

Um Estudo de Ferramentas de Gerenciamento de Requisição de Mudança

Julho de 2017

Vagner Clementino
Rodolfo Resende - Orientador

Departamento de Ciência da Computação
Universidade Federal de Minas Gerais

Agenda

Contexto

Agenda

Contexto

Problema

Agenda

Contexto

Problema

Objetivos

Agenda

Contexto

Problema

Objetivos

Atividades

Agenda

Contexto

Problema

Objetivos

Atividades

Resultados

Agenda

Contexto

Problema

Objetivos

Atividades

Resultados

Discussão

Agenda

Contexto

Problema

Objetivos

Atividades

Resultados

Discussão

Ameaças à Validade

Agenda

Contexto

Problema

Objetivos

Atividades

Resultados

Discussão

Ameaças à Validade

Conclusões e Trabalhos Futuros

[Contexto] Importância da Manutenção de Software

- ▶ Dentro do ciclo de vida do software o processo de Manutenção de Software tem papel fundamental.
 - ▶ Evolução do software (Leis de Lehman [Lehman, 1980]).
 - ▶ Correção de falhas
 - ▶ Alto custo, que pode variar entre 60% e 90% do preço final [Kaur and Singh, 2015].

[Contexto] Conceito de Manutenção de Software

- ▶ A **Manutenção de Software** é o processo de modificar um componente ou sistema de software após a sua entrega com o objetivo de *corrigir falhas, melhorar o desempenho ou adaptá-lo devido à mudanças ambientais* [IEEE, 1990].
- ▶ Com a adoção das práticas propostas pelos agilistas essa definição pode não ser adequada em determinados contextos.

[Contexto] Tipos de Manutenção de Software

- ▶ A Manutenção de Software pode ser dividida em *Corretiva*, *Adaptativa*, *Perfectiva* e *Preventiva* [Lientz and Swanson, 1980, IEEE, 1990].
- ▶ A *ISO 14764* [ISO/IEC, 2006] propõe que exista um elemento denominado **Requisição de Mudança (RM)** que corresponde a uma agregação de características das quatro categorias.

[Contexto] Tipos de Manutenção em Software

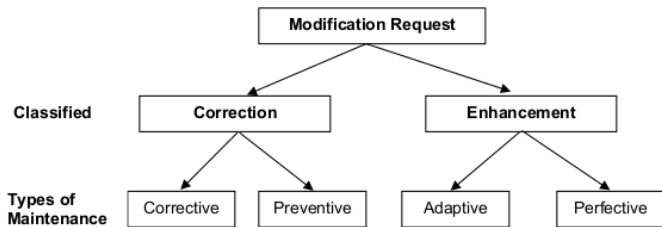


Figura 1: Tipos de manutenção segundo a norma ISO/IEC 14764 [ISO/IEC, 2006]

[Contexto] Modelo Conceitual de uma RM

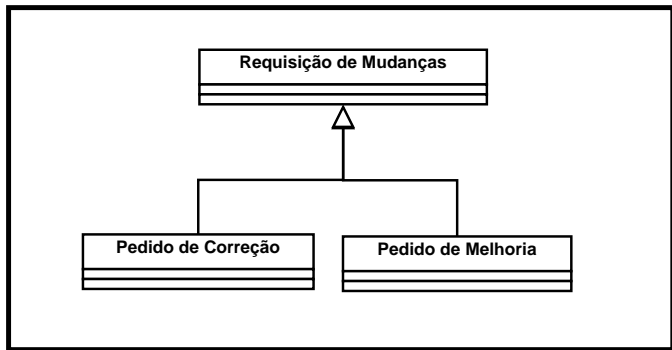


Figura 2: Modelo conceitual de uma Requisição de Mudança. Baseado em Tripathy & Naik [Tripathy and Naik, 2015].

[Contexto] Atributos de uma RM

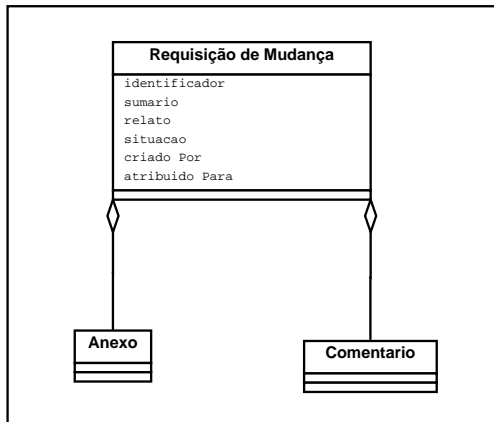


Figura 3: Modelo de uma RM como diagrama UML. Adaptado de Singh & Chaturvedi [Singh and Chaturvedi, 2011]

[Contexto] Exemplo de uma RM

Bug 105832 - Dead Lock in DeltaProcessor.resourceChanged

Status: NEW
Fixed: YES
Component: Core
Version: 3.6
Hardware: PC Windows Vista
Installation: P2 major with 1 sub minor
Target Milestone: ---
Assigned To: [jira@apache.org](#)
QA Contact: ---
URL: ---
Whiteboard: ---
Keywords: ---
Tags: ---
Depends on: 249551
Blocks: ---
[Show dependency logs](#)

Reported: 2010-03-15 06:55 EDT by [jira@apache.org](#)
Modified: 2011-09-03 13:03 EDT ([history](#))
CC: [mailto:jira@apache.org](#)
4 users ([view](#))

See Also: ---
Flags: None yet set ([add flag](#))

Attachments:
[Stack Trace \(text/plain\)](#) [Add](#) [Open](#) [Details](#)
[Stack Trace \(text/plain\)](#) [Add](#) [Open](#) [Details](#)
[Stack Trace \(text/plain\)](#) [Add](#) [Open](#) [Details](#)

Additional Comments:

Status: NEW [Show Changes](#)

Stack Trace (text/plain) 2010-03-15 06:55 EDT [Details](#) [Stack Trace](#) [Comments](#) [Add Comment](#)

1. java.lang.Thread: 2010-03-15 06:55 EDT
2. java.lang.Thread: 2010-03-15 06:55 EDT
3. java.lang.Thread: 2010-03-15 06:55 EDT
4. java.lang.Thread: 2010-03-15 06:55 EDT
5. java.lang.Thread: 2010-03-15 06:55 EDT
6. java.lang.Thread: 2010-03-15 06:55 EDT
7. java.lang.Thread: 2010-03-15 06:55 EDT
8. java.lang.Thread: 2010-03-15 06:55 EDT

Figura 4: Modelo de uma RM baseado em GUI.

[Contexto] Problemas e Desafios Relacionados com as RMs

- ▶ Localização do Problema
- ▶ Baixa Qualidade do Relato
- ▶ Identificação de RMs Duplicadas
- ▶ Atribuição (Triagem) de RM
- ▶ Classificação da RM
- ▶ Estimativa de Esforço da RM

[Contexto] Papéis na Manutenção de Software

Nesta dissertação, utilizamos a classificação proposta por Polo e outros [Polo et al., 1999]:

- ▶ Usuário Afetado
- ▶ Reportador
- ▶ Gerente de Requisição de Mudança
- ▶ Agente de Triagem
- ▶ Desenvolvedor
- ▶ Analista de Qualidade
- ▶ Chefe da Manutenção

[Contexto] Volume de RMs do Editor Emacs

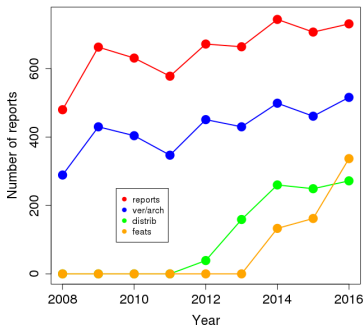


Figura 5: Número de RMs por ano¹.

¹<https://debugs.gnu.org/stats/emacs.html>

[Contexto] Ferramentas de Gerenciamento de Requisição de Mudança (FGRM)

- ▶ Dependendo do tamanho do projeto de software é necessário a utilização de uma **FGRM** para gerenciar as suas requisições de mudança.
- ▶ As **FGRMs** proporcionam um espaço único onde as partes interessadas podem registrar as falhas e as melhorias.

[Contexto] Exemplos de FGORMs



Figura 6: Exemplos de FGORMs disponíveis comercialmente ou por código aberto.

[Contexto] Além do Gerenciamento de RMs

- ▶ As FGRRMs gerenciam RMs e também:
 - ▶ Proporciona um ponto central para a comunicação e coordenação [Bertram et al., 2010].
 - ▶ Possibilita a participação do processo de solução das RMs [Breu et al., 2010].
 - ▶ Auxilia em atividades tais como [Cavalcanti et al., 2013]:
 - ▶ estimativa de custo
 - ▶ planejamento do projeto
 - ▶ extração de conhecimento

Problema

- ▶ Desacoplamento das funcionalidades das FGRLs com as necessidades de seus usuários [Baysal and Holmes, 2012, Just et al., 2008].
- ▶ Os desenvolvedores têm dificuldade em manter uma compreensão global das RLs com as quais estão envolvidos [Baysal et al., 2013].
- ▶ Extensões (plugins) propostas na literatura [Rocha et al., 2015, Thung et al., 2014, Kononenko et al., 2014].

Objetivos

- ▶ Elaboramos um estudo sobre as FGRMs com os seguintes objetivos:
 - (i) analisar as funcionalidades oferecidas por este tipo de ferramenta;
 - (ii) mapear as melhorias que estão sendo propostas na literatura;
 - (iii) verificar como os profissionais avaliam as funcionalidades das ferramentas que têm contato;
 - (iv) propor melhorias para as funcionalidades das FGRMs.

Atividades

- ▶ Estudo sobre as funcionalidades das FGRRMs
- ▶ Mapeamento Sistemático da Literatura [Petersen et al., 2008]
- ▶ Levantamento (Survey) com desenvolvedores [Wohlin et al., 2012]
- ▶ Sugestões de melhorias para as FGRRMs
- ▶ Implementação de extensão para FGRRM

[Atividades] Estudo sobre as funcionalidades das FGRMs

- ▶ Etapas do estudo
 - (i) Seleção das Ferramentas
 - (ii) Inspeção da Documentação
 - (iii) Agrupamento das Funcionalidades

[Atividades] Estudo sobre as funcionalidades das FGRMs

- ▶ Seleção das Ferramentas
 - ▶ Levantamento por Questionário
 - ▶ Dois grupos de participantes
 - ▶ 52 participações
 - ▶ 06 ferramentas escolhidas

[Atividades] Estudo sobre as funcionalidades das FGRMs

- ▶ Inspeção da Documentação
 - ▶ Leitura do material disponível na Internet
 - ▶ As funcionalidades foram classificadas através da técnica de *Cartões de Classificação - Sorting Cards* [Just et al., 2008].

[Atividades] Estudo sobre as funcionalidades das FGRMs

- ▶ Agrupamento das Funcionalidades
 - ▶ Análise Individual: O autor e um outro especialista realizam de forma separada os agrupamentos.
 - ▶ Análise Compartilhada: Em um segundo momento tanto o autor quanto o especialista discutem as possíveis divergências até que um consenso seja obtido.

[Atividades] Mapeamento Sistemático da Literatura

- ▶ Mapeamento com base nas diretrizes propostas por Petersen e outros [Petersen et al., 2008].
- ▶ Questões de Pesquisa
 - ▶ *Questão 01*: Quais as melhorias e novas funcionalidades estão sendo propostas para as FGRM?
 - ▶ *Questão 02*: Quais papéis envolvidos no processo de manutenção de software as melhorias das funcionalidades visam dar suporte?

[Atividades] Mapeamento Sistemático da Literatura

- ▶ Os estudos primários coletados das bases de pesquisa *IEEE Explore*, *ACM Digital Library*, *Scopus*, e *Inspec/Compendex*.
- ▶ As sentenças de buscas foram produzidas com base na metodologia PICO (Population, Intervention, Comparison and Outcomes) [Keele, 2007].

[Atividades] Mapeamento Sistemático da Literatura

- ▶ Os estudos foram divididos pela pertinência do com as seguintes categorias:
 - (i) *dimensões de melhoria*, conforme proposto por Zimmermann e outros [Zimmermann et al., 2009];
 - (ii) *função desempenhada* no processo de manutenção de software, a partir de um conjunto de papéis desempenhados por pessoas envolvidas com Manutenção e discutidos por Polo e outros [Polo et al., 1999].

[Atividades] Mapeamento Sistemático da Literatura

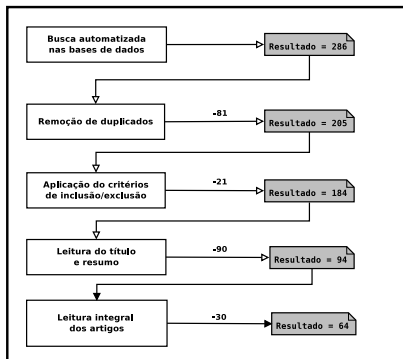


Figura 7: Número de artigos incluídos durante o processo de seleção dos estudos. Figura baseada em [Petersen et al., 2015]

[Atividades] Levantamento com Desenvolvedores

- ▶ Questão 01: Qual a opinião dos profissionais envolvidos em manutenção de software com relação as funcionalidades oferecidas pelas FGRM?
- ▶ Questão 02: Na visão dos profissionais envolvidos em manutenção de software quais das melhorias nas funcionalidades das FGRMs propostas na literatura teriam maior relevância em suas atividades?

[Atividades] Levantamento com Desenvolvedores

- ▶ Questão 03: As práticas propostas pelos agilistas estão sendo utilizadas no processo de manutenção de software?
- ▶ Questão 04: Como as FGRLs podem ajudar as equipes de manutenção na adoção das práticas propostas pelos agilistas?

[Atividades] Levantamento com Desenvolvedores

- ▶ Fonte de Amostragem corresponde a um banco de dados em que um subconjunto válido da população pode ser recuperado. [de Mello et al., 2014].

Identificador	Fonte de Amostragem	URL
FA01	Python	https://bugs.python.org/
FA02	Stack Overflow	https://stackoverflow.com

Tabela 1: Fontes de Amostragem utilizadas no estudo

[Atividades] Levantamento com Desenvolvedores

- ▶ Formulário preenchido por 85 participantes

Função Desempenhada	Total
Desenvolvedor	23
Engenheiro de Software	17
Gerente	12
Arquiteto de Software	5
Pesquisador	5
Consultor	4
Estudante	3
Analista de Qualidade	1
Designer	1

Tabela 2: Função desempenhada pelos participantes

[Atividades] Sugestões de Melhorias

- ▶ Sugestões foram compiladas utilizando a literatura da área e os levantamentos realizados nesta dissertação, especialmente com Mapeamento Sistemático e Levantamento com Profissionais;
- ▶ E nos estudos que propõem melhorias para as FG RM [Zimmermann et al., 2009, Bettenburg et al., 2008, Singh and Chaturvedi, 2011].

[Atividades] Sugestões de Melhorias

- ▶ Propostas melhorias nomeadas como Sugestão #1 até Sugestão #8.
- ▶ Avaliadas com um levantamento com profissionais que contribuem em projetos hospedados no Github.

Projeto	Participantes
DEBBUGS	4
MANTISBT	4
TRAC	4
FOSSIL	3
BUGZILLA	2
REDMINE	2
OUTROS	6

Tabela 3: Projetos que os participantes contribuem.

[Atividades] Implementação de Extensão

- ▶ Implementação da Sugestão #1 na plataforma Github.
- ▶ Cliente para API do Github² que possibilita analisar a qualidade da informação fornecida no relato.
- ▶ Batizada de *IssueQuality*

Sugestão #01: As FGRMs devem fornecer realimentação (feedback) relacionado com a qualidade do texto relatado.

²<https://api.github.com/>

[Atividades] Implementação de Extensão

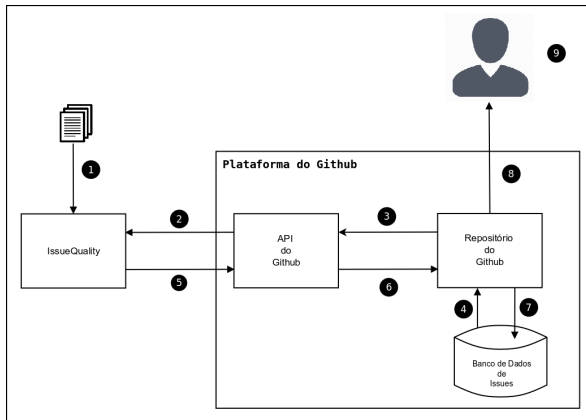


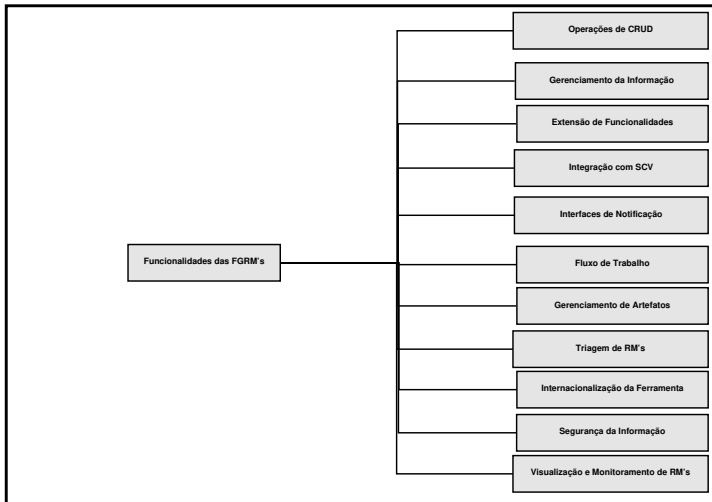
Figura 8: Visão geral do funcionamento da extensão *IssueQuality*

[Resultados] Estudo sobre as funcionalidades das FGRMs

Ferramenta	Classificação	Versão	URL
Bugzilla	Ferramenta	5.0.3	https://www.bugzilla.org
Mantis Bug Tracker	Ferramenta	1.3.2	https://www.mantisbt.org
Redmine	Ferramenta	3.3.1	http://www.redmine.org/
JIRA Software	Serviço	7.2.4	https://br.atlassian.com/software/jira
Github Issue System	Serviço	-	https://github.com/
Gitlab Issue Tracking System	Serviço	-	https://gitlab.com/

Tabela 4: Ferramentas utilizados no estudo sobre funcionalidades das FGRMs

[Resultados] Estudo sobre as funcionalidades das FGRMs



[Resultados] Mapeamento Sistemático da Literatura

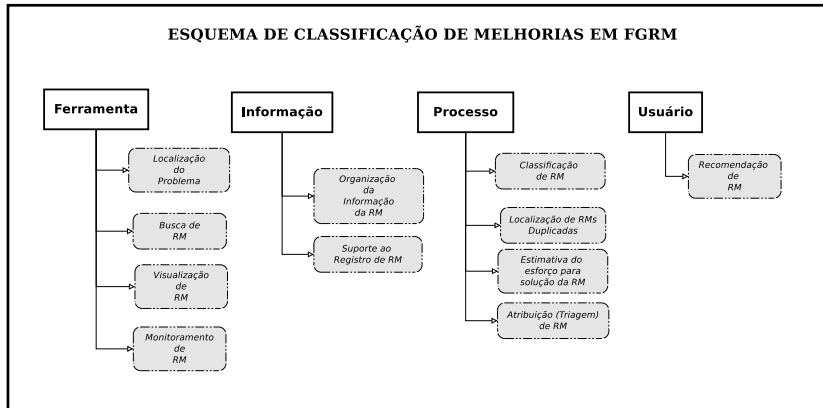


Figura 10: Esquema de classificação das melhorias propostas na literatura

[Resultados] Mapeamento Sistemático da Literatura

Papel	Total de Artigos
Agente de Triagem	37
Desenvolvedor	26
Analista de Qualidade	13
Gerente de Requisição de Mudança	11
Reportador	6
Líder da Manutenção	4
Todos	3

Tabela 5: Total de artigos por papel na manutenção de software

[Resultados] Levantamento com Desenvolvedores

Nível de Satisfação	Total
Nem satisfeito ou insatisfeito	32
Satisfeito	21
Muito Satisfeito	11
Desapontado	6
Muito Desapontado	2

Tabela 6: Nível de satisfação com as Ferramentas

[Resultados] Levantamento com Desenvolvedores

Recomendação da Ferramenta Utilizada	Total
Provável	32
Com certeza	22
Pouco Provável	9
Não	5
Muito Improvável	3

Tabela 7: Probabilidade de Recomendação da Ferramenta Utilizada

[Resultados] Levantamento com Desenvolvedores

Funcionalidade	Total
Identificação de RMs Duplicadas	42
Atribuição Automática de RM	20
Análise de Impacto de RM	18
Análise Automática de Esforço	15
Não sente falta de nenhuma	13
Análise da Qualidade do Relato	13
Suporte à Integração Contínua	4
RM's trabalhadas anteriormente	1

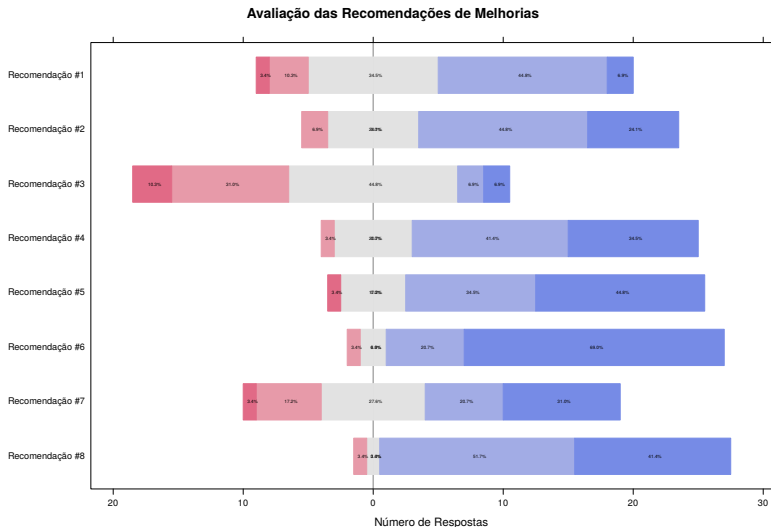
Tabela 8: Funcionalidades que o participantes sentem falta.

[Resultados] Levantamento com Desenvolvedores

Melhorias Propostas	Classificação
Priorização automatizada de RMs urgentes e inesperadas	1
Sugestão automatizada das RMs que farão parte da iteração.	2
Suporte aos desenvolvedores na preparação para reunião diária	3
Suporte à divisão de tarefas de forma compartilhada	4
Facilitar a propriedade compartilhada de código	5

Tabela 9: Classificação das funcionalidades que possam dar suporte ao uso das metodologias dos agilistas.

[Resultados] Sugestões de Melhorias

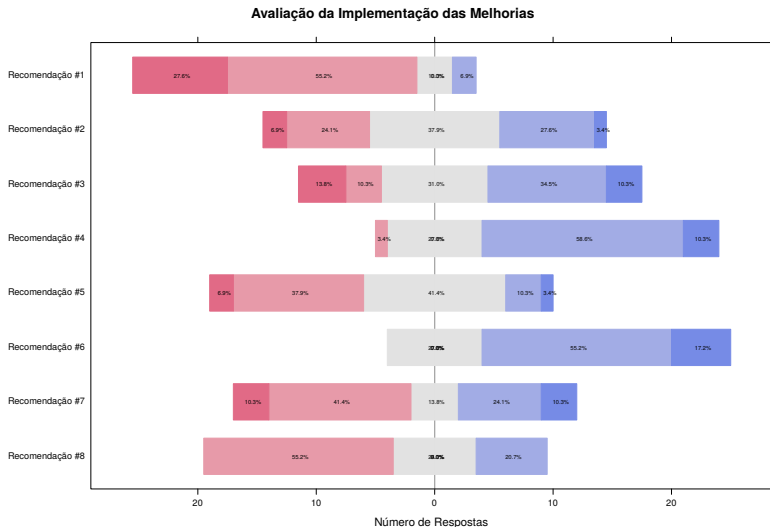


[Resultados] Sugestões de Melhorias

Recomendações	Discordo Fortemente	Discordo	Não concordo e nem discordo	Concordo	Concordo Fortemente	Ranking
<i>Sugestão #6</i>	0	1	2	6	20	45
<i>Sugestão #8</i>	0	1	1	15	12	38
<i>Sugestão #5</i>	1	0	5	10	13	34
<i>Sugestão #4</i>	0	1	6	12	10	31
<i>Sugestão #2</i>	0	2	7	13	7	25
<i>Sugestão #7</i>	1	5	8	6	9	17
<i>Sugestão #1</i>	1	3	10	13	2	12
<i>Sugestão #3</i>	3	9	13	2	2	-9

Tabela 10: Ranking das sugestões propostas

[Resultados] Sugestões de Melhorias



[Resultados] Sugestões de Melhorias

Recomendações	Muito Difícil	Difícil	Neutro	Fácil	Muito Fácil	Ranking
Sugestão #6	0	0	8	16	5	26
Sugestão #4	0	1	8	17	3	22
Sugestão #3	4	3	9	10	3	5
Sugestão #2	2	7	11	8	1	-1
Sugestão #7	3	12	4	7	3	-5
Sugestão #5	2	11	12	3	1	-10
Sugestão #8	0	16	7	6	0	-10
Sugestão #1	8	16	3	2	0	-30

Tabela 11: Ordenamento das sugestões pelo grau de dificuldade.

[Discussão] Estudo sobre as Funcionalidades

- ▶ As FGRMs evoluíram da gerência simples de RMs para colaborar no processo de desenvolvimento e manutenção do software.
- ▶ Seria importante que as FGRMs incorporassem outros comportamentos: busca de duplicados, melhoria da qualidade do relato e atribuição e classificação automatizadas das RMs.

[Discussão] Mapeamento Sistemático da Literatura

- ▶ Prevalência de estudos na dimensão *Processo* especialmente para os tópicos de *Localização de RMs Duplicadas*, *Atribuição (Triagem) de RMs* e *Classificação de RMs*, respectivamente.
- ▶ Total de 10 estudos foram implementados como extensões ou protótipos, este número poderia ser maior.

[Discussão] Mapeamento Sistemático da Literatura

- ▶ Prevalência de estudos com foco no papel de *Agente de Triage*. Existe possivelmente uma crença de que é possível melhorar a produtividade do processo de manutenção de software reduzindo o esforço de encontrar o desenvolvedor mais apto.
- ▶ As FGRMs deveriam dar suporte ao Reportador que, na maioria da vezes, é o primeiro a registrar as informações que serão necessárias à solução da RM.

[Discussão] Levantamento com Desenvolvedores

- ▶ Em geral, o nível de satisfação com as funcionalidades oferecidas pelas FGRMs é alto.
- ▶ As funcionalidades que os participantes mais sentiram falta, também representam a maior quantidade de estudos na literatura.
- ▶ As FGRMs poderiam oferecer suporte às praticas propostas pelos agilistas.

[Discussão] Sugestões de Melhorias

- ▶ Em geral podemos considerar que as sugestões propostas tiveram uma boa aceitação dos participantes.
- ▶ Com relação às recomendações propostas, verificamos que a utilização de uma linguagem além do texto simples, como por exemplo o Markdown, foi muito bem aceita.

[Discussão] Sugestões de Melhorias

- ▶ O suporte à tarefas compartilhadas (sugestão #8) também foi muito bem aceita.
- ▶ Por outro lado, as sugestões que têm algum tipo de relação com a interface das FGRMs (sugestões #6, #4, #3 e #2) foram consideradas como mais “fácil” de implementar.

[Ameaças à Validade] Estudo sobre as Funcionalidades

- ▶ Uma ameaça à validade do trabalho está no processo de seleção das ferramentas.
- ▶ Como a extração dos dados dos Cartões foi realizada de forma manual pode ter ocorrido algum tipo de equívoco no processo, como por exemplo a não coleta de algum dado de determinada ferramenta por algum descuido.
- ▶ A classificação dos cartões pode ter falhas o que pode acarretar em limitação dos resultados apresentados.

[Ameaças à Validade] Mapeamento Sistemático da Literatura

- ▶ É possível que as perguntas de pesquisa definidas possam não abranger completamente o campo de investigação sobre as funcionalidades das FGRLs.
- ▶ As sentenças de busca podem não ser as mais adequadas para a pesquisa do maior número de documentos relevantes.

[Ameaças à Validade] Levantamento com Desenvolvedores

- ▶ Uma ameaça à validade deste trabalho está no número de respondentes da pesquisa.
- ▶ A amostragem de conveniência implica que as generalizações são limitadas já que a amostra pode não representar a população.
- ▶ Não temos garantias que as regras para seleção de participantes resultaram em um conjunto bem representativo da população.

[Ameaças à Validade] Sugestões de Melhorias

- ▶ O total de participantes não nos permite extrapolar os resultados para todos os contextos em que as FGRMs estão inseridas.
- ▶ A utilização de apenas projetos públicos hospedados no Github pode ter causado algum tipo de direcionamento, como por exemplo foco em projetos de código aberto.

[Ameaças à Validade] Sugestões de Melhorias

- ▶ A estrutura das perguntas do formulário pode ter causado impacto na quantidade de respostas ou na opção escolhida pelos participantes.
- ▶ Acreditávamos que a escolha destas 08 sugestões envolveria as principais características, mas existe possibilidade de ficar alguma característica relevante de fora.

Conclusão e Trabalhos Futuros

- ▶ A contribuição deste trabalho de dissertação está na proposição de melhorias para as FGRMs tomando como base a literatura da área e a opinião de profissionais.
- ▶ Em algumas plataformas, tais como o Github e o Gitlab, foi possível perceber a tendência em que não existe uma clara separação entre o gerenciamento das RMs e o controle de versão do código.

Conclusão e Trabalhos Futuros

- ▶ Foi possível verificar um desacoplamento entre as necessidades dos desenvolvedores e o que está sendo proposto na literatura.
- ▶ As metodologias propostas pelos agilistas vêm sendo adotadas por algumas equipes de manutenção de software. As FGRRMs poderiam implantar funcionalidades com o objetivo de suportar algumas destas práticas.

Conclusão e Trabalhos Futuros

- ▶ Entendemos que seria importante a condução de um novo trabalho com o objetivo de descrever e avaliar os papéis realizados no processo de manter um software.
- ▶ Entendemos que seria importante a realização de um estudo com o objetivo de melhorar a organização dos conceitos da área de Manutenção de Software, em especial sobre as RMs e FGRLMs.

Conclusão e Trabalhos Futuros

- ▶ O processo de criação de RMs poderia ser melhorado com a utilização de uma interface que utilize um *chatbot* [Mauldin, 1994, Huang et al., 2007], permitindo a criação iterativa de uma RM.

Disponibilidade dos Dados e Artefatos

- ▶ Os dados e artefatos utilizados neste trabalho de dissertação estão disponíveis em `https://goo.gl/bj3pBY`

Dúvidas?



References I

[Baysal and Holmes, 2012] Baysal, O. and Holmes, R. (2012).

A Qualitative Study of Mozillas Process Management Practices.

*David R. Cheriton School of Computer Science,
University of Waterloo, Waterloo, Canada, Tech.
Rep. CS-2012-10.*

References II

[Baysal et al., 2013] Baysal, O., Holmes, R., and Godfrey, M. W. (2013).

Situational awareness: Personalizing issue tracking systems.

In *Proceedings of the 2013 International Conference on Software Engineering, ICSE '13*, pages 1185–1188, Piscataway, NJ, USA. IEEE Press.

References III

- [Bertram et al., 2010] Bertram, D., Voida, A., Greenberg, S., and Walker, R. (2010). Communication, collaboration, and bugs: The social nature of issue tracking in small, collocated teams.
In Proceedings of the 2010 ACM Conference on Computer Supported Cooperative Work, CSCW '10, pages 291–300, New York, NY, USA. ACM.

References IV

[Bettenburg et al., 2008] Bettenburg, N., Just, S., Schröter, A., Weiss, C., Premraj, R., and Zimmermann, T. (2008).

What makes a good bug report?

In *Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of software engineering*, pages 308–318. ACM.

References V

[Breu et al., 2010] Breu, S., Premraj, R., Sillito, J., and Zimmermann, T. (2010).

Information needs in bug reports: Improving cooperation between developers and users.

In *Proceedings of the 2010 ACM Conference on Computer Supported Cooperative Work, CSCW '10*, pages 301–310, New York, NY, USA. ACM.

References VI

[Cavalcanti et al., 2013] Cavalcanti, Y. C., Neto, P. A. d. M. S., Lucrédio, D., Vale, T., de Almeida, E. S., and de Lemos Meira, S. R. (2013).

The bug report duplication problem: an exploratory study.

Software Quality Journal, 21(1):39–66.

References VII

- [de Mello et al., 2014] de Mello, R. M., da Silva, P. C., Runeson, P., and Travassos, G. H. (2014). Towards a framework to support large scale sampling in software engineering surveys. In *Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, page 48. ACM.

References VIII

[Huang et al., 2007] Huang, J., Zhou, M., and Yang, D. (2007).

Extracting chatbot knowledge from online discussion forums.

In *IJCAI*, volume 7, pages 423–428.

[IEEE, 1990] IEEE (1990).

IEEE Standard Glossary of Software Engineering Terminology.

IEEE Std 610.12-1990, pages 1–84.

References IX

[ISO/IEC, 2006] ISO/IEC (2006).

International Standard - ISO/IEC 14764 IEEE Std 14764-2006 Software Engineering 2013; Software Life Cycle Processes 2013; Maintenance.

ISO/IEC 14764:2006 (E) IEEE Std 14764-2006 Revision of IEEE Std 1219-1998), pages 01–46.

References X

- [Just et al., 2008] Just, S., Premraj, R., and Zimmermann, T. (2008).
Towards the next generation of bug tracking systems.
In *2008 IEEE Symposium on Visual Languages and Human-Centric Computing*, pages 82–85. IEEE.

References XI

[Kaur and Singh, 2015] Kaur, U. and Singh, G.
(2015).

A review on software maintenance issues and
how to reduce maintenance efforts.

International Journal of Computer Applications,
118(1).

References XII

[Keele, 2007] Keele, S. (2007).

Guidelines for performing systematic literature reviews in software engineering.

In Technical report, Ver. 2.3 EBSE Technical Report. EBSE.

[Kononenko et al., 2014] Kononenko, O., Baysal, O., Holmes, R., and Godfrey, M. W. (2014).

Dashboards: Enhancing developer situational awareness.

References XIII

In Companion Proceedings of the 36th International Conference on Software Engineering, ICSE Companion 2014, pages 552–555, New York, NY, USA. ACM.

[Lehman, 1980] Lehman, M. M. (1980).
On understanding laws, evolution, and
conservation in the large-program life cycle.
Journal of Systems and Software, 1:213–221.

References XIV

- [Lientz and Swanson, 1980] Lientz, B. P. and Swanson, E. B. (1980).
Software Maintenance Management.
Addison-Wesley Longman Publishing Co., Inc.,
Boston, MA, USA.
- [Mauldin, 1994] Mauldin, M. L. (1994).
Chatterbots, tinymuds, and the turing test:
Entering the loebner prize competition.
In *AAAI*, volume 94, pages 16–21.

References XV

[Petersen et al., 2008] Petersen, K., Feldt, R., Mujtaba, S., and Mattsson, M. (2008). Systematic mapping studies in software engineering.

EASE'08 Proceedings of the 12th international conference on Evaluation and Assessment in Software Engineering, pages 68–77.

References XVI

[Petersen et al., 2015] Petersen, K., Vakkalanka, S., and Kuzniarz, L. (2015).

Guidelines for conducting systematic mapping studies in software engineering: An update.

Information and Software Technology, 64:1–18.

[Polo et al., 1999] Polo, M., Piattini, M., Ruiz, F., and Calero, C. (1999).

Roles in the maintenance process.

ACM SIGSOFT Software Engineering Notes, 24(4):84–86.

References XVII

- [Rocha et al., 2015] Rocha, H., Oliveira, G., Marques-Neto, H., and Valente, M. T. (2015). Nextbug: a bugzilla extension for recommending similar bugs.
Journal of Software Engineering Research and Development, 3(1).

References XVIII

- [Singh and Chaturvedi, 2011] Singh, V. and Chaturvedi, K. K. (2011).
Bug tracking and reliability assessment system (btras).
International Journal of Software Engineering and Its Applications, 5(4):1–14.

References XIX

[Thung et al., 2014] Thung, F., Le, T.-D. B., Kochhar, P. S., and Lo, D. (2014).

Buglocalizer: Integrated tool support for bug localization.

In *Proceedings of the 22Nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*, FSE 2014, pages 767–770, New York, NY, USA. ACM.

References XX

[Tripathy and Naik, 2015] Tripathy, P. and Naik, K.
(2015).

Software Evolution and Maintenance.

Wiley.

[Wohlin et al., 2012] Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., and Wesslén, A.
(2012).

Experimentation in software engineering.

Springer Science & Business Media.

References XXI

[Zimmermann et al., 2009] Zimmermann, T., Premraj, R., Sillito, J., and Breu, S. (2009). Improving bug tracking systems. In *ICSE Companion*, pages 247–250. Citeseer.