Uma Abordagem Probabilística para Análise Causal de Defeitos de Software

Marcos Kalinowski[1], Guilherme Horta Travassos[1]

[1]COPPE/UFRJ – Universidade Federal do Rio de Janeiro
Caixa Postal 68511 – CEP 21.945-970 – Rio de Janeiro – Brasil

{mkali,ght}@cos.ufrj.br

Resumo. *Análise causal de defeitos tem se mostrado uma forma eficiente para melhoria de processos com base no produto. Nesta tese uma abordagem probabilística para análise causal, chamada DPPI (Defect Prevention-Based Process Improvement) foi elaborada com base em evidências obtidas a partir de quatro rodadas de revisão sistemática da literatura e feedback obtido de especialistas da área. DPPI representa uma abordagem inovadora que integra mecanismos de aprendizado de causa e efeito (redes Bayesianas) nos procedimentos de análise causal de defeitos. Adicionalmente, para facilitar o uso destes mecanismos em reuniões de análise causal, o tradicional diagrama de causa e efeito foi estendido para um diagrama de causa e efeito probabilístico. DPPI foi aplicada a um projeto real e avaliada através de três rodadas de um estudo experimental. A aplicação ao projeto real indicou sua viabilidade e permitiu refinar requisitos para a construção de apoio ferramental. As rodadas do estudo experimental forneceram indícios de que o uso dos diagramas de causa e efeito probabilísticos de DPPI aumenta a eficácia e reduz o esforço na identificação de causas de defeitos, quando comparado à identificação de causas de defeitos sem o uso dos diagramas.*

1. Introdução

1.1. Contexto

Análise causal e resolução é uma maneira de se identificar causas de problemas e tomar ações para preveni-los de reocorrer no futuro. Ela é considerada em normas, modelos e abordagens de melhoria de processos, tais como Six Sigma (Eckes, 2001), ISO/IEC 12207 (ISO/IEC, 2008), CMMI (Chrissis et al., 2011) e MPS (SOFTEX, 2011) . Robitaille (2004) destaca a análise causal e resolução de problemas como uma forma de identificar oportunidades de melhoria para os ativos de processo da organização e não apenas para projetos isolados.

A análise causal de defeitos de software compreende aplicar análise causal e resolução a um tipo específico de problema: os defeitos dos artefatos de software gerados ao longo do processo de desenvolvimento. O escopo desta tese é relacionado à análise causal de defeitos de software identificados através de inspeções de software. Outros problemas, como desvios no cronograma, não foram tratados diretamente nesta pesquisa.

Análise causal de defeitos de software pode ainda ser considerada parte do processo de prevenção de defeitos, que trata também a implementação das ações e a comunicação das mudanças implementadas à equipe de desenvolvimento. Uma

representação das atividades do processo tradicional de prevenção de defeitos de software (Mays et al., 1990) encontra-se na Figura 1.

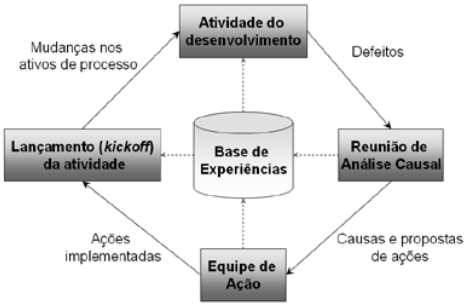

Figura 1. Atividades do Processo Tradicional de Prevenção de Defeitos. Adaptado de (Mays *et al.*, 1990).

Além da reunião de análise causal, a Figura 1 destaca a implementação das ações e a comunicação das mudanças à equipe de desenvolvimento. Adicionalmente, a figura indica que a aplicação de atividades de análise causal de defeitos pode colaborar para a formação de uma base de experiências na organização.

Considerando a reunião de análise causal de defeitos em si, Card (2005) a descreve resumidamente através dos seis seguintes passos: (1) selecionar uma amostra dos defeitos, (2) classificar os defeitos selecionados, (3) identificar erros sistemáticos, (4) identificar as principais causas, (5) desenvolver itens de ação e (6) documentar os resultados da reunião de análise causal. Neste contexto, um erro sistemático é um erro que resulta em defeitos similares se repetindo em diferentes ocasiões. Encontrar erros sistemáticos indica a existência de oportunidades de melhoria para os ativos de processo. Além destes seis passos, Card (2005) destaca a importância de efetivamente gerenciar a implementação dos itens de ação até sua conclusão e de comunicar as modificações à equipe de desenvolvimento.

A análise causal de defeitos de software vem sendo discutida desde a década de 70 (Endres, 1975) e é apontada por Boehm (2006) como uma das principais contribuições desta década para a engenharia de software. Desde então, o processo de análise causal de defeitos de software têm sido implantado na indústria, tanto para projetos envolvendo centenas de pessoas (Mays *et al*., 1990) (Leszak *et al*., 2002) quanto para projetos menores (Dangerfield *et al*., 1992) (Yu, 1998) (Jalote e Agrawal, 2005).

Diversos benefícios associados à aplicação de análise causal de defeitos têm sido relatados. Um dos benefícios diretos é a redução na taxa de defeitos, que com análise causal de defeitos pôde ser reduzida em mais de cinquenta porcento em diferentes contextos organizacionais, como o da IBM (Mays *et al*., 1990), da Computer Science Corporation (Dangerfield *et al*., 1992), da HP (Grady, 1996) e da InfoSys (Jalote e Agrawal, 2005). Como consequência, a análise causal de defeitos de software tem mostrado diminuir a quantidade de retrabalho (Jalote e Agrawal, 2005) e aumentar a probabilidade de alcançar os objetivos de qualidade e de desempenho do processo de desenvolvimento. Adicionalmente, ela pode servir como instrumento para a difusão de lições aprendidas entre as equipes envolvidas nos diferentes projetos da organização (Chrissis et al., 2011).

1.2. Motivação

Apesar de seus benefícios e adoção na indústria, pouca pesquisa tem sido feita a respeito de análise causal de defeitos de software e pouco conhecimento científico tem sido gerado e publicado (Card, 2005). Assim, muitas dúvidas permanecem sobre como implementar análise causal de defeitos de maneira sistemática e eficiente em organizações de software.

Visando fornecer respostas mais isentas e baseadas em evidência para estas dúvidas, uma revisão sistemática foi conduzida, permitindo a compilação de diretrizes para apoiar a implementação de análise causal de defeitos em organizações de software e a identificação de oportunidades para investigação futura (Kalinowski et al., 2012) (Kalinowski et al., 2008a).

As abordagens encontradas durante a revisão sistemática não forneceram um detalhamento para a realização das tarefas associadas à análise causal de defeitos de software, considerando evidências, identificando e apoiando o uso das melhores práticas de análise causal de defeitos.

Adicionalmente, nenhuma das abordagens encontradas provia mecanismos para manter e utilizar o conhecimento gerado a respeito de relacionamentos causais ao longo das diferentes sessões de análise causal. Ou seja, o objetivo das abordagens é somente a prevenção dos defeitos ao atuar em sua causa, sendo a elucidação de relacionamentos causais que permitam uma compreensão mais ampla das efetivas causas dos defeitos para aquele contexto organizacional negligenciada.

Para ser mais preciso, a Figura 2 ilustra os elementos de um sistema causal (Card, 2005). Esta figura destaca os elementos observáveis, as ações que devem ser tomadas sobre estes elementos e os objetivos em tomar tais ações. Entender os relacionamentos causais entre os elementos de tal sistema para um contexto organizacional específico poderia apoiar uma prevenção de defeitos mais eficiente, fornecendo respostas para perguntas como: “Dado um contexto organizacional, que causas costumam levar a que tipo de defeito”, ou “Dado um contexto organizacional, qual costuma ser o impacto de tomar uma ação para prevenir uma causa específica sobre um tipo específico de defeito”.

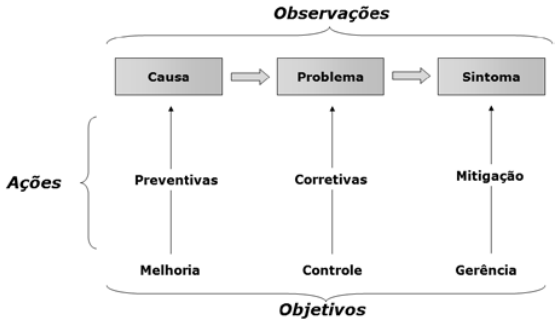

Figura 2. Elementos de um Sistema Causal. Adaptado de (Card, 2005).

De acordo com Pearl (2000), o objetivo central de muitas pesquisas, nas diferentes ciências, é a elucidação de relações de causa e efeito entre variáveis ou eventos. A inexistência de uma abordagem de análise causal de defeitos que permita estabelecer e utilizar conhecimento a respeito de relações causais associadas aos

defeitos da organização reflete uma lacuna a ser preenchida de modo que hipóteses associadas a tal abordagem possam ser avaliadas.

1.3. Objetivos

O objetivo principal desta tese foi definir uma abordagem para análise causal de defeitos de software que forneça para as organizações um detalhamento das práticas a serem realizadas e permita obter conhecimento a respeito das relações causais dos defeitos do contexto organizacional. Este objetivo pode ser desmembrado nos seguintes objetivos específicos:

- O1: Estabelecer um mecanismo sistemático e dinâmico para a identificação e constante atualização de um conjunto de boas práticas para a realização eficiente das tarefas associadas à análise causal de defeitos de software.
- O2: Organizar este conjunto de práticas no contexto de uma abordagem de análise causal de defeitos de software. Esta abordagem deve prover um mecanismo para manter e utilizar o conhecimento gerado a respeito de relacionamentos causais ao longo das diferentes sessões de análise causal.
- O3: Avaliar a viabilidade de aplicação da abordagem resultante em projetos de software reais e prover apoio ferramental para facilitar sua utilização.
- O4: Avaliar os benefícios que esta abordagem traz em relação a diferentes aspectos, como: (i) eficácia na identificação das principais causas, (ii) esforço na identificação das principais causas e (iii) satisfação dos usuários ao realizar análise causal.

1.4. Organização do Trabalho

Tendo apresentado o contexto, a motivação e os objetivos da pesquisa, a seção seguinte deste artigo (Seção 2) descreve a metodologia que foi adotada visando atingir estes objetivos. As demais seções contém informações resumidas sobre as revisões sistemáticas conduzidas (Seção 3), as diretrizes elaboradas para apoiar a implementação de análise causal de defeitos em organizações de software (Seção 4), a abordagem probabilística para análise causal de defeitos (Seção 5), a prova de conceito e o apoio ferramental (Seção 6) e as rodadas do estudo experimental (Seção 7). A seção 8 apresenta as considerações finais.

2. Metodologia de Pesquisa

A metodologia de pesquisa utilizada neste trabalho foi fundamentada nos conceitos da engenharia de software experimental (Wohlin et al., 2000) e na metodologia descrita em (Mafra et al., 2006), sendo apoiada pela condução de estudos secundários e primários. Conforme sugerido em (Mafra et al., 2006), a metodologia é dividida em dois passos: (1) definição e (2) refinamento da tecnologia.

O primeiro destes passos envolve a realização de estudos secundários (revisões sistemáticas) com o intuito de fundamentar a elaboração de proposta para uma nova tecnologia. Assim, esta proposta utiliza o conhecimento adquirido e as evidências identificadas através da condução de revisões sistemáticas. O segundo passo, por sua vez, envolve a aplicação de estudos primários (provas de conceito, estudos de caso e

estudos experimentais, entre outros) para refinar a tecnologia ou ampliar a compreensão a respeito da mesma.

A Figura 3 representa a instanciação desta metodologia, no contexto desta pesquisa, do início do doutorado até o exame de qualificação, destacando as atividades realizadas e os respectivos períodos. Como pode ser visto na Figura 3, este período teve foco nos objetivos específicos O1 (identificação do conjunto de boas práticas) e O2 (concepção da abordagem). Desta forma, envolveu principalmente a execução de estudos secundários e a concepção inicial da abordagem de análise causal de defeitos.

Inicialmente, uma revisão informal da literatura foi conduzida antes do planejamento da revisão sistemática, possibilitando ampliar o conhecimento sobre a área a ser pesquisada. Após o planejamento da revisão sistemática a mesma foi conduzida (execução e análise dos resultados) duas vezes, permitindo a identificação de conhecimento e de oportunidades de pesquisa a respeito de análise causal de defeitos, considerando publicações indexadas até Setembro de 2007. Com base neste conhecimento, diretrizes para apoiar a implementação de análise causal de defeitos de software puderam ser elaboradas (Kalinowski et al., 2008a).

Tendo em vista as oportunidades de pesquisa e estas diretrizes, uma proposta de pesquisa pode ser definida e uma abordagem inicial endereçando esta proposta pôde ser elaborada (Kalinowski et al., 2008b). Por fim, para atualizar a revisão bibliográfica para a escrita do exame de qualificação uma nova execução da revisão sistemática foi conduzida considerando publicações indexadas até Janeiro de 2009.

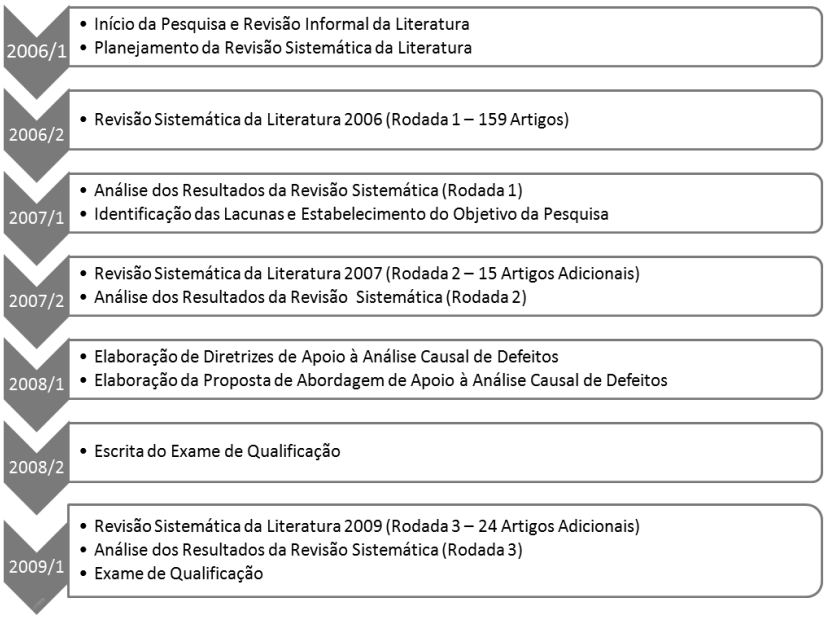

Figura 3. Instanciação da Metodologia de Pesquisa Adotada até o Exame de Qualificação.

A Figura 4 representa a instanciação do restante da metodologia, no contexto desta pesquisa, para o período entre o exame de qualificação e a apresentação da tese de doutorado, destacando as atividades realizadas e os respectivos períodos.

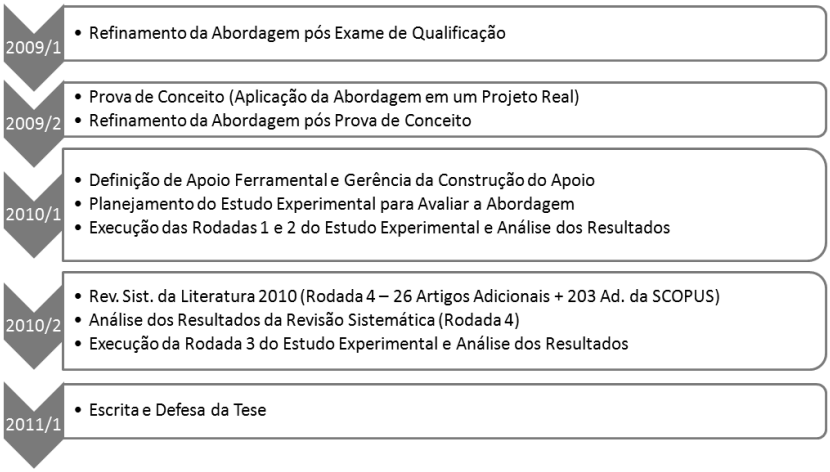

Figura 4. Instanciação da Metodologia de Pesquisa Adotada entre o Exame de Qualificação e a Defesa da Tese.

Como pode ser visto na Figura 4, este período teve foco em refinar a abordagem e nos objetivos específicos O3 (avaliar a viabilidade de utilizar a abordagem e prover apoio ferramental) e O4 (avaliar benefícios obtidos pelo uso da abordagem). Assim, além do refinamento da abordagem, envolveu uma prova de conceito (Kalinowski *et al*., 2010), a construção de apoio ferramental e a avaliação dos benefícios através de três rodadas de um estudo experimental (Kalinowski et al., 2011). Adicionalmente, para atualizar a revisão bibliográfica para a escrita da tese, uma nova rodada da revisão sistemática foi conduzida considerando publicações indexadas até Julho de 2010, permitindo também a atualização das diretrizes (Kalinowski et al., 2012).

3. Revisões Sistemáticas e Diretrizes para Análise Causal de Defeitos

O objetivo da revisão sistemática foi conduzir uma revisão imparcial e justa a respeito do estado da arte de análise causal de defeitos de software. Este objetivo pôde ser desmembrado nos seguintes objetivos específicos:

- Resumir os processos, abordagens e diretrizes (conhecimento) que têm sido propostos para análise causal de defeitos de software.
- Adicionalmente, resumir e analisar os esquemas de classificação dos defeitos e das causas utilizados por estes processos e abordagens. É importante deixar claro que este objetivo não trata de analisar todos os esquemas de classificação existentes para defeitos e causas de defeitos, mas focar naqueles provenientes de fontes relacionadas a análise causal de defeitos de software.

Duas questões de pesquisa foram formuladas com base nestes objetivos. A partir destas questões de pesquisa strings de busca puderam ser derivadas e ajustadas para que pudessem ser executadas em diferentes bibliotecas digitais. As bibliotecas digitais escolhidas inicialmente foram: *ACM Digital Library*, *EI Compendex*, *IEEE*, *Inspec* e *Web of Science*.

Conforme sugerido em (Kitchenham, 2004) o protocolo da revisão foi revisto por outros pesquisadores da área antes de sua execução e a string de busca foi avaliada contra um conjunto de sete artigos de controle a respeito de análise causal de defeitos de software, lidos antes de executar a revisão sistemática.

O critério para incluir um artigo recuperado na busca era que ele deveria conter processos, abordagens ou conhecimento a respeito de analisar causas de defeitos ou análise causal de defeitos (prevenção de defeitos). Para cada um dos artigos incluídos as seguintes informações foram extraídas: título, referência completa, fonte, esquema de classificação de defeitos, esquema de classificação de causas, identificação do processo ou abordagem, descrição do processo ou abordagem, identificação de conhecimento, descrição do conhecimento e tipo de estudo.

Com o protocolo da revisão sistemática estabelecido, a execução do estudo pôde ser iniciado. O protocolo da revisão sistemática foi planejado para que pudessem ocorrer repetições periódicas de atualização. O protocolo foi executado em Agosto de 2006, Setembro de 2007, Janeiro de 2009 e Julho de 2010. Adicionalmente, após uma verificação das bibliotecas utilizadas, em Agosto de 2010 incluiu-se também a biblioteca SCOPUS e uma execução complementarfoi realizadavisando aumentar a cobertura. Ao todo, 427 artigos (sem duplicatas) foram retornados e 72 atendiam explicitamente aos critérios de inclusão.

As seguintes informações, referentes aos objetivos da revisão sistemática, foram agrupadas em tabelas e organizadas por data de publicação após sua extração dos artigos incluídos:

- Processos e abordagens utilizadas para análise causal de defeitos (veja Anexo C1);
- esquemas de classificação de defeitos utilizados nos artigos (veja Anexo C2);
- esquemas de classificação de causas utilizados nos artigos (veja Anexo C3), e;
- outros conhecimentos a respeito de análise causal de defeitos gerado ou citado pelos artigos (veja Anexo C4).

As análises que puderam ser realizadas com base nestas tabelas estão detalhadas no texto da tese (Kalinowski, 2011). Os resultados da revisão sistemática foram então utilizadas para a elaboração de diretrizes iniciais (Kalinowski et al., 2008a) que pudessem apoiar a implementação de análise causal de defeitos em organizações desenvolvedoras de software, devidamente atualizadas após a realização da ultima execução do protocolo da revisão sistemática em 2010 (Kalinowski et al.,2012).

As diretrizes oferecem respostas para perguntas que podem ser realizadas por engenheiros de software quando tentam implementar análise causal de defeitos em organizações de software. Estas respostas podem ser utilizadas em conjunto com as tabelas de conhecimentos a respeito do processo de análise causal de defeitos e suas atividades (um dos anexos da tese) para apoiar a implementação eficiente de análise causal de defeitos. Mais sobre as diretrizes pode ser encontrado no corpo da tese ou em (Kalinowski et al., 2008a) e (Kalinowski et al., 2012).

Embora importantes, as diretrizes representam apenas uma contribuição parcial desta tese. Sua elaboração visou a obtenção de uma compreensão mais ampla da área para posterior proposição e avaliação de uma abordagem probabilística para análise causal de defeitos conforme descrito na seção seguinte.

4. Abordagem Probabilística para Análise Causal de Defeitos de Software

Com base nos resultados das duas primeiras rodadas da revisão sistemática (2006 e 2007) e nas diretrizes resultantes, uma proposta inicial de abordagem para melhoria de

processos baseada em prevenção de defeitos pôde ser elaborada (Kalinowski e Travassos, 2008) e refinada (Kalinowski *et al*., 2008b). Esta proposta representa uma inovação, no sentido que foi a primeira sugerindo a integração de um mecanismo de aprendizado das relações de causa e efeito (utilizando Redes Bayesianas) às reuniões de análise causal de defeitos de software para facilitar a identificação das causas.

Posteriormente esta proposta foi evoluída e detalhada, resultando na abordagem DPPI (*Defect Prevention-based Process Improvement*) (Kalinowski et al., 2010). A evolução e o refinamento se deram com base em: (i) resultados de uma execução adicional da revisão sistemática (realizada em 2009), (ii) no retorno obtido de especialistas da área e (iii) da experiência de aplicar a abordagem a um projeto real de desenvolvimento de software . O restante desta seção fornece uma visão geral da abordagem DPPI e de sua principal inovação, maiores detalhes sobre cada uma de suas atividades e tarefas são discorridos na tese (Kalinowski, 2011).

DPPI representa uma abordagem prática para análise causal de defeitos provenientes de inspeções de software, visando sua prevenção. Quando comparada ao processo tradicional de prevenção de defeitos, descrito em (Jones, 1985), a principal inovação é a integração de conhecimento obtido em sucessivas reuniões de análise causal para prover um maior entendimento a respeito das relações de causa e efeito dos defeitos da organização. Isto trata uma das oportunidades de pesquisa identificadas nas revisões sistemáticas e destacada em (Kalinowski *et al*., 2008a). Até a data da condução da última rodada da revisão sistemática, nenhuma outra abordagem considerava esta possibilidade de integração.

Tal integração permite estabelecer e manter modelos causais para a organização. Estes modelos podem então ser utilizados para apoiar o raciocínio diagnóstico, facilitando a identificação das principais causas dos defeitos sob análise em uma nova reunião de análise causal. Com estes modelos é possível, por exemplo, apoiar o entendimento, com base no aprendizado obtido de projetos similares da organização, sobre quais causas normalmente contribuem para determinado tipo de defeitos.

Além desta inovação, DPPI segue as diretrizes para implementar análise causal em organizações de software, produzidas como parte da tese (Kalinowski *et al*., 2008a) (Kalinowski et al., 2012). O uso das diretrizes permitiu detalhar as atividades de prevenção de defeitos em tarefas mais específicas, fornecendo detalhes adicionais sobre as técnicas a serem utilizadas para realizar estas tarefas eficientemente.

Ainda com base nas diretrizes, DPPI integra a prevenção de defeitos na estratégia de medição e controle das atividades de engenharia do software para as quais a prevenção de defeitos está sendo realizada, permitindo observar se as melhorias implementadas para estas atividades trouxeram benefícios em relação à taxa de inserção de defeitos. Isto é feito ao utilizar as métricas relacionadas a defeitos definidas nas diretrizes para medir e controlar o desempenho destas atividades.

Adicionalmente, DPPI trata todas as práticas específicas da área de processos CAR (*Causal Analysis and Resolution*) do modelo CMMI (Chrissis et al., 2011). Desta forma, seguir a abordagem DPPI resulta em aderência a estas práticas em relação à análise causal de defeitos de software provenientes de inspeções. Entretanto, a área de processos CAR do CMMI pode também ser utilizada como referência de práticas para realizar a análise causal de outros problemas, que estão fora do escopo de DPPI.

DPPI visa a realização de análise causal para reduzir continuamente as taxas de inserção de defeitos das atividades de engenharia do software e, conforme sugerido nas diretrizes, ela foi projetada para ocorrer de forma integrada ao processo de desenvolvimento, sempre imediatamente após as inspeções dos artefatos produzidos por estas atividades. Ou seja, DPPI deve ser aplicada tanto quando o desempenho da atividade estiver dentro do esperado (ou estável) ou não. Assim, o momento de aplicação de DPPI dentro de um processo de desenvolvimento de software é após as inspeções dos artefatos produzidos pelas principais atividades de engenharia do software.

A Figura 5 ilustra os momentos de aplicação de DPPI considerando um processo iterativo e incremental. Nesta figura DPPI foi aplicada duas vezes, após a inspeção da especificação funcional produzida na atividade de levantamento e especificação de requisitos, visando melhorar esta atividade para os próximos módulos e após a inspeção da especificação técnica produzida na atividade de projeto do produto, visando melhorar também esta atividade para os próximos módulos.

Nesta figura é possível ainda observar que DPPI tem a lista de defeitos e o modelo causal da atividade como entrada. O modelo causal visa apoiar a identificação das causas dos defeitos e será atualizado a cada aplicação de DPPI. Maiores detalhes sobre o modelo causal e esta atualização serão fornecidos mais adiante. É certo que este contexto de utilização impõe algumas premissas para a aplicação de DPPI. Estas premissas se encontram detalhadas na tese.

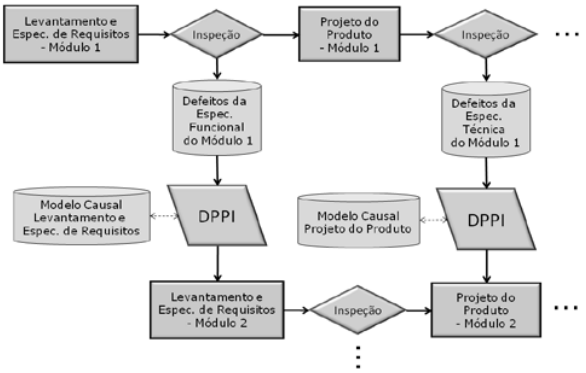

Figura 5. Momentos de Aplicação de DPPI em um Processo Iterativo Incremental.

Tendo compreendido o momento de aplicação de DPPI, a abordagem em si envolve quatro atividades: (i) Análise da Atividade de Desenvolvimento; (ii) Preparação para Análise Causal; (iii) Reunião de Análise Causal; e (iv) Melhoria da Atividade de Desenvolvimento. A Figura 6 fornece uma visão das tarefas e dos papéis envolvidos na execução destas atividades. Mais detalhes sobre as atividades de DPPI, suas tarefas e técnicas a ser empregadas, bem como exemplos de sua aplicação podem ser encontrados na tese ou em (Kalinowski et al., 2010).

A Figura 6 também destaca a proposta de manter, para cada atividade de engenharia do software, um modelo causal com informações a respeito das relações de causa e efeito dos defeitos da organização. Em DPPI tais modelos causais devem ser estabelecidos e mantidos ao alimentar redes Bayesianas.

Pearl (2000) afirma que modelos causais probabilísticos podem ser elaborados utilizando redes Bayesianas caso exemplos concretos de relações causais entre variáveis

aleatórias possam ser obtidos para alimentar a rede. Assim, a estrutura e os parâmetros de distribuição de probabilidade podem ser aprendidos a partir destes exemplos. Após a alimentação destas redes elas podem ser utilizadas tanto para inferência preditiva quanto para inferência diagnóstica.

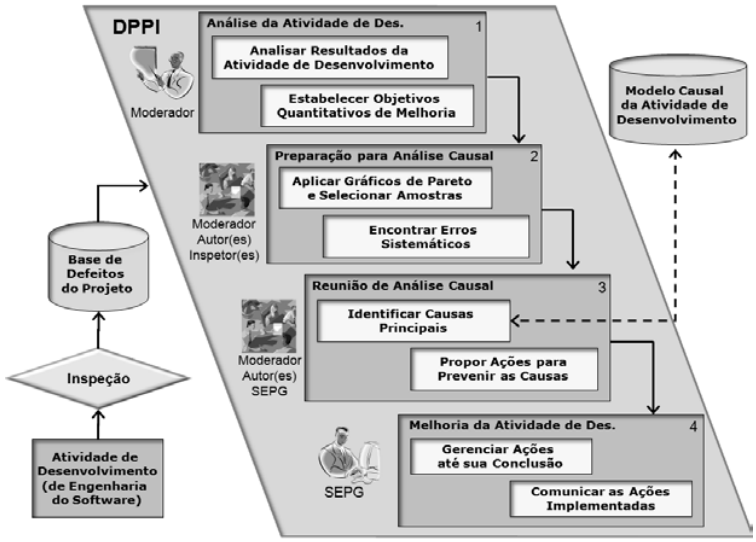

Figura 6. Visão Geral da Abordagem DPPI.

No caso de DPPI, os exemplos de relações causais para alimentar a rede Bayesiana são os próprios resultados de cada uma das reuniões de análise causal (consenso da equipe a respeito das principais causas dos defeitos analisados). Posteriormente, a inferência diagnóstica desta rede deve ser utilizada para apoiar as próximas reuniões de análise causal, fechando explicitamente um ciclo de retroalimentação a respeito das causas de defeitos de software da organização. A Figura 7 ilustra o ciclo de retroalimentação e inferência proposto por DPPI para apoiar a identificação de causas de defeitos.

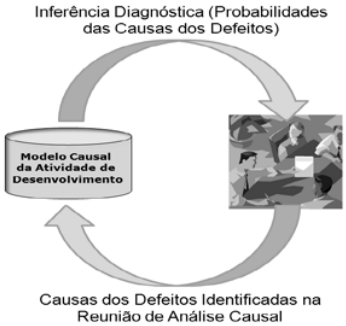

Figura 7. Ciclo de Retroalimentação de DPPI – Os Modelos Causais são Construídos com os Próprios Resultados das Reuniões de Análise Causal.

A proposta de DPPI de atualizar as redes bayesianas dinamicamente com relações causais concretas, identificadas nas reuniões de análise causal de defeitos de software, e utilizar a inferência diagnóstica da rede para apoiar a identificação de causas de defeitos nas próximas reuniões de análise causal representa uma proposta nova e anteriormente não explorada.

Adicionalmente, para facilitar a utilização das inferências diagnósticas durante reuniões de análise causal, o tradicional diagrama de causa e efeito (Ishikawa, 1976) foi estendido em um diagrama de causa e efeito probabilístico (Kalinowski et al., 2008b). As extensões envolvem (i) mostrar as probabilidades de cada uma das causas estar contribuindo para o tipo de defeito sendo analisado e (ii) representar as causas utilizando tons de cinza, onde os tons mais escuros são utilizados para causas com maior probabilidade e devem ficar mais próximos da linha central. As probabilidades são obtidas transcrevendo a inferência Bayesiana para a categoria de defeito em questão.

A inferência diagnóstica Bayesiana e o diagrama de causa e efeito probabilístico correspondente para fatos incorretos em especificações funcionais do projeto real em que a abordagem foi aplicada estão ilustrados na Figura 8 (a figura se encontra em inglês em função dos dados utilizados para a construção da rede bayesiana). Esta figura mostra que, neste projeto, as causas para fatos incorretos usualmente são "Falta de Conhecimento do Domínio" (25%), "Tamanho e Complexidade do Domínio do Problema" (18,7%), "Descuido" (15,6%) e "Ferramental limitado para Rastreabilidade" (12,5%).

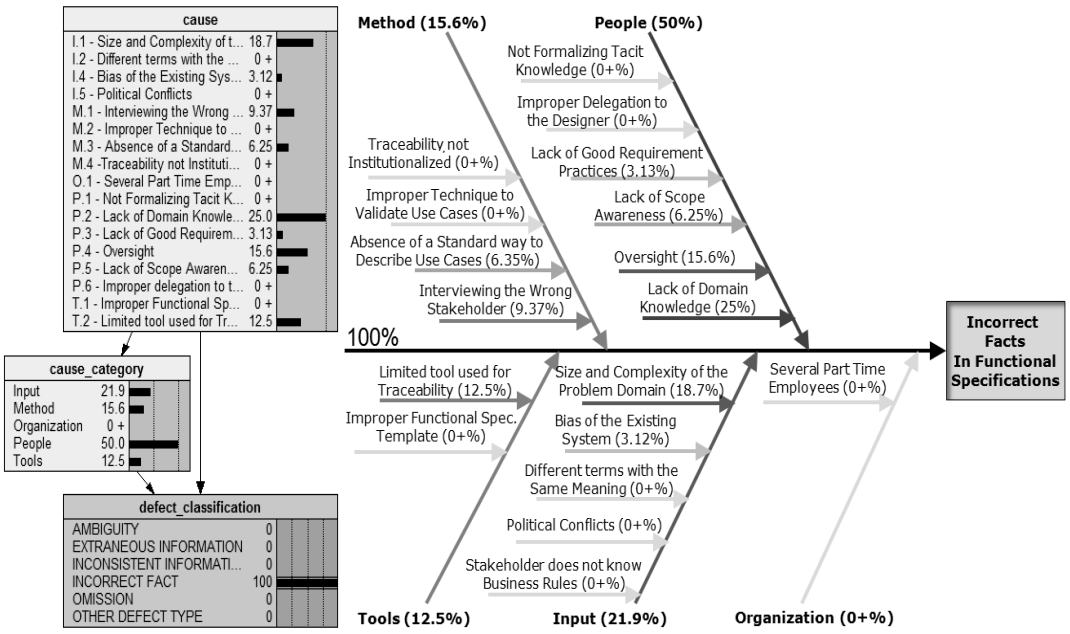

Figura 8. Inferência Bayesiana e o Diagrama de Causa e Efeito Probabilístico Correspondente.

Tendo fornecido uma visão geral de DPPI e de sua principal inovação, a seção seguinte resume a prova de conceito que envolveu a aplicação das atividades da abordagem a um projeto real e discorre sobre o apoio ferramental produzido com base nesta experiência.

5. Prova de Conceito e Apoio Ferramental

Visando avaliar a viabilidade de aplicar a projetos reais, DPPI foi aplicada manualmente e retroativamente para realizar análise causal de defeitos na atividade de especificação funcional de um projeto real Web de larga escala. O projeto utilizou um ciclo de vida

iterativo e incremental e DPPI foi aplicada para apoiar a análise causal dos defeitos da atividade de especificação funcional do quarto módulo.

Esta experiência indicou a viabilidade de utilização da abordagem e mostrou que sua aplicação manual pode envolver um esforço significativo. Desta forma, forneceu indicações relevantes a respeito do apoio ferramental necessário para sua aplicação na indústria (Kalinowski et al., 2010). Durante esta experiência a inferência diagnóstica da rede *Bayesiana* apoiou a identificação das causas principais eficientemente, motivando avaliações adicionais.

O apoio ferramental construído, por sua vez, facilita a aplicação sistemática de DPPI e fecha o ciclo automatizado de retroalimentação proposto, permitindo a geração dos diagramas de causa e efeito probabilísticos (refletindo a inferência diagnóstica Bayesiana) sem requerer esforço adicional, resolvendo a questão do esforço da aplicação manual. Entretanto, os benefícios de utilizar tais diagramas durante reuniões de análise causal ainda não haviam sido avaliados objetivamente; embora a utilização durante a experiência de aplicar DPPI ao projeto real tenha mostrado resultados promissores.

A investigação adicional através de três rodadas de um estudo experimental (conduzido em um problema real e com profissionais de desenvolvimento de software) dos benefícios da utilização de diagramas de causa e efeitos probabilísticos durante reuniões de análise causal para apoiar a identificação das causas é descrita na seção seguinte.

6. Avaliação Experimental da Abordagem

Esta seção resume o estudo experimental conduzido para avaliar se o uso da abordagem de DPPI, que faz uso de diagramas de causa e efeitos probabilísticos gerados a partir de sessões de análise causal anteriores, traz benefícios para a identificação de causas de defeitos de software provenientes de inspeções. Informações mais detalhadas sobre o projeto do estudo, o tratamento dado às ameaças de validade e os resultados podem ser encontrados na tese ou em (Kalinowski et al., 2011).

O projeto do estudo envolveu diversos arranjos entre dois tratamentos (aplicar DPPI ou não), dois objetos (defeitos de dois módulos diferentes de um projeto real) e quatro participantes, organizados em três rodadas separadas do estudo. Estas rodadas permitiram analisar as hipóteses do estudo com base em três cenários distintos de observação.

Os resultados indicaram uma melhoria significativa quando o tratamento foi DPPI, em relação à eficácia na identificação das principais causas dos defeitos e ao tempo gasto na realização da tarefa. No último cenário de observação, em particular, que envolveu a comparação de cada um dos participantes aplicando os dois tratamentos no mesmo objeto, todos os participantes apresentaram maior eficácia na identificação das causas (pelo menos 40% maior) e um menor tempo gasto (pelo menos 20% menor) para a aplicação de DPPI.

De maneira geral, o projeto do estudo e seu arranjo experimental permitiram fornecer argumentos consistentes e que servem como indícios preliminares concretos de benefícios associados ao uso dos diagramas de causa e efeito probabilísticos de DPPI para apoiar a identificação de causas de defeitos em reuniões de análise causal.

7. Considerações Finais

Nesta tese uma metodologia científica baseada em evidência foi seguida para, com base em estudos secundários e primários, definir e refinar uma abordagem para análise causal de defeitos de software. A abordagem representa uma inovação que integra mecanismos de aprendizado de probabilidades de causa e efeito (redes Bayesianas) nos procedimentos de análise causal de defeitos (Kalinowski et al., 2010). Adicionalmente, uma representação (diagrama de causa e efeito probabilístico) foi criada para facilitar o uso do conhecimento provido por estes mecanismos em reuniões de análise causal (Kalinowski et al., 2010).

A aplicação da abordagem em projetos reais se mostrou viável e sua avaliação experimental apontou para benefícios concretos de sua utilização (Kalinowski et al., 2011). Adicionalmente, durante o período de pesquisa um corpo de conhecimento sobre análise causal de defeitos pôde ser reunido (Kalinowski et al., 2012).

7.1. Contribuições e Resultados Obtidos

Analisando os quatro objetivos específicos definidos para a tese, descritos na seção 1.3, as seguintes contribuições e resultados foram obtidos:

- **O1: Estabelecer um mecanismo sistemático e dinâmico para a identificação e constante atualização de um conjunto de boas práticas para a realização eficiente das tarefas associadas à análise causal de defeitos de software.** Um protocolo de revisão sistemática foi elaborado e executado em quatro anos diferentes (2006, 2007, 2009 e 2010), permitindo sua reexecução futura. Diretrizes foram elaboradas com base neste conhecimento (Kalinowski et al., 2008a) (Kalinowski et al., 2012).
- **O2: Organizar este conjunto de práticas no contexto de uma abordagem de análise causal de defeitos de software. Esta abordagem deve prover um mecanismo para manter e utilizar o conhecimento gerado a respeito de relacionamentos causais ao longo das diferentes sessões de análise causal.** A abordagem DPPI foi criada (Kalinowski e Travassos, 2008) (Kalinowski et al., 2008b) e refinada (Kalinowski et al., 2010), organizando as boas práticas identificadas durante a revisão sistemática e explicitadas nas diretrizes. A contribuição principal de DPPI é integrar mecanismos de aprendizado de relações de causa e efeito nas reuniões de análise causal, fazendo uso de redes Bayesianas e dos diagramas de causa e efeito probabilísticos de DPPI.
- **O3: Avaliar a viabilidade de aplicação da abordagem resultante em projetos de software reais e prover apoio ferramental para facilitar sua utilização.** Uma prova de conceito foi realizada, em que a aplicação de DPPI a um projeto de desenvolvimento Web real de larga escala se mostrou viável (Kalinowski et al., 2010). Com base nesta experiência, requisitos para apoio ferramental foram definidos e posteriormente o mesmo foi construído (Kalinowski et al., 2011).
- **O4: Avaliar os benefícios que esta abordagem traz em relação a diferentes aspectos, como: (i) eficácia na identificação das principais causas, (ii) esforço na identificação das principais causas e (iii) satisfação dos usuários ao realizar análise causal.** Um estudo experimental foi conduzido para avaliar

se o uso de DPPI, com diagramas de causa e efeitos probabilísticos gerados a partir de sessões de análise causal anteriores, traz benefícios para a identificação de causas de defeitos de software provenientes de inspeções (Kalinowski et al., 2011). Os resultados gerais deste estudo fornecem indícios de uma melhoria significativa para o tratamento de DPPI, tanto em relação à eficácia na identificação das principais causas dos defeitos quanto ao tempo gasto na tarefa.

7.2. Trabalhos Futuros

Além de representar contribuições para a engenharia de software, em particular na área de análise causal de defeitos, os resultados obtidos até o momento possibilitaram a organização de um arcabouço de pesquisa que permite explorar novas direções de investigação, conforme exemplificado a seguir:

- Avaliar a possibilidade de ampliar o escopo de DPPI para defeitos provenientes de outras atividades de garantia e controle da qualidade.
- Investigar uma forma apropriada para caracterizar a similaridade entre projetos. Desta forma seria possível saber de antemão para quais grupos de projeto DPPI deve ser instanciada separadamente.
- Realizar avaliações adicionais, replicando as já realizadas em outros contextos ou ainda para outras atividades de engenharia do software como, por exemplo, o projeto e construção do produto.
- Estender DPPI para que a abordagem seja capaz de lidar, além de defeitos de software, também com outros problemas que possam vir a ocorrer durante o desenvolvimento de software.

Agradecimentos

Aos pesquisadores David Card e Emilia Mendes por suas colaborações nesta pesquisa. Ao CNPq e à Fundação COPPETEC.

Referências

Boehm, B. (2006) “A View of 20th and 21st Century Software Engineering”, In: Proc. of ICSE '06: 28th International Conference on Software Engineering, ACM Press, New York, NY, USA, pp. 12-29.

Card, D.N. (2005) “Defect Analysis: Basic Techniques for Management and Learning”, In: Advances in Computers, vol. 65, chapter 7, pp. 259-295.

Chrissis, M. B., Konrad, M., Shrum, S. (2011) “CMMI: Guidelines for Process Integration and Product Improvement”, Third Edition, Addison Wesley Professional.

Dangerfield, O., Ambardekar, P., Paluzzi, P., Card, D., Giblin, D. (1992) “Defect Causal Analysis: A Report from the Field”, In: Proc. of the International Conference on Software Quality, American Society for Quality Control.

Eckes, G. (2000) “The Six Sigma Revolution: How General Electric and Others Turned Process Into Profits”, John Wiley and Sons.

Endres, A. (1975) “An Analysis of Errors and Their Causes in Systems Programs”, In: IEEE Transactions on Software Engineering, SE-1, 2, June 1975, pp. 140-149.

Grady, R. B., “Software Failure Analysis for High-Return Process Improvement Decisions”, In: Hewlett-Packard Journal, 47 (4), 15 - 24, 1996.

ISO/IEC (2008) "ISO/IEC 12207:2008 – Systems and software engineering – Software life cycle processes", ISO/IEC 12207:2008.

Jalote, P., Agrawal, N., "Using Defect Analysis Feedback for Improving Quality and Productivity in Iterative Software Development", In: Proc. of the 3rd International Conference on Information and Communication Technology, ICICT, pp. 701–713, Cairo, 2005.

Kalinowski, M., Card, D.N., Travassos, G.H. (2012) "Evidence-based Guidelines on Defect Causal Analysis", IEEE Software Magazine (aceito para publicação).

Kalinowski, M. (2011), "Uma Abordagem Probabilística para Análise Causal de Defeitos de Software", Tese de Doutorado defendida no PESC/COPPE/UFRJ em 21/03/2011, disponível em http://www.cos.ufrj.br/

Kalinowski, M., Mendes, E., Travassos G.H. (2011) "Automating and Evaluating the Use of DPPI's Probabilistic Cause-Effect Diagrams to Improve Defect Causal Analysis", 12th International Conference on Product Focused Software Development and Process Improvement (PROFES 2011), Springer LNCS 6759, pp. 232-246, Bari, Italy.

Kalinowski, M., Travassos G.H., Mendes, E., Card, D.N. (2010) "Applying DPPI: A Defect Causal Analysis Approach Using Bayesian Networks", 11th International Conference on Product Focused Software Development and Process Improvement (PROFES 2010), Springer LNCS 6156, pp. 92–106, Limerick, Ireland.

Kalinowski, M., Travassos, G. H., Card, D. N. (2008b) "Towards a Defect Prevention Based Process Improvement Approach", 34th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), 199-206, Parma, Italy.

Kalinowski, M., Travassos, G.H., Card, D.N. (2008a) "Guidance for Efficiently Implementing Defect Causal Analysis", VII Simpósio Brasileiro de Qualidade de Software (SBQS), Florianópolis.

Kalinowski, M., Travassos, G.H. (2008) "Towards a Defect Causal Analysis Approach for Software Process Improvement and Organizational Learning", Workshop de Teses e Dissertações em Qualidade de Software (WTDQS) do VII Simpósio Brasileiro de Qualidade de Software, Florianópolis.

Kitchenham, B.A., "Procedures for Performing Systematic Reviews", In: Joint Technical Report Keele University and National ICT Australia Ltd., 2004.

Leszak, M., Perry, D. E., Stoll, D. (2002) "Classification and Evaluation of Defects in a Project Retrospective", In: Journal of Systems and Software, 61(3), 173 - 187.

Mafra, S.N., Barcelos, R.F., Travassos, G.H., "Aplicando uma Metodologia Baseada em Evidência na Definição de Novas Tecnologias de Software", In: Proc. of the XX Simpósio Brasileiro de Engenharia de Software (SBES), Florianópolis, 2006.

Pearl, J. (2000) "Causality: Reasoning, Models and Inference", Cambridge University Press, ISBN: 978-0-521-77362-1.

Robitaille, D., "Root Cause Analysis – Basic Tools and Techniques", Paton Press, ISBN: 1-932828-02-8, 2004.

SOFTEX (2011) "Modelo MPS – Melhoria de Processo do Software Brasileiro, Guia Geral do MPS, Versão 2011". Sociedade Softex, Brasil. Disponível em http://www.softex.br/mpsbr.

Wohlin, C., Runeson, P., Host, M., Ohlsson, M.C., Regnell, B., Wesslén, A. (2000) "Experimentation in Software Engineering – An Introduction", Kluwer Academic Publishers, ISBN 0-7923-8682-5.

Yu, W. D. (1998) "Software fault prevention approach in coding and root cause analysis", In: Bell Labs Technical Journal, Vol.3 (2), pp.3-21.