira de computação é que o software que está sendo entregue é um software precário e de baixa qualidade. Por ser uma palavra abstrata, o conceito de qualidade é bem amplo, porém o termo qualidade normalmente está associado a uma medida relativa, essa qualidade pode ser entendida como “conformidade às especificações”. Conceituando dessa forma, a não conformidade às especificação é igual a ausência de qualidade.

Uma das grandes dificuldades nos órgãos públicos está no acompanhamento das manutenções prestadas por terceirizadas. Esse problema se agrava ainda mais quando a empresa contratante não consegue acompanhar ou não tem parametros concretos de indicadores de qualidade. Este trabalho tem como proposta a criação de uma dashboard de monitoramento para softwares legados onde é possível acompanhar de maneira simples e totalmente visual, indicadores de qualidade de código para projetos selecionados.

O grande desafio está em criar uma maneira de visualização de dados de maneira simplificada e objetiva para acompanhamento de software em um órgão público federal. Tomando este problema como base tem-se a seguinte questão de pesquisa:

"Uma vez definido os indicadores das métricas, como criar uma interface de visualização da informação? "

1.3 Justificativa

A motivação deste trabalho se deu como uma extensão de um trabalho de conclusão de curso elaborado anteriormente em que eram tratados aspectos para monitoramento da qualidade dentro de um órgão público federal. O trabalho abordava aspectos de contratação de software dentro desses órgãos e como acontecia o acompanhamento desses softwares e propunha uma solução com integração contínua e gerência de configuração. Após a conclusão do trabalho citado algumas lacunas continuaram, essas lacunas estão ligadas à apresentação destes dados para a equipe de gestão com o intuito de facilitar o acompanhamento das métricas.

1.4 Objetivos

1.4.1 Objetivos Gerais

Desenvolver uma solução intuitiva para monitoramento da qualidade de código de software

1.4.2 Objetivos Específicos

Para que seja possível alcançar o objetivo geral alguns outros objetivos menores precisam ser alcançados para garantir o objetivo geral

- Identificar métricas de código já existentes que mais se adequem às necessidades de um projeto de software
- Propor um ambiente integralizado e automatizado, englobando soluções de análise estática de código, integração contínua e versionamento de código
- Uma solução que agregue valor à equipe de manutenção

1.5 Resultados Esperados

1.6 Organização do trabalho

2 Referencial teórico

Este capítulo tem como objetivo servir como referencial teórico para todo o documento. As idéias discutidas neste capítulo são

2.1 Qualidade

O principal produto da engenharia de software é o software, contudo o que tem se vivenciado na realidade brasileira de computação é que o software que está sendo entregue é um software precário e de baixa qualidade. Por ser uma palavra abstrata, o conceito de qualidade é bem amplo, porém o termo qualidade normalmente está associado a uma medida relativa, essa qualidade pode ser entendida como "conformidade às especificações". Conceituando dessa forma, a não conformidade às especificação é igual a ausência de qualidade [?].

A ISO 9126-1 proposta em 2001 descreve o modelo de qualidade voltado para o produto de software como sendo composto por duas categorias como pode ser visto na Figura 1. A primeira categoria está relacionada a qualidade interna e a qualidade externa do software. A segunda categoria se relaciona com a qualidade de uso do software [NBR ISO/IEC 9126-1 2016]

Sob o aspecto de modelo de qualidade, a ISO 9126 classifica a qualidade interna do produto como sendo o somatório das características do ponto de vista interno do software. Os principais produtos desta categoria são os de cunho intermediário, entre eles: relatórios de análise estática do código fonte, revisão dos documentos produzidos, entre outros. A qualidade externa por sua vez já apresenta o seu foco mais voltado para as relações

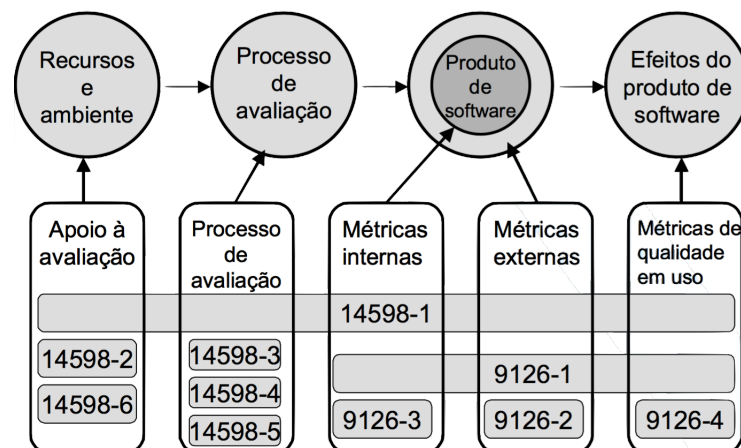


Figura 1 – Relação entre as NBR ISO/IEC 9126 e NBR ISO/IEC 14598 .Fonte: [NBR ISO/IEC 9126-1 2016]

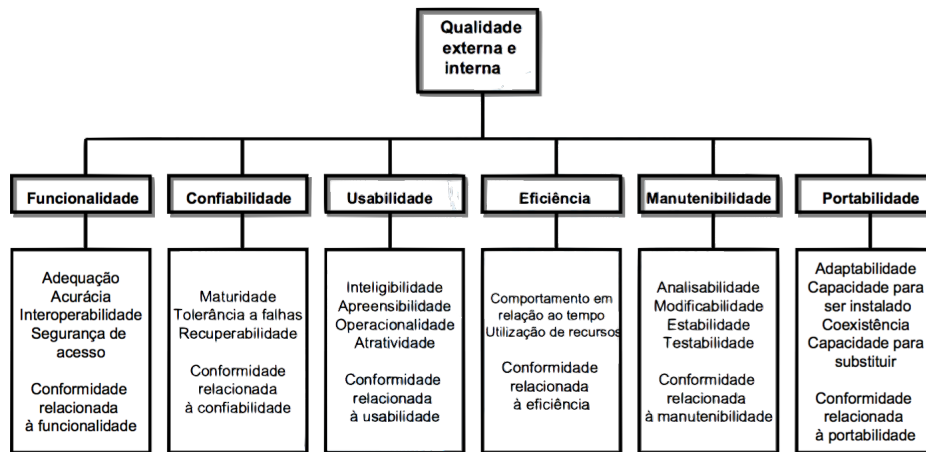


Figura 2 – Modelo de Qualidade para Qualidade Interna e Externa .Fonte: [NBR ISO/IEC 9126-1 2016]

externas do software, normalmente esta relacionado com a execução do código coletando suas métricas enquanto o software está em funcionamento. A Figura 2 apresenta a divisão proposta pela [NBR ISO/IEC 9126-1 2016] onde são categorizados seis aspectos de qualidade de software e suas subcaracterísticas, essas podem ser medidas por meio de métricas internas e externas.

Segundo a ISO 9126 essas características podem ser definidas como:

- **Funcionalidade:** Capacidade do produto de software de prover funções que atendam às necessidades explícitas e implícitas, quando o software estiver sendo utilizado sob condições especificadas.
- **Confiabilidade:** Capacidade do produto de software de manter um nível de desempenho especificado, quando usado em condições especificadas.
- **Usabilidade:** Capacidade do produto de software de ser compreendido, aprendido, operado e atraente ao usuário, quando usado sob condições especificadas.
- **Eficiência:** Capacidade do produto de software de apresentar desempenho apropriado, relativo à quantidade de recursos usados, sob condições especificadas.
- **Manutenibilidade:** Capacidade do produto de software de ser modificado. As modificações podem incluir correções, melhorias ou adaptações do software devido a mudanças no ambiente e nos seus requisitos ou especificações funcionais.
- **Portabilidade:** Capacidade do produto de software de ser transferido de um ambiente para outro.

3 Suporte Tecnológico

O objetivo desta seção é explicitar todo o aparato tecnológico a nível de software que foi utilizado durante o desenvolvimento deste trabalho. Esta seção está dividida em *Ferramentas para programação*.

3.1 Ferramentas para Programação

Neste tópico, serão apresentadas ferramentas e tecnologias voltadas ao contexto da Engenharia de Software que são utilizadas durante este trabalho, como, por exemplo, ferramentas para gerência de configuração e versionamento dos artefatos gerados.

3.1.1 GIT

A ferramenta GIT¹ foi desenvolvida por Linus Torvalds durante a criação do Kernel Linux, pois Linus percebeu que existia a necessidade de criar uma ferramenta *open-source* que fizesse o controle de versão [BENTO et al. 2013].

O motivo para escolha da ferramenta se deve ao fato de que o Git contém o suporte para desenvolvimento linear o que garante um paralelismo de diversas áreas do desenvolvimento. Outro diferencial do Git está nos *snapshots* dos objetos que são armazenados, isso significa que o Git não rearmazena arquivos que não foram alterados [MARTINHO e MUNIZ 2013].

3.1.2 Github

O Github² é um repositório *online* que fornece a criação de projetos públicos gratuitos. A ferramenta também provê um sistema de gestão para acompanhamento do desenvolvimento envolvendo um sistema de *logs*, gráficos de visualização e uma *Wiki* integrada a cada projeto [MARTINHO e MUNIZ 2013].

O principal motivo pela escolha do Github é que deseja-se disponibilizar a solução futuramente para consulta e aprimoramento de pessoas interessadas.

3.1.3 Bonita

A ferramenta Bonita³ foi escolhida graças a sua facilidade de utilização e portabilidade para o sistema operacional Mac OS X e o fato de ser gratuita. Esta ferramenta

¹ <https://git-scm.com/>

² <https://github.com>

³ <http://www.bonitasoft.com/>

auxilia na modelagem de processos.

3.1.4 Mac OS X

O sistema operacional Mac OS foi baseado no kernel Unix e fabricado e desenvolvido pela empresa Apple Inc. Utilizou-se a versão 10.11 do sistema também conhecida como "*El Capitan*".

3.1.5 LaTeX

O LaTeX⁴ foi desenvolvido na década de 80 cujo objetivo era simplificar a diagramação de textos científicos e matemáticos, onde atualmente dispõe de uma grande quantidade de macros para bibliografia, referencias, gráficos entre outros.

3.1.6 Sublime Text 3

O Sublime Text 3⁵ é um editor de texto bastante utilizado por programadores, por possuir apoio para diversas linguagens de programação, incluindo textos em LaTeX.

3.1.7 Zotero

Zotero⁶ é um software para gerenciamento de referências bibliográficas. Ele possui integração com o *browser*, sincronização online e criação de bibliografias estilizadas.

⁴ <https://www.latex-project.org/>

⁵ <https://www.sublimetext.com/3>

⁶ <https://www.zotero.org>

4 Proposta

5 Metodologia

5.1 Metodologia

A partir da Metodologia

6 Resultados Parciais

7 Considerações finais

Considero nada a ninguém

Referências

BENTO, G. V. et al. Análise Comparativa de Sistemas de Controle de Versões com foco em Versionamento de Banco de Dados Oracle. *e-RAC*, v. 3, n. 1, 2013. Citado na página 27.

MARTINHO, M.; MUNIZ, E. GIT SCM: Sistema de Controle de Versionamento Distribuído. *GIT SCM: Sistema de Controle de Versionamento Distribuído*, 2013. Citado na página 27.

NBR ISO/IEC 9126-1. [S.l.], 2016. Citado 3 vezes nas páginas 13, 25 y 26.