Uma Revisão Sistemática sobre Fatores que Levam à Redução de Falhas em Software

Israel de Almeida Mello
israelalmeida91@hotmail.com
PUC Minas

Marcelo Werneck Barbosa
mwerneck@pucminas.br
PUC Minas

Maria Augusta Vieira Nelson
guta@pucminas.br
PUC Minas

Resumo: Produzir software de qualidade tem sido uma necessidade frente aos desafios do mercado de desenvolvimento de software. Neste sentido, reduzir falhas nos produtos de software entregues tem sido uma meta para as organizações desenvolvedoras. Este trabalho tem como objetivo realizar uma revisão sistemática com a finalidade de buscar e agregar evidências que respondam à pergunta: Quais os fatores levam à redução do número de falhas em um software? Como metodologia de pesquisa, foi seguido o processo formal de revisão sistemática, que coletou 56 artigos que tratam de alguma forma sobre aspectos possivelmente relacionados a redução de defeitos em software. Como resultado, foi estabelecido um ranking dos principais fatores responsáveis pela diminuição dos erros com base nas análises dos artigos identificados na revisão sistemática. O trabalho concluiu que melhorias no processo de desenvolvimento de software tem sido consideradas como o principal motivador de redução de falhas.

Palavras Chave: Falhas de Software - Revisão Sistemática - Defeitos - Qualidade Software -

1. INTRODUÇÃO

Entende-se por processo de desenvolvimento de software, um conjunto de atividades correlatas parcialmente ordenadas com o objetivo comum de construir um produto de software. É de desejo comum de todos que fazem parte do processo, que os artefatos produzidos em cada etapa estejam sempre dentro da conformidade esperada e também dentro dos limites de entrega estabelecidos pelo cronograma, para que o processo flua continuamente com qualidade e pontualidade. Este é o cenário ideal, mas nem sempre é o que acontece.

Ao se analisar cada atividade exercida dentro do processo de desenvolvimento de software, percebe-se que, independentemente da atividade, sempre há a possibilidade de defeitos serem inseridos, seja por conta do processo, em que o defeito é passado despercebido de um artefato a outro, ou por conta da complexidade da atividade, como por exemplo, uma falha na estruturação lógica de um algoritmo complexo (PÁDUA FILHO, 2009).

É essencial saber diferenciar os tipos diferentes de falhas, que podem ser resultantes de requisitos funcionais ou requisitos não funcionais do software, como também a classificação de cada falha encontrada no sistema para se ter um controle e classificação de cada falha. Contudo, mais importante do que se ter uma classificação das falhas mais frequentes, é saber quais os principais fatores que levam à redução do número de falhas em um software.

Pensando neste questionamento, este trabalho teve como objetivo levantar informações na literatura por meio de um processo de revisão sistemática para identificar os principais fatores que levam à redução do número de falhas em software. O método de revisão sistemática consiste na revisão da literatura seguindo critérios de seleção pré-estabelecidos sobre determinado assunto ou tema e consiste em: 1) definir uma pergunta; 2) buscar fontes primárias de informação (artigos, livros, etc.) relacionadas com a pergunta a ser respondida; 3) definir critérios de inclusão e exclusão das fontes primárias encontradas; 4) analisar a qualidade das fontes primárias com base nos critérios de inclusão estabelecidos e 5) apresentar os resultados do estudo (KITCHENHAN et al., 2008).

Este artigo está organizado da seguinte forma. A Seção 2 descreve os conceitos do método de revisão sistemática. A Seção 3 apresenta os trabalhos relacionados. A Seção 4 descreve o processo de revisão sistemática realizado neste trabalho para atender os objetivos da pesquisa incluindo todos os trabalhos identificados no processo. A Seção 5 apresenta os resultados do trabalho identificando e classificando os fatores que levam à redução de falhas em software. A Seção 6 conclui o trabalho apresentando as possibilidades de investigação futura.

2. REVISÃO SISTEMÁTICA

Uma revisão sistemática tem o objetivo principal de obter evidências científicas sobre o assunto ou tema a ser desenvolvido. Pode ser conceituada como um estudo de revisão que utiliza como fonte de dados a literatura disponível sobre determinado tema (SAMPAIO; MANCINI, 2007).

Uma das razões para se utilizar o método de revisão sistemática da literatura é buscar fundamentação teórica obtendo, com isso, agregação de informações para a resolução de um problema proposto ou a busca de uma resposta. Através desta sintetização de evidências, pretende-se obter um resultado de maior valor quantitativo, devido ao grande número de fontes primárias de informação selecionadas.

Assim como todo método de pesquisa, uma revisão sistemática segue uma série de etapas pré-definidas: Etapa de Planejamento, Etapa de Desenvolvimento e Etapa de Resultados.

Na etapa de Planejamento, primeiramente, deve-se definir qual será o foco ou qual será a pergunta a ser respondida, ou seja, qual é o "por que" da elaboração da revisão sistemática. A partir disso, tem-se a base para a obtenção do volume inicial de fontes de informação primárias. É importante ressaltar que a qualidade da fonte primária é um fator de diferenciação para o desenvolvimento de uma revisão sistemática de qualidade. Por esse motivo, é recomendada a utilização de bases de dados confiáveis que tenham materiais revisados por terceiros (SciElo, IEEE, CAPES, etc).

Após as definições citadas anteriormente, é necessário elaborar os critérios de inclusão e exclusão das fontes primárias de informação. São critérios de inclusão e exclusão todas as condições que uma fonte de informação (Artigo, Livros e Periódicos) obrigatoriamente deve possuir. Por exemplo, se um critério define que um artigo deve estar disponível em uma base de dados confiável, então todos os artigos que obedecem a esta condição serão incluídos. Os artigos que, de alguma maneira, não obedecem a este critério de seleção deverão ser então excluídos (SAMPAIO; MANCINI, 2007).

A etapa de Desenvolvimento envolve a seleção e validação das fontes primárias de informação através dos critérios de inclusão e exclusão levantados na Etapa de Planejamento, ou seja, devem-se aplicar os critérios nas fontes de informação pesquisadas, a fim de filtrar o que está de acordo com os critérios pré-estabelecidos. Após a seleção das fontes de informação, é hora de analisar cada exemplar incluído na coleção. Esta análise serve como parâmetro para a extração dos dados que serão exibidos na Etapa de Resultados.

A etapa de Resultados é a final do processo de elaboração de uma revisão sistemática. Consiste em mostrar os dados em um formato que possa ser analisado e estudado. Esta etapa fornece subsídios para que a pergunta definida na Etapa de Planejamento seja respondida. Geralmente, os resultados são exibidos em forma de tabelas ou gráficos, tendo como base as fontes de informação primária selecionadas.

3. TRABALHOS RELACIONADOS

Durante o processo da pesquisa, não foi encontrado um trabalho acadêmico que, por meio da utilização do processo da revisão sistemática, procurou realizar um levantamento de fatores responsáveis pela diminuição de erros em software. Entretanto, foram recuperados trabalhos acadêmicos que adotaram o processo de revisão sistemática com a finalidade de reunir evidências que respondam a algum questionamento proposto.

De modo a relacionar este trabalho com outros artigos acadêmicos, foram considerados os seguintes critérios: o artigo deverá ter sido conduzido utilizando a metodologia de revisão sistemática; e o assunto deverá estar relacionado à mesma área de pesquisa deste artigo.

(Carrillo, A.B., 2012) utilizou a metodologia de revisão sistemática para métricas para avaliar a qualidade funcional de um software. O alvo das pesquisas realizadas foram as métricas existentes que podem ser usadas para medir a qualidade funcional (conforme definido na norma internacional ISO 25010) de um produto de software. Como resultado da revisão realizada, 23 métricas foram encontradas, a maioria delas com base em testes. As conclusões obtidas a partir deste estudo mostraram que, apesar de que mais métricas deveriam ser definidas, existe um número suficiente de métricas para medir todas as características funcionais de qualidade definidas na norma internacional ISO 25010.

Por meio de um processo de revisão sistemática, (Saha, S.K, 2012) realizou uma revisão sistemática recuperando artigos relacionados às técnicas criativas de elicitação de requisitos, engenharia de requisitos e avaliação sobre técnicas de criatividade para a engenharia de requisitos. O autor utilizou bases de dados científicas como IEEE Xplore, ACM, Compendex , Inspec , SpringerLink e Science Direct. Após as definições iniciais, foi executada uma busca

com termos de pesquisa relevantes. A partir da revisão sistemática foram determinados os papéis de criatividade e identificado o impacto de técnicas de criatividade na área de engenharia de requisitos.

Em (Julio Reis, 2012), a metodologia de revisão sistemática foi utilizada para identificar os fatores de projeto e de sistemas que influenciam o tempo gasto nas atividades de levantamento de requisitos. O autor se baseou nos conceitos apresentados por (Kitchenham, 2008) para o desenvolvimento da revisão sistemática e focou nas seguintes bases de dados científicas para a captação dos artigos: IEEEXplore, ACM Digital Library, entre outras.

A metodologia de revisão sistemática tem sido amplamente utilizada para solução de questionamentos buscando a comprovação com base na literatura disponível (artigos, revistas científicas, etc.). Quando o processo é bem definido e as etapas são bem executadas, observando-se também a qualidade das fontes primárias de informação, o resultado da aplicação desta metodologia é a resposta ou resolução do questionamento inicialmente proposto, nos permitindo extrair conclusões comprovadas através da análise da literatura.

4. O PROCESSO DE REVISÃO SISTEMÁTICA REALIZADO

O processo da revisão sistemática foi realizado com base nos conceitos apresentados por (Mancini, Sampaio, 2007) e (Kitchenhan, 2008), com a finalidade de descobrir quais são os principais fatores que contribuem para a diminuição dos erros que ocorrem nos softwares.

Na primeira etapa da revisão sistemática, foi realizada uma busca informal vislumbrando obter um conjunto inicial de artigos para a obtenção das palavras-chave (*Keywords*) e, por conseguinte, a realização da pesquisa real após a seleção das palavras-chave mais relacionadas ao assunto da pesquisa. A busca informal foi conduzida considerando as seguintes bases de dados científicas: IEEEXplore, SciElo e Portal CAPES.

A lista inicial de palavras-chave (*Keywords*) foi obtida selecionando as palavras-chave presentes nos artigos retornados na primeira busca, ou seja, na busca informal. Ao formular esta lista, foram excluídas as *Keywords* que estavam em duplicidade.

A lista de palavras-chave inicialmente recuperadas pode ser visualizada na Tabela 1.

Tabela 1. Lista Inicial de Palaras-Chave

Keywords
1 – Engenharia de Requisitos; 2 – Software Quality; 3 – Software Errors; 4 - Regression testing; 5 - field failure data; 6 – software improvement; 7 – software modeling; 8 – Software Commom Faults; 9 – Reducing Software Faults; 10 - software dependability; 11 - software failures; 12 - software faults; 13 - Risk analysis; 14 - Safety-critical Software; 15 - Software reliability prediction; 16 - Software Testing Model; 17 - Software Testing; 18 - Testing Type; - 19 - software development; 20 - software reliability; 21 – Design Patterns; - 22 - software quality assurance; 23 - Software engineering; 24 – Software Systematic Review; 25 - Best practices; 26 - Business processes; 27 - Gerência de requisitos; 28 - formal technical review (FTR); 29 – Prevention Errors; 30 – Processo de Desenvolvimento de Software; 31 – Software Common Errors

Como critérios de seleção das palavras-chave obtidas após a busca informal, foram desconsideradas as palavras-chave dos artigos que não se relacionavam diretamente com o tema da pesquisa ou trouxeram, como maioria, resultados não relacionados à pergunta proposta (durante a primeira busca). As palavras-chave selecionadas foram as que retornaram resultados mais relevantes. A lista de palavras-chave final após validação junto aos critérios acima relacionados pode ser vista na Tabela 2.

Tabela 2. Lista Final de Palavras-Chave (Keywords)

Keywords
1 – Software Commom Faults; 2 – Software Failures; 3 – Design Patterns; 4 – Software Commom Errors; 5 – Formal Technical Review (FTR); 6 – Best Practices; 7 – Software Engineering; 8 – Software Flaws; 9 - software reliability; 10 – Software Quality; 11 – Software Testing; 12 - Reducing Software Faults; 13 – Prevention Errors; 14 – Software Development Process; 15 – Business Process; 16 - software quality assurance

Considerando as palavras-chave relacionadas na Tabela 2, fez-se uma nova busca nas bases de dados (definidas na fase de planejamento) com o intuito de refinar ainda mais a busca e obter um novo conjunto de artigos contendo informações mais relevantes. A partir daí, procurou-se validar os resultados das pesquisas inicial e final aplicando os critérios de inclusão e os critérios de exclusão definidos a seguir.

A validação dos artigos foi conduzida selecionando aqueles que tinham relação com o assunto da pergunta proposta; os artigos que estivessem escritos em português ou inglês; artigos que estivessem disponíveis na Web; artigos que possuíam as palavras-chave em seu título ou resumo; e artigos que apresentassem pelo menos um fator ou referência a um aspecto responsável pela diminuição ou erradicação de algum erro proveniente de software ou do processo de desenvolvimento de software. (*Critérios de Inclusão*)

Com a finalidade de se obter um número maior de artigos para análise, alguns artigos da busca informal foram mantidos para o processo da análise dos resultados da busca. Não foram considerados os artigos que estivessem escritos em outras línguas a não ser português e inglês; artigos que não fizessem parte da área de pesquisa (tecnologia e sistemas de informação); artigos que não estivessem disponíveis na Web em uma das três bases de dados escolhidas (SciElo, CAPES ou IEEE) (*Critérios de Exclusão*).

Após a análise do conjunto final de artigos, buscou-se formular um ranking com os principais fatores de influência na diminuição de erros em software. O critério de relevância utilizado neste ranking foi a quantidade de artigos que utilizaram um determinado fator de melhora. Em outras palavras, quanto mais bem colocado no ranking, mais artigos utilizaram este fator na diminuição de erros de software.

A Tabela 3 representa uma síntese dos dados extraídos dos artigos. Ela é constituída pelos seguintes itens:

- *N°:* Número sequencial identificador do artigo selecionado;
- *Título:* Título do artigo selecionado;
- *Potencial Fator Encontrado:* Provável fator responsável pela diminuição de erros em software identificado no artigo após análise.
- *Síntese e Observações:* Síntese e observações pertinentes ao artigo selecionado.

É importante destacar como contribuição deste trabalho o fato de a Tabela 3 contar com a identificação de 56 trabalhos científicos relacionados ao tema central deste trabalho. A compilação destas referências é uma grande contribuição deste trabalho, além de claro relacionar os fatores que poderiam levar à redução de defeitos em software.

Tabela 3. Síntese da Revisão Sistemática

Nº	Título do Artigo	Potencial Fator Encontrado	Síntese e
1	Reducing Field Failures in System Configurable Software: Cost-Based Prioritization (SRIKANTH, COHEN e XIAO, 2009)	Métodos de Detecção de Falhas (gestão de configuração)	Após a análise em um estudo de caso, os priorização e concluiram que: (1) utiliz podem melhorar a taxa de detecção de fa também a taxa de detecção de falha configurações testadas.
2	Software Dependability in the Tandem GUARDIAN System (INHWAN e IYER, 1995)	Análise de Impacto de Falhas (para determinar pontos de verificação)	O artigo classifica as causas de falhas d tolerar falhas de software. Propõe um m software sobre a confiabilidade de um sis significado dos principais fatores que identificar áreas de melhoria.
3	Reliability Modeling for Safety-Critical Software (SCHNEIDEWIND, 1997)	Integração de informações para o setor de teste	Utilização de uma nova abordagem para análise de risco, a previsão confiabilid abordagem se aplica a outros softwares de
4	Research and establishment of quality cost oriented software testing model (YUYU e SHEN, 2006)	Teste de Software (com gestão de controle de custos)	O ponto-chave deste trabalho é construir de controle de custos e analisar *trade-off* e
5	Predicting fault-prone software modules in embedded systems with classification trees (KHOSHGOFTAAR e ALLEN, 1999)	Previsão/Detecção de Falhas através de um algoritmo	O artigo parte do princípio argumentando ser ferramentas valiosas para a engenhari algumas técnicas de aprimoramento de so práticas para serem aplicadas a todos os algoritmo CART (*Classification And Reg* software têm alto risco de falhas a serem d
6	Software crisis, what software crisis? (YU, 2009)	Melhoria no processo de desenvolvimento	O artigo expõe que a indústria de software complexo. O artigo defende a idéia de encomenda" do setor de "software pré-em
7	Modeling the Reliability of Existing Software using Static Analysis (SCHILLING, 2006)	Aplicação de modelos de confiabilidade de software	Modelos de confiabilidade de software tra a serem coletados durante o desenvol confiabilidade do software. Estes dados para serem aplicados. O autor propõe apropriado combinando a análise estática limitados com captura de caminho e redes
8	From Off-Line to Continuous On-line Maintenance (PEZZE, 2012)	Mecanismo de auto - Correção (a nível de	O autor expõe a dificuldade e alto custo antes da implantação de siste

		código)	Uma abordagem interessante baseia-se n são redundantes, por natureza, e explora a para corrigir falhas, minimizando, assim, para alimentar o mecanismo de auto-corre
9	Experiments in software reengineering (LEACH, 1997)	Reestruturação Técnica (a nível de código)	O trabalho descreve sobre a reestruturaçã software e de atividades de manutenção. I de caminhos de execução do programa.
10	Stress Testing Software to determine Fault Tolerance for Hardware Failure and Anomalies (WU, 2012)	Teste de Software (testes de Hardware e Software)	O trabalho descreve sobre a dificuldade software e a tolerância a falhas. A partir existem para teste de software destac concentrar em um teste de software ef desempenho e tolerância a falhas de hardv
11	Fault Injection Techniques and Tools (MEI-CHEN, TSAI, IYER e 1997)	Detecção de Erros através de métodos de Injeção de Falhas	Injeção de falhas é importante para avalia Pesquisadores e engenheiros criaram nov implementados em hardware e software.
12	The influence of syntactic and semantic errors on the quality of software (DOGSA e ROZMAN, 1991)	Análise de Fatores de Erros (causa de erros)	Os autores descrevem uma experiência infrutíferas correlaciona-se com o número
13	An Instrumented Approach to Improving Software Quality through Formal Technical Review (JOHNSON, 1994)	Revisão Técnica Formal (a nível de código)	O autor apresenta um instrument chama formal do código fonte. CSRS é um ar computador para revisões técnicas formais
14	The Simulation of Formal Management and Technical Reviews to Increase Quality in Undergraduate Software Engineering Projects (TERRY and HOPE, 1998)	Revisão Técnica Formal (a nível de código)	A introdução de revisões técnicas em p Cowan University (ECU) resultou em me processo utilizado, bem como proporcion
15	Best practices of RUP ® in software product line development (YASAR et al, 2008)	Implementação das Melhores Práticas do RUP (Melhoria no Processo de Desenvolvimento)	O trabalho apresenta a aplicação de desenvolvimento de um e-commerce.
16	Making processes from best practice frameworks actionable (GRAUPNER at al, 2009)	Framework de Melhores Práticas (melhoria no processo de desenvolvimento)	O trabalho apresenta um framework de testes de software.
17	When standards and best practices are ignored (JACKELEN e JACKELEN,	Utilização de Padrões e Melhores Práticas	Este trabalho apresenta uma situação real, software e as melhores práticas foram

	1999)	(Melhoria do processo de desenvolvimento)	quando a sequência do processo de dese riscos e custos aumentaram consideravelm
18	Best practices for software security: An overview (YASAR et al, 2008)	Melhores Práticas (de desenvolvimento)	O trabalho aborda a utilização de algumas de software seguro e categorização de t desenvolvimento de software. Os resultad melhores práticas em desenvolvimento de
19	Using Behavioral Profiles to Detect Software Flaws in Network Servers (ANTUNES e NEVES, 2011)	Detecção de Erros (Analisando comportamento de Hardware)	O artigo mostra uma abordagem de c comportamental de um servidor de rede são detectadas automaticamente, se o co durante o processamento dos casos de test
20	Whose bug is it anyway? The battle over handling software flaws (APPLEWHITE, 2004)	Melhoria da Segurança do Software (diminuição de vulnerabilidades)	O texto explora a dificuldade de se criar u foca no cuidado com a crescente utilizaç usuários.
21	Instilling a Defect Prevention Philosophy (WILLIAMS, 1998)	Utilização do processo de desenvolvimento de software "Cleanroon"	Cleanroom, um processo de desenvolvim acrescenta rigor de engenharia para o centra-se na prevenção de defeitos e contr
22	Evaluating the Use of Requirement Error Abstraction and Classification Method for Preventing Errors during Artifact Creation: A Feasibility Study (WALIA e CARVER, 2010)	Taxonomia de Erros de Requisitos	Este artigo investiga a utilização da taxon de prevenção de defeitos no documento familiarizar os analistas com os erros para
23	Reliability engineering as applied to software (YATES e SHALLER, 1990)	Engenharia de Confiabilidade (no desenvolvinmento de software)	O texto fala sobre a Engenharia de Co confiável, o objetivo deve ser a prevençã desenho), e não pela descoberta de erros a
24	Increasing Quality in Scenario Modelling with Model-Driven Development (SANTOS et al, 2010)	Padrão (regras) para automatizar a migração da atividade para os modelos de seqüência.	Este documento define regras de tran atividade (cenários de casos de uso) pa sequência - Interações entre objetos). O diminuição de erros durante a migração.
25	Automated Software Specification and Design Using the SOFL (SHAOYING et al, 2009) Formal Engineering Method	Ferramenta de apoio à construção de especificações de projeto.	Este artigo descreve as técnicas para a co para apoiar a construção de especificaç incluem: (1) detecção e prevenção de er orientação "inteligente" e sistemática par especificações.
26	An Analysis of Errors and Their Causes In System Programs (ENDERS, 1975)	Classificação de Erros (Análise dos Erros)	Usando uma classificação dos erros de conclusões sobre as possíveis causas dess em uma discussão sobre os métodos mais

27	Common Trends in Software Fault and Failure Data (HAMILL e GOSEVA-POPSTOJANOVA, 2009)	Análise das Falhas (Controle das Falhas)	Neste artigo, analisaram-se os dados de f do mundo real. Especificamente, avalia falhas de software individuais e 2) a d software. Os resultados mostraram qu causadas por múltiplas falhas espalhadas
28	Architectural Design Patterns for Flight Software (FANT et al, 2011)	Aplicação de Padrões de Projeto (Design Patterns)	Este trabalho apresenta os padrões de pro de vôo espacial. Este artigo descreve com usados para construir as características co
29	Proposta de integração da Engenharia de Software nas estratégias empresariais (ADALBERTO e COSTA, 2005)	Uso da Engenharia de Software (estratégia da organização)	A engenharia de software estabelece organização e a geração de valor no plane
30	Definição de requisitos de software baseada numa arquitetura de modelagem de negócios (AZEVEDO e CAMPOS, 2008)	Melhoria no Levantamento de Requisitos (abordando o negócio)	Neste artigo é apresentada uma metodolo da incorporação de atividades propostas p requisitos, baseadas em uma arquitetura d
31	Contribuição dos modelos de qualidade e maturidade na melhoria dos processos de software (TONINI et al, 2008)	Implementação de Modelos de Maturidade (Melhoria do Processo de desenvolvimento)	O artigo discute a implementação de me principais modelos de qualidade e de matu
32	Práticas do CMMI® como regras de negócio (MORGADO et al, 2007)	Melhoria no Processo de desenvolvimento de Software (Formalização das práticas do CMMI)	O artigo discute vantagens como o certificação/auditoria do CMMI e a melho apóia o processo de desenvolvimento de s
33	Recommendation system for design patterns in software development: An DPR overview (PALMA et al, 2012)	Padrões de Projeto (Design Patterns)	A partir da utilização de Padrões de Rec projeto reutilizável.
34	What Do We Know about the Effectiveness of Software Design Patterns (CHENG e BUDGEN, 2012)	Padrões de Projeto (Design Patterns)	Foi realizada uma revisão sistemática bus sobre o uso dos 23 padrões catalogados.
35	Improving a web application using design patterns: A case study (PHEK et al, 2010)	Padrões de Projeto (Design Patterns)	Este trabalho analisa e compara uma séri de projeto de arquitetura e navegação para
36	An Approach for Evaluating the Effectiveness of Design Patterns in Software Evolution (NIEN-LIN et al, 2011)	Padrões de Projeto (Design Patterns)	Neste artigo é proposta uma abordagem evolução de sistemas.
37	A study of process improvement best	Melhores Práticas	O objetivo do artigo foi analisar as aborda

	practices (RAHMAN et al, 2011)		processos de software. Outro objetivo fo adoção de melhorias de processos e pa questões significativas de melhoria de pro genéricas para melhorias de processos.
38	The effectiveness of real-time embedded software testing (BO e XIANGHENG, 2011)		O artigo aborda testes em sistemas de te eficácia dos testes de software permite qu
39	Fast software implementation of error detection codes (FELDMEIER, 1995)	Códigos de detecção de erros	Dado o rendimento de vários códigos de um protocolo de código com a velocidad erros.
40	Reliability Implications of Register Utilization: An Empirical Study (ROMER and TROGER, 2011)	Prevenção de erro baseada em software para falhas de arquivo de registro	Neste artigo, analisa-se como a escolha ce estratégia de prevenção de erro baseado e
41	Early introduction of software metrics (HOFFMAN, 1999)	Utilização de métricas	A metodologia geral que pode auxiliar o de um programa eficaz utilizando métrica a compreensão e melhoria do processo d inicial do ciclo de vida do software.
42	Toward trustworthy software systems (HASSELBRING e REUSSNER, 2006)	Aplicação de Design Patterns	Foram desenvolvidos novos métodos pa de confiança.
43	Metrics to evaluate functional quality (CARRILLO, 2012)	Melhoria no processo de desenvolvimento (Aplicação da ISO)	As conclusões obtidas a partir deste estud sido definidas, existe um número su características funcionais de qualidade de
44	A Systematic Review on Creativity Techniques for Requirements Engineering (SAHA et al, 2012)	Melhoria na Captação/Análise dos Requisitos	A partir de uma revisão sistemática for identificado o impacto de técnicas de criat
45	The impacts of software process improvement on developers: A systematic review (LAVALLÉE e ROBILLARD, 2012)	Melhoria no Processo de Desenvolvimento de Software	Impactos positivos incluem uma reduçã comunicações entre e equipe, bem como n
46	User-Centered Design and Agile Methods: A Systematic Review (SILVA et al, 2011)	Desenvolvimento ágil (Design centrado no usuário)	Este artigo apresenta os resultados de u sobre a integração do desenvolvimento centrado no usuário
47	Systematic Review of Software Product Certification (RODRIGUEZ e PIATTINI, 2012)	Certificação do produto de software	Esta revisão sistemática visa identificar in de produtos de software, os recursos qu aplicação.
48	Effectiveness of Requirements	Técnicas de Elicitação de	Este artigo apresenta uma revisão sistemá

	Elicitation Techniques: Empirical Results Derived from a Systematic Review (DAVIS et al, 2006)	Requisitos	técnicas de elicitação
49	Dispersion, coordination and performance in global software teams: A systematic review (NGUYEN et al, 2012)	Gerência da equipe / Coordenação da equipe	Coordenação de equipe eficaz é crucial p O objetivo deste artigo é resumir as evidê coordenação de equipe e atuação em proje
50	Managing Quality Requirements: A Systematic Review (SVENSSON et al, 2010)	Requisitos de Qualidade	A revisão investiga o que se sabe sobr requisitos de qualidade.
51	The role of software process simulation modeling in software risk management: A systematic review (DAPENG et al, 2009)	Gerenciamento de Riscos	Este trabalho apresenta uma revisão sist estado da arte das aplicações da MSF na g
52	Metrics and indicators to assist in the distribution of process steps for distributed software development: A systematic review (MATUSSE et al, 2012)	Métricas de software	Neste contexto, o presente trabalho tem c para reunir provas sobre a existência de m desenvolvimento de software distribuído.
53	A Systematic Review on Architecting for Software Evolvability (BREIVOLD e CRNKOVIC, 2010)	Padrões de Projeto / Arquitetura do projeto	Neste trabalho, realizou-se uma revisão estudos existentes na promoção da evoluç
54	A Systematic Review of Contemporary Metrics for Software Maintainability (ABILIO et al, 2012)	Métricas de software	Este artigo apresenta os resultados da apl para identificar métricas contemporânea propôs para as tecnologias a característica
55	Complexity measures for software systems : towards multi-agent based software testing (DHAVACHELVAN e UMA, 2005)	Teste de Software	Este trabalho apresenta um estudo analisa de software, o que motiva os esforços rec teste de software orientado a agentes (AO
56	Software reliability modeling and cost estimation incorporating testing-effort and efficiency (CHIN-YU et at, 1999)	Teste de Software	Este trabalho apresenta duas questões imp do software e economia de confiabilidade

5. RESULTADOS

Todos os artigos foram lidos e para cada um foi elaborada uma síntese. Após a análise da síntese dos artigos, foi estabelecido um critério de agrupamento de todos os fatores encontrados nos trabalhos selecionados, a fim de se obter um *ranking* mais coeso e enxuto.

Os fatores foram agrupados considerando-se os seguintes tópicos:

- *Técnicas de Melhoria na Especificação (Ferramenta de apoio à construção de especificações de projeto, Melhoria na Captação/Análise dos Requisitos);*
- *Gerenciamento de Erros (Classificação de Erros, Método de Injeção de Erros);*
- *Métodos de Detecção de Falhas;*
- *Melhoria no Processo de Desenvolvimento de Software (RUP, Modelos de Maturidade, etc);*
- *Melhoria no Desenvolvimento de Software (Aplicação de Design Pattterns, Melhores Práticas, etc);*
- *Gerenciamento de Riscos;*
- *Melhoria na área de Testes de Software (Métodos que auxiliem para um melhor teste, etc);*

Antes do desenvolvimento do Ranking de Fatores, a Tabela 4 foi elaborada com a finalidade de relacionar os artigos aos fatores agrupados, considerando-se os tópicos relacionados anteriormente.

Tabela 4. Fator após agrupamento vs Artigo selecionado

Fator	Índices dos artigos
Técnicas de Melhoria na Especificação	1, 2, 22, 23, 24, 25, 29, 30, 44, 48, 50,
Gerenciamento de Erros	19, 26, 39, 40
Métodos de Detecção de Falhas	5, 8, 11, 12, 21, 27,
Melhoria no Processo de Desenvolvimento de Software	6, 15, 16, 17, 31, 32, 37, 41, 43, 45, 47, 49, 52, 54
Melhoria no Desenvolvimento de Software	9, 13, 14, 18, 28, 33, 34, 35, 36, 42, 46, 53
Gerenciamento de riscos	51
Melhoria nos Testes de Software	3, 4, 7, 10, 20, 38, 55, 56

A Tabela 5 demonstra o ranking contendo a relação entre os fatores e os respectivos artigos. O ranking foi elaborado considerando-se as soluções extraídas a partir da análise dos artigos obtidos, utilizando a metodologia de revisão sistemática. Ela é composta pelos seguintes itens:

- *Fator:* Fator identificado, após análise do artigo, que contribuiu para a diminuição da ocorrência de erros em software.
- *N° de Artigos com o Fator:* Quantidade de artigos que utilizaram um respectivo fator como medida de diminuição de erros em software.

Tabela 5: Ranking dos fatores responsáveis pela diminuição de erros em software

Posição	Fator	N° de Artigos com o fator
1º	Melhoria no Processo de Desenvolvimento de Software	14
2°	Melhoria no Desenvolvimento de Software	12
3°	Técnicas de Melhoria na Especificação	11

4°	Melhoria na área de Testes de Software	8
5°	Métodos de Detecção de Falhas (Gerenciamento de Falhas)	6
6º	Gerenciamento de Erros	4
7º	Gerenciamento de Riscos	1
		Total: 56

6. CONCLUSÃO E TRABALHOS FUTUROS

O objetivo deste trabalho foi a realização de uma revisão sistemática para identificar os principais fatores responsáveis pela diminuição de erros em softwares. Assim com base na Lei de Pareto, que diz que 80% dos problemas possuem 20% das causas, este trabalho pode orientar organizações desenvolvedoras de software a decidir em quais ações deve-se investir para promover a melhoria da qualidade de software em geral.

A partir da análise do ranking, foi possível concluir que melhorias e investimentos no processo de desenvolvimento de software foram citados como o principal fator responsável pela diminuição de erros em softwares. Este fator está relacionado principalmente ao nível de maturidade das organizações. As empresas que possuem níveis de maturidade mais elevados possuem processos mais definidos e quantificáveis. Percebeu-se também que melhorias no desenvolvimento de software geraram impactos muito positivos na qualidade do produto de software. A análise da identificação dos fatores mais relevantes também permitiu concluir que as atividades mais relacionadas a gerenciamento e análise não parecem contribuir diretamente na qualidade final do produto de software.

Como trabalhos futuros, pretende-se utilizar o resultado deste trabalho em organizações reais com o intuito de auxiliá-las na tomada de decisões estratégicas como, por exemplo, determinar qual atividade ou área teria prioridade em receber investimentos e recursos.

REFERÊNCIAS BIBLIOGRÁFICAS

ABILIO, R.; TELES, P.; COSTA, H. ; FIGUEIREDO, E. A Systematic Review of Contemporary Metrics for Software Maintainability - Sixth Brazilian Symposium on Software Components Architectures and Reuse (SBCARS), 2012.

ADALBERTO F. R; COSTA I.; Proposta de integração da Engenharia de Software nas estratégias empresariais. Revista Produção, v. 15, n. 3, p. 448-455, Dez. 2005.

AHMED, F.; CAPRETZ, L.F. Best practices of RUP ® in software product line development - ICCCE 2008. International Conference on Computer and Communication Engineering, 2008.

ANTUNES, J.; NEVES, N.F. Using Behavioral Profiles to Detect Software Flaws in Network Servers - IEEE 22nd International Symposium on Software Reliability Engineering (ISSRE), 2011.

APPLEWHITE, A. Whose bug is it anyway? The battle over handling software flaws - IEEE Software v.21, 2004.

AZEVEDO D.P.; CAMPOS R. Definição de requisitos de software baseada numa arquitetura de modelagem de negócios. Produção, v. 18, n. 1, p. 026-046, Abr. 2008

BEACH, R.; MUHLEMANN, A. P.; PRICE, D. H. R.; PATERSON, A. & SHARP, J. A. A review of manufacturing Flexibility. European Journal of Operational Research, v. 122, 2000, pp. 41-57.

BREIVOLD, H.P.; CRNKOVIC, I. A Systematic Review on Architecting for Software Evolvability - 21st Australian Software Engineering Conference (ASWEC), 2010.

BO Z.; XIANGHENG S. The effectiveness of real-time embedded software testing - 9th International Conference on Reliability, Maintainability and Safety (ICRMS), 2011.

CARRILLO, A.B.; MATEO, P.R. ; MONJE, M.R. Metrics to evalute fuctional quality: A sistematic review - 7th Iberian Conference on Information Systems and Technologies (CISTI), 2012

CHENG Z.; BUDGEN, D. What Do We Know about the Effectiveness of Software Design Patterns - IEEE Transactions on Software Engineering, 2012.

CHIN-YU H.; JUNG-HUA L.; SY-YEN K. ; LYU, M.R. Software reliability modeling and cost estimation incorporating testing-effort and efficiency - 10th International Symposium on Software Reliability Engineering, 1999.

DAPENG L.; QING W.; JUNCHAO X. The role of software process simulation modeling in software risk management: A systematic review - ESEM 2009. 3rd International Symposium on Empirical Software Engineering and Measurement, 2009.

DAVIS, A.; DIESTE, O. ; HICKEY, A. ; JURISTO, N. ; MORENO, A.M. Effectiveness of Requirements Elicitation Techniques: Empirical Results Derived from a Systematic Review - 14th IEEE International Conference Requirements Engineering, 2006.

DHAVACHELVAN, P.; UMA, G.V. Complexity measures for software systems : towards multi-agent based software testing - Proceedings of 2005 International Conference on Intelligent Sensing and Information Processing, 2005.

DOGSA, T.; ROZMAN, I. The influence of syntactic and semantic errors on the quality of software - International Symposium on Software Reliability Engineering, 1991.

ENDRES, A. An Analysis of Errors and Their Causes In System Programs - IEEE Transactions on Software Engineering, 1975.

FANT, J.S.; GOMAA, H. ; PETTIT, R.G. Architectural Design Patterns for Flight Software - 14th IEEE International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops (ISORCW), 2011.

FELDMEIER, D.C. Fast software implementation of error detection codes - IEEE/ACM Transactions on Networking, 1995.

FILHO, Wilson de Pádua. Engenharia de Software: Fundamentos, Métodos e Padrões 3ª ed. Rio de Janeiro, LTC Editora, 2009, Cap. 3, p. 60-63.

GRAUPNER, S.; MOTAHARI-NEZHAD, H.R. ; Singhal, S. ; Basu, S. Making processes from best practice frameworks actionable - EDOCW 2009. 13th Enterprise Distributed Object Computing Conference Workshops, 2009.

HAMILL, M.; GOSEVA-POPSTOJANOVA, K. Common Trends in Software Fault and Failure Data - IEEE Transactions on Software Engineering, 2009.

HASSELBRING, W.; REUSSNER, R. Toward trustworthy software systems – Computer v.39, 2006.

HOFFMAN, G.D. Early introduction of software metrics - Proceedings of the IEEE 1989 National Aerospace and Electronics Conference, 1989. NAECON 1989.

JACKELEN, G.; JACKELEN, M. When standards and best practices are ignored - Fourth IEEE International Symposium and Forum on Software Engineering Standards, 1999.

JOHNSON, P.M. An Instrumented Approach to Improving Software Quality through Formal Technical Review - 16th International Conference on Software Engineering, 1994.

KHOSHGOFTAAR, T.M.; ALLEN, E.B. Predicting fault-prone software modules in embedded systems with classification trees - 4th IEEE International Symposium on High-Assurance Systems Engineering, 1999.

KITCHENHAN, Barbara et al. Systematic literature reviews in software engineering: A systematic literature review, Information and Software Technology. Technical report. Nov. 2008.

INHWAN Lee ; Iyer, R.K. Software Dependability in the Tandem Guardian System - IEEE Transactions on Software Engineering, mai. 1995.

LAVALLÉE, M.; ROBILLARD, P.N. The impacts of software process improvement on developers: A systematic review - 34th International Conference on Software Engineering (ICSE), 2012.

LEACH, R.J. Experiments in software reengineering - Proceedings of the IEEE 1997 National Aerospace and Electronics Conference, 1997.

MATUSSE, E.A.; HUZITA, E.H.M. ; TAIT, T.F.C. Metrics and indicators to assist in the distribution of process steps for distributed software development: A systematic review - XXXVIII Conferencia Latinoamericana En Informatica (CLEI), 2012.

MEI-CHEN H.; TSAI, T.K. ; IYER, R.K. Fault Injection Techniques and Tools - Computer v.30, 1997.

MORGADO, G.P.; GESSER I.; SILVEIRA D.S.; MANSO F.S; LIMA P.M.; SCHMITZ E. A. Práticas do CMMI® como regras de negócio. Produção, v. 17, n. 2, p. 383-394, Ago. 2007.

NGUYEN D.A.; CRUZES, D.S. ; CONRADI, R. DISPERSION. Coordination and performance in global software teams: A systematic review - ACM-IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM), 2012.

NIEN-LIN H.; LIN-CHIEH W.; DER-HONG T. ; CHU, W. ; CHIH-HUNG C.; CHORNG-SHIUH K. An Approach for Evaluating the Effectiveness of Design Patterns in Software Evolution - IEEE 35th Annual Computer Software and Applications Conference Workshops (COMPSACW), 2011.

OLIVEIRA, U. R. Gerenciamento de riscos operacionais na indústria por meio da seleção de diferentes tipos de flexibilidade de manufatura. 2009. 246 f. Tese (Doutorado em Engenharia Mecânica) – Faculdade de Engenharia do Campus de Guaratinguetá, Universidade Estadual Paulista, Guaratinguetá, 2009.

PADOVEZE, C. L. & BERTOLUCCI, R. G. Proposta de um Modelo para o Gerenciamento do Risco Corporativo. In: Anais XXV Encontro Nacional de Engenharia de Produção, Porto Alegre, 2005

PALMA, F.; FARZIN, H. ; GUEHENEUC, Y. ; MOHA, N. Recommendation system for design patterns in software development: An DPR overview - Third International Workshop on Recommendation Systems for Software Engineering (RSSE), 2012.

PEZZE, M. From Off-Line to Continuous On-line Maintenance - 28th IEEE International Conference on Software Maintenance (ICSM), 2012.

PHEK L.T.; CHU J.N.; SWEE J.T. ; SULAIMAN, S. Improving a web application using design patterns: A case study - International Symposium in Information Technology (ITSim), 2010**.**

PRESSMAN, R.S. Engenharia de Software. 3ª ed. São Paulo, Pearson Education do Brasil, 2003, Cap. 3, p. 60-63.

RAHMAN, A.A.; SAHIBUDDIN, S. ; IBRAHIM, S. A study of process improvement best practices - International Conference on Information Technology and Multimedia (ICIM), 2011.

Rodriguez, M.; PIATTINI, M. Systematic Review of Software Product Certification - 7th Iberian Conference on Information Systems and Technologies (CISTI), 2012.

ROMER, P.; TROGER, P. Reliability Implications of Register Utilization: An Empirical Study - IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing (DASC), 2011.

SAHA, S.K.; SELVI, M. ; BUYUKCAN, G. ; MOHYMEN, M. A Systematic Review on Creativity Techniques for Requirements Engineering - International Conference on Informatics, Electronics & Vision (ICIEV), 2012.

SAMPAIO, RF; MANCINI, MC. Estudos de Revisão Sistemática: Um guia para síntese criteriosa da evidência científica. Rev. Bras. Fisioter. São Carlos, v. 11, n. 1, p. 83-89, jan. 2007.

SANTOS, J.P.; MOREIRA, Ana ; ARAÚJO, J. ; GOULAO, M. Increasing Quality in Scenario Modelling with Model-Driven Development - Seventh International Conference on the Quality of Information and Communications Technology (QUATIC), 2010.

SCHILLING, W.W. Modeling the Reliability of Existing Software using Static Analysis - IEEE International Conference on Electro/information Technology. Páginas 366-371. Maio de 2006

SCHNEIDEWIND, N.F. Reliability Modeling for Safety-Critical Software - IEEE Transactions on Reliability, mar 1997.

SILVA, T.; MARTIN, A. ; MAURER, F. ; SILVEIRA, M. User-Centered Design and Agile Methods: A Systematic Review - Agile Conference (AGILE), 2011.

SHAOYING L.; XIANG X. Automated Software Specication and Design Using the SOFL Formal Engineering Method - WCSE '09. WRI World Congress on Software Engineering, 2009.

SRIKANTH, H.; COHEN, M.B.; XIAO Qu. Reducing Field Failures in System Configurable Software: Cost-Based Prioritization - ISSRE '09. 20th International Symposium on Software Reliability Engineering, 2009.

SVENSSON, R.B.; HOST, M. ; REGNELL, B. Managing Quality Requirements: A Systematic Review - 36th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA), 2010.

TERRY, J.E.; HOPE, S. The Simulation of Formal Management and Technical Reviews to Increase Quality in Undergraduate Software Engineering Projects - 1998 International Conference Software Engineering: Education & Practice, 1998.

TONINI A.C.; CARVALHO M.M.; SPINOLA M.M. Contribuição dos modelos de qualidade e maturidade na melhoria dos processos de software. Produção, v. 18, n. 2, ago. 2008, p. 275-286

TRIOLA, M. F. Introdução à Estatística. 9ª Edição. Rio de Janeiro: LTC, 2005

YASAR, A.; PREUVENEERS, D. ; Berbers, Y. ; Bhatti, G. Best practices for software security: An overview - IEEE International Multitopic Conference, 2008. INMIC 2008.

YATES, W.D.; SHALLER, D.A. Reliability engineering as applied to software - Annual Reliability and Maintainability Symposium, 1990.

YUYU Y.; SHEN G. Research and establishment of quality cost oriented software testing model - CCECE '06. Canadian Conference on Electrical and Computer Engineering, 2006.

YU, A.G. Software crisis, what software crisis? - MASS '09. International Conference on Management and Service Science, 2009.

WALIA, G.S.; CARVER, J.C. Evaluating the Use of Requirement Error Abstraction and Classification Method for Preventing Errors during Artifact Creation: A Feasibility Study - IEEE 21st International Symposium on Software Reliability Engineering (ISSRE), 2010.

WILLIAMS, L. Instilling a Defect Prevention Philosophy - FIE '98. 28th Annual Frontiers in Education Conference, 1998.

WU, J. Stress Testing Software to determine Fault Tolerance for Hardware Failure and Anomalies - 2012 IEEE AUTOTESTCON.