ANÁLISE EXPERIMENTAL DE UMA INFRAESTRUTURA VISUAL MULTIESTRATÉGIA PARA ANÁLISE DE EVOLUÇÃO DE SOFTWARE

Creidiane Muniz Conceição Brito - IFBA *Campus* Santo Amaro. E-mail: creidianemcb@gmail.com;
Renato Lima Novais - IFBA *Campus* Santo Amaro. E-mail: renato@ifba.edu.br.

RESUMO: O *Source Miner Evolution* (SME) é um ambiente de visualização de evolução de software que utiliza diferentes estratégias de análise visual para suporte a determinadas atividades de evolução de software. Como qualquer ambiente proposto, é importante que o SME seja avaliado experimentalmente para medir sua efetividade na realização de atividades de evolução de software. Neste contexto, este trabalho consistiu em planejar e executar estudos experimentais para avaliação do SME. Foram planejados e realizados um *survey* e um estudo de caso, e por fim, foi planejado um experimento controlado. Os resultados dos estudos apontaram pontos fortes e fracos do SME, destacando a importância de avaliações experimentais e abrindo novas oportunidades de pesquisa para o ambiente.

PALAVRAS-CHAVE: estratégias de análise visual, estudo de caso, *survey*, experimento controlado, visualização de software.

1. INTRODUÇÃO

Sistemas de software evoluem com o passar do tempo (Lehman, 1978) (Lehman, 1980). A atividade de desenvolvimento de software torna-se mais complexa à medida que os sistemas vão evoluindo, o que conduz à necessidade de manutenções, preventivas e corretivas. Esta complexidade se reflete nos custos do software associado à manutenção. Por volta de 90% dos custos totais do software estão associados à manutenção (Erlikh, 2000), sendo que 50% do tempo gasto com a manutenção são utilizados na compreensão do software (Fjeldsta e Hamlen, 1983). Custo de tempo muito alto.

A evolução de software está forte e previamente ligada à atividade de compreensão dos artefatos de software. Sendo assim, engenheiros de software têm utilizado métodos, técnicas e ferramentas para facilitar as atividades de compreensão e, consequentemente, a evolução de software. Duas importantes subáreas da Engenharia de Software (ES) têm ajudado nesse sentido: métricas e visualização de software.

Medições têm sido utilizadas para garantir o controle de projetos, produtos e processos de software (Wohlin et al., 2000). Muitas métricas foram propostas na literatura. Através destas métricas é possível caracterizar, por exemplo, questões relativas ao tamanho, complexidade e acoplamento do código (Chidamber e Kemerer, 1994) (Lanza e Marinescu, 2010). Seguindo a mesma linha, as técnicas de visualização de informação (Chen, 2006) e de software (Diehl, 2007) têm sido utilizadas na Engenharia de Software como uma possível solução para a árdua tarefa de compreender, manter e evoluir sistemas de software (Storey et al., 2005). A Visualização de Software (VisSoft) utiliza recursos visuais para facilitar o entendimento do software por parte de programadores e engenheiros de software (Diehl, 2007).

Há uma notória interseção dessas áreas, uma vez que diversos trabalhos de visualização de software utilizam métricas como atributos do software a serem visualizados. Estas métricas podem ser visualizadas seguindo diferentes estratégias de análise visual (Novais et al., 2013). A estratégia diz respeito à forma como a análise da evolução deve ser visualmente apresentada. Diferentes tipos de estratégias são mais interessantes para determinados tipos de atividades de manutenção de software.

O *SourceMiner Evolution* (SME) é um ambiente de visualização de evolução de software que utiliza diferentes estratégias de análise visual para suporte a determinadas atividades de evolução de software (Novais et al., 2011a)(Novais et al., 2011b)(Novais et al., 2011c)(Novais et al., 2012a) (Novais et al., 2012b) (Novais et al., 2012c) (Novais and Mendonça, 2013). Como qualquer am-

biente proposto, é importante que o SME seja avaliado experimentalmente para medir sua efetividade na realização de atividades de evolução de software. Este trabalho consistiu do planejamento e realização de um conjunto de estudos experimentais para avaliação do SME.

No escopo deste trabalho, foi planejada a realização de dois estudos experimentais para validar o SME. No primeiro estudo, foi planejado e executado um estudo de caso (Runeson e Höst, 2009) para avaliar a efetividade do SME para a realização de atividades de evolução de software. Para realizar esse estudo de caso, foi necessário fazer um levantamento na literatura das atividades de evolução de software comumente realizadas por desenvolvedores e engenheiros de software. Durante este levantamento, foi possível observar que muitas questões reportadas na literatura eram similares. Foi feito então um agrupamento das mesmas. Estas questões foram então utilizadas em um *survey* com engenheiros de software e analistas de sistemas, para avaliar qualitativamente tais questões, identificadas na literatura. Finalmente, foi realizado o estudo de caso, que consistiu em avaliar o SME quanto à realização ou não de cada uma das questões do *survey*. No segundo estudo, foi planejado o experimento controlado. Este experimento está em fase de execução, de tal forma que os resultados deste estudo não serão apresentados neste artigo.

Este artigo está organizado da seguinte forma: a Seção 2 apresenta os materiais e métodos utilizados neste projeto, dando ênfase no planejamento do *survey*, estudo de caso e experimento controlado; a Seção 3 apresenta os resultados do *survey* e do estudo de caso; por fim, a Seção 4 conclui este artigo, destacando os trabalhos futuros.

2. MATERIAIS E MÉTODOS

Durante o projeto foram desenvolvidas várias atividades. Tais atividades estão descritas a seguir.

2.1 REVISÃO DE LITERATURA SOBRE ESTUDO DE CASO

Através da revisão bibliográfica relativa à teoria de Estudo de Caso, identificou-se que se trata de uma abordagem de investigação especialmente adequada quando se procura compreender, explorar ou descrever acontecimentos e contextos complexos, nos quais estão simultaneamente envolvidos diversos fatores. Segundo Robson (2002), Yin (2003) et al & Benbasat (1987) apud Runeson et al. (2008), "o estudo de caso é um método empírico que visa investigar fenômenos contemporâneos em seu contexto".

Robson chama de uma estratégia de pesquisa e enfatiza o uso de múltiplas fontes de evidência; Yin denota um inquérito e observações que a fronteira entre o fenômeno e seu contexto pode não ser claro; enquanto Benbasat et al. torna as definições um pouco mais específicas mencionando a coleta de informações a partir de poucas entidades (pessoas, grupos, organizações), e a falta de controle experimental.

As teorias do estudo de caso foram utilizadas para avaliar o ambiente visual multiestratégia para análise de evolução de software.

2.2 LEVANTAMENTO DE QUESTÕES DE EVOLUÇÃO DE SOFTWARE

No estudo de caso, o SME seria avaliado quanto a sua efetividade em se realizar atividades de evolução de software. Foi necessário então, buscar na literatura tipos de atividades de evolução de software que desenvolvedores e engenheiros de software realizam no dia a dia. Para realizar uma atividade de manutenção de software, é comum os desenvolvedores e engenheiros de software fazerem perguntas a respeito da evolução do software.

Foi então feito um levantamento de questões relacionadas à evolução do software. Para tal fim, foram identificados os artigos (Sillito et al., 2006; Zazworka and Ackermann, 2010; Alwis et al., 2008; Hattori et al., 2011; Fritz and Murphy, 2010; D'Ambros and Lanza, 2007; Tu and Godfrey, 2002; German et al., 2006; D'Ambros and Lanza, 2009) como fonte primária dessas questões.

A partir das leituras dos artigos, houve um levantamento das possíveis questões reportadas. Todas elas foram catalogadas em um documento estruturado. Observou-se que muitas das questões eram semelhantes. Após a revisão do material foram agrupadas as questões semelhantes e escolhidas as questões relacionadas à evolução de softwa-

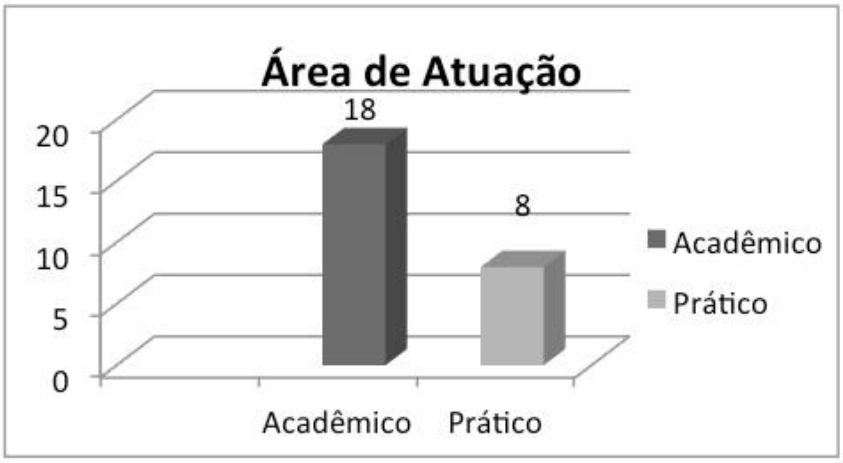
re. Essa coletânea de questões foi utilizada posteriormente como base para a preparação de um *survey*. O *survey* foi aplicado com diferentes pessoas que estão ligadas à academia e a indústrias de desenvolvimento de software.

2.3 CONSTRUÇÃO E EXECUÇÃO DE *SURVEY*

Uma vez identificado um conjunto de questões de evolução de software, foi decidido avaliar o quanto que um grupo de pessoas envolvidas com desenvolvimento de software concordavam com a importância das questões escolhidas. Desta forma, o objetivo do *survey* foi saber se as perguntas escolhidas eram importantes e suficientes para a realização do estudo do estudo de caso. As questões selecionadas foram analisadas por desenvolvedores e gerentes de software. Os participantes do *survey* deveriam responder o quanto que eles achavam relevantes cada uma das questões no contexto de evolução de software.

Como mencionado anteriormente, para a execução do *survey*, houve uma seleção das questões a partir dos artigos encontrados na literatura. O survey foi aplicado com 26 pessoas, todas inseridas na área de engenharia de software, na qual 18 se consideravam mais práticas e 8 mais acadêmicas (Figura 1).

Figura 1 – Área de atuação dos participantes do *survey*.



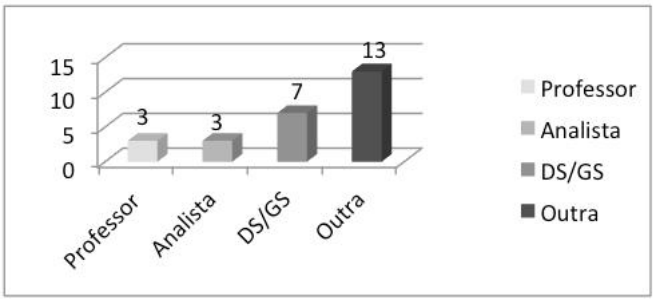
A Tabela 1 apresenta a lista das questões selecionadas para o survey.

Tabela 1 – Lista das questões utilizadas no survey.

#	Questão
1	Qual o impacto de uma mudança no código fonte?
2	Qual módulo tem sido modificado recentemente?
3	Qual módulo tem sofrido mais alterações?
4	Quando uma mudança no código fonte ocorreu?
5	Quem já modificou um dado módulo?
6	Quem foi que mais modificou um dado módulo?
7	Quanto trabalho determinada pessoa realizou em um dado módulo?
8	Onde determinada pessoa tem trabalhado?
9	Para um dado módulo, quais são as mudanças que o modificaram?
10	Qual foi o motivo por trás da modificação feita em determinado módulo?
11	Como foi o processo de mudança de um dado módulo?

Quanto à posição na empresa em que trabalham atualmente, existiam 3 professores, 3 analistas, 7 desenvolvedores e gerentes de software e 13 que não declararam o setor de trabalho. A Figura 2 apresenta estes dados[2].

Figura 2 – Perfil dos participantes do survey.



3. ESTUDO DE CASO

Após a aplicação do *survey*, foi realizado um estudo de caso com o objetivo de avaliar quais questões do survey o SME era capaz de responder. Foi utilizado como software objeto, o *Mobile Media*. Este estudo de caso foi desenvolvido pelos autores deste artigo.

4. REVISÃO DA LITERATURA SOBRE EXPERIMENTO CONTROLADO

O experimento controlado é o método de pesquisa explicativa em que há controle do objeto que está sendo estudado (Wohlin et al., 2000). Ao realizar este tipo de estudo é importante verificar cuidadosamente os efeitos causados. Em nosso contexto, o importante é perceber os benefícios do SME para a visualização de evolução de software através de uma coleta eficiente dos resultados do experimento. Através das leituras feitas nesta tarefa, foi possível entender que, com o método do experimento controlado, podemos avaliar práticas em engenharia de software. Esses experimentos são caros e exigem pessoas experientes para serem realizados. Assim como para outras metodologias de pesquisas, o planejamento é essencial para garantir a validade dos resultados do experimento. Devido ao alto risco e ao custo associado, é importante a presença de um grupo de pesquisadores experientes para a condução do experimento. Um mau planejamento do experimento pode conduzir a resultados errôneos e insatisfatórios.

5. CONTRUÇÃO DO TUTORIAL SME

Para apoiar a realização do experimento, foi decido realizar a construção do tutorial do *SourceMiner Evolution*. Este tutorial serviria como um documento ilustrativo para que os futuros usuários (participantes do experimento controlado) tenham maior facilidade na utilização da ferramenta.

6. PLANEJAMENTO DO EXPERIMENTO CONTROLADO

Antes de se realizar o experimento controlado é necessário fazer o seu planejamento detalhado. O objetivo do experimento controlado é avaliar a efetividade do SME na realização de determinadas atividades de evolução de software quando comparado com o uso de SVN[3] no Eclipse. Pretende-se rodar o experimento em 02 sessões, sendo 01 para cada tratamento (SME e SVN). O experimento controlado será rodado no IFBA Campus Santo Amaro com estudantes inseridos no curso da área de computação. As ferramentas serão instaladas e testadas antes da execução do experimento. O código fonte do objeto experimental estará disponível para os participantes. O experimento deverá ser concluído em 120 minutos. Este tempo poderá ser ajustado após a realização de um estudo piloto.

[2] Na Figura 2, DS significa Desenvolvedor de Software e GS significa Gerente de Software.
[3] **Apache Subversion**. É um sistema de controle de versão.

Inicialmente, os participantes do experimento deverão realizar um treinamento, no qual se familiarizarão com o ambiente SME e com o SVN, e aprenderão a utilizar a ferramenta com melhor desempenho no seu uso.

Procedimentos para a execução do experimento controlado:

1. **Instalação e configuração das ferramentas:** As ferramentas SME e SVN serão instaladas nos computadores do laboratório de informática do IFBA. Os participantes estarão distribuídos em duas equipes: uma para realizar atividade de evolução de software com a ferramenta SME e a outra, com a ferramenta SVN;

2. **Perfil dos participantes:** Os participantes preencherão um formulário de caracterização de perfil que serão alocados para um tratamento;

3. **Realização de Treinamento:** Será oferecido um treinamento de duas horas para os participantes do experimento controlado. O objetivo do treinamento é explanar e apresentar a utilização das ferramentas para a realização de atividade de evolução de software. Adicionalmente, também serão explicados os principais conceitos e a motivação por trás do estudo. Haverá uma pequena tarefa com os participantes observando como se utilizar as ferramentas durante o experimento para que quaisquer dúvidas sejam sanadas;

4. **Exercício do uso da ferramenta:** Um trabalho prático será enviado para cada participante para que eles possam praticar e se familiarizar mais com a ferramenta que eles irão trabalhar. Utilizaremos o *MobileMedia* (MM), como objeto experimental durante este exercício. As equipes serão divididas e cada uma receberá tarefas para realizarem sobre o MM. O manual do *SourceMine Evolution* será enviado para os participantes, com o objetivo de guiar eles durante a realização desta atividade. Todas as respostas serão enviadas por e-mail para os experimentadores, garantindo assim que os participantes realizem as tarefas propostas;

5. **Execução do experimento controlado:** Para testar se a nossa compreensão a partir do estudo de caso está correta, realizaremos um experimento controlado. O experimento será rodado no laboratório do IFBA Campus Santo Amaro, local onde todo o ambiente será montado. Neste experimento, os participantes receberão a lista de questões utilizada no *survey*. Essas questões deverão ser respondidas através do uso da ferramenta SME e a ferramenta SVN do Eclipse. O objetivo dessa etapa do projeto será analisar se as questões podem ou não ser respondidas através da utilização do *SourceMiner Evolution* e do SVN do Eclipse. Os dados coletados neste estudo serão analisados através de testes estatísticos. A partir disso, será feita uma avaliação do número de questões mais respondidas entre os participantes, assim identificaremos oportunidades de melhoria na infraestrutura.

6. **Resultados:** O resultado do experimento será descrito em um documento onde terão todas as informações a respeito do trabalho desenvolvido.

7. APRESENTAÇÃO DOS RESULTADOS

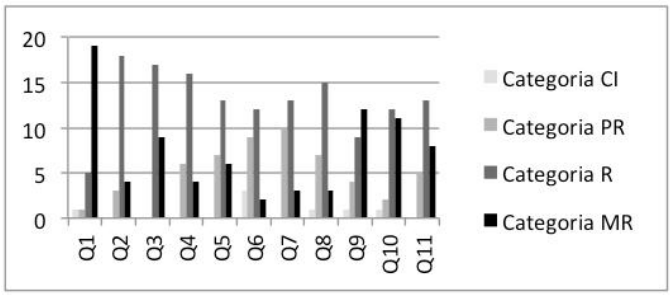
A seguir, estão descritos os resultados relacionados ao *survey* e ao estudo de caso. O experimento controlado ainda está em fase de execução, de tal forma que não é possível reportar neste artigo os seus resultados.

7.1 RESULTADOS DA AVALIAÇÃO DAS QUESTÕES DO SURVEY

A seguir, são apresentados os resultados das respostas das 11 questões por parte dos 26 participantes do *survey*. As 11 questões do *survey* foram respondidas seguindo uma categoria previamente definida. As pessoas poderiam respondê-las como (CI - Completamente Irrelevante, PR - Pouco Relevante, R - Relevante e MR - Muito Relevante). Foi perguntado aos participantes também qual era o principal profissional interessado em cada questão. Assim, o participante deveria responder se quem estaria interessado naquele tipo de questão era DS- Desenvolvedor de Software, GS- Gerente de Software, A- Ambos ou N- Nenhum.

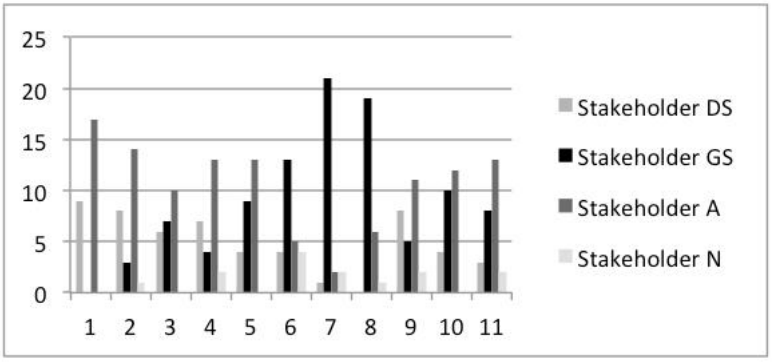
A Figura 3 apresenta o resultado da avaliação das questões quanto à categoria.

Figura 3 – Resultado das questões quanto à categoria.



A seguir, temos o resultado da avaliação das questões quanto aos principais interessados (*stakeholders*).

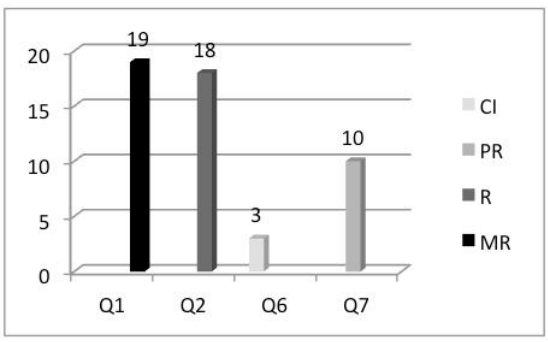
Figura 4 – Resultado das questões quanto aos principais interessados (*Stakeholder*).



O objetivo deste estudo foi saber dos gerentes e desenvolvedores de software suas opiniões em relação às perguntas coletadas. O *survey* nos permitiu saber o nível de utilidade das questões apresentadas para o desenvolvimento da pesquisa, isto é, identificou quais questões possuíam o nível de contribuição baixo, médio ou alto para a avaliação da ferramenta estudada e as pessoas relacionadas a cada tipo de pergunta.

Com o resultado do *survey*, percebeu-se que gerentes e desenvolvedores de software considerou a maioria das questões como Relevante e Muito Relevante. Foram analisadas também as questões que mais se destacaram em cada uma das categorias. A Figura 5 ilustra estes resultados. Desta forma, a Questão 1 foi considerada por 73,08% participantes como "Muito Relevante"; a Questão 2 foi considerada por 69,23% dos participantes como "Relevante"; a Questão 6 foi considerada por 11,54% dos participantes como "Completamente Irrelevante"; e a Questão 7 foi considerada por 38,46% dos participantes como "Pouco Relevante". A partir dessa análise, considera-se que as questões estão de forma geral nos níveis médio e alto de contribuição para a realização do experimento controlado.

Figura 5 – Resultado da avaliação das questões mais votadas quanto à categoria.



7.2 RESULTADOS DO ESTUDO DE CASO

As questões da Tabela 1 foram utilizadas para avaliar o SME analisando quais delas poderiam ser respondidas através do uso da ferramenta. Foram utilizadas cinco versões do software MobileMedia para a avaliação e foi percebido que o SME não responde a muitas questões, entre elas estão as consideradas as mais importantes para visualização de evolução de sistemas. A justificativa para isso é que as questões utilizadas para a avaliação estão relacionadas aos dados de repositórios de código fonte e não para análise de evolução de software entre versões utilizando o código fonte propriamente dito. Isto implica em dizer que o SME deve dar suporte a dados de repositórios de código fonte.

8. CONCLUSÃO

Este artigo apresenta o trabalho realizado em um projeto de pesquisa para avaliação experimental de um ambiente visual multiestratégia, chamado *SourceMiner Evolution* (SME). Como qualquer ambiente proposto, faz-se necessário avaliar o SME experimentalmente para medir sua efetividade na realização de atividades de evolução de software. Neste contexto, o projeto consistiu em planejar e executar estudos experimentais para avaliação do ambiente.

Foi planejada a realização de dois estudos experimentais para validar o SME. Foi planejado e executado um estudo de caso para avaliar a efetividade na realização de atividades de evolução de software usando o SME. Este estudo foi baseado em um conjunto de questões de evolução de software coletadas da literatura. Antes de aplicar o estudo, foi feito um survey, para ver como desenvolvedores e engenheiros de software concordavam com a real aplicabilidade destas questões. O *survey* mostrou que a maior parte das questões foram classificadas como Relevantes ou Muito Relevantes. Após esta etapa, foi realizado o estudo de caso, que consistiu em avaliar o SME quanto à realização ou não de cada uma das questões do *survey*. O estudo de caso mostrou que o SME não respondia a uma boa parte das questões do *survey*. Isto aconteceu porque as questões utilizadas para a avaliação estão relacionadas a dados de repositórios de código fonte, enquanto que o SME dá suporte à evolução de software analisando o código fonte de diferentes versões. O resultado do estudo de caso aponta para uma necessidade em evoluir o SME para suporte a dados de repositório de código fonte, aumentando assim seu escopo de atuação.

Por fim, foi planejado o experimento controlado. Este experimento está em fase de execução, de tal forma que os resultados deste estudo não serão apresentados neste artigo.

Os estudos realizados por este trabalho destacaram a importância da avaliação experimental incremental. Os estudos foram planejados incrementalmente para uma avaliação efetiva do SME. Pretende-se, como trabalhos futuros, realizar o experimento controlado para validar a ferramenta e avaliar as diferentes estratégias de análise disponíveis no ambiente.

9. REFERÊNCIAS

B. de Alwis and G. C. Murphy. **Answering Conceptual Queries with Ferret.** Dept of Computer Science. University of British Columbia. Vancouver, B.C., Canada. pag. 21-23, 25, 2008.

C. Chen. Information Visualization: **Beyond the Horizon.** Springer-Verlag New York, Inc., Secaucus, NJ, USA. 2006.

C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B., Regnell, and A. Wesslén, A. **Experimentation in software engineering: an introduction.** Kluwer Academic Publishers, Norwell, MA, USA. 2000

D. M. German and A. Hindley. **Visualizing the Evolution of Software Using Softchange.** Software Engineering Group, Department of Computer Science, University of Victoria, Victoria, BC, Canada V8W3P6, 2006.

J. Sillito, G. C. Murphy and K. De Volder. **Questions Programmers AskDuring Software Evolution Tasks.** Department of Computer Science. University of British Columbia. Vancouver, B.C. Canada, 2006.

L. Erlikh. **Leveraging legacy system dollars for e-business.** IEEE IT Pro, 2000, pp. 17–23.

L. Hattori, M. D'Ambros, M. Lanza, M. Lungu. **Software Evolution Comprehension: Replay to the Rescue.** Program Comprehension (ICPC), 2011 IEEE 19th International Conference on , vol., no., pp.161,170, 22-24 June 2011.

M. D'Ambros, M. Lanza. BugCrawler: **Visualizing Evolving Software Systems.** Software Maintenance and Reengineering, 2007. CSMR '07. 11th European Conference on , vol., no., pp.333,334, 21-23 March 2007.

M. D'Ambros, M. Lanza. **Visual software evolution reconstruction.** J. Softw. Maint. Evol. 21, 3 (May 2009), 217-232. 2009.

M. Lanza, R. Marinescu. O**bject-Oriented Metrics in Practice: Using Software Metrics to Characterize, Evaluate, and Improve the Design of Object-Oriented Systems.** Springer Publishing Company, Incorporated, 1st edition. 2010.

M. M. Lehman. Laws of Program Evolution - Rules and Tools for Programming Management. In Proc. Infotech State of the Art Conference, Why Software Projects Fail, April 9-11, 1978, pp. IV1-IV25.

M. M. Lehman. **Program Life Cycles and Laws of Software Evolution.** Proceedings of IEEE, Special Issue on Software Engineering, September, 1980, pp. 1060-1076.

M.-A. D. Storey, D. C˘ubranic´, D. M. and German. **On the use of visualization to support awareness of human activities in software development: a survey and a framework.** In Proceedings of the 2005 ACM symposium on Software visualization, SoftVis '05, pages 193–202, New York, NY, USA. ACM. D. M. 2005.

N. Zazworka, C. Ackermann. **CodeVizard: a tool to aid the analysis of software evolution.** In Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM '10). ACM, New York, NY, USA, , Article 63 , 1 pages. 2010.

P. Runeson, M. HOST. **Guidelines for conducting and reporting case study research in software engineering.** Empir Software Eng, Volume 14, Issue 2,Pages 131 – 164, 2009.

R. Fjeldstad, W. Hamlen. **Application program maintenance: Report to our respondents.** Tutorial on Software Maintenance, Parikh, G. & Zvegintzov, N. (Eds.). IEEE Computer Soc. Press. 1983, pp. 13–27.

R. L. Novais, A. Torres, T. S. Mendes, M. Mendonça, N. Zazworka. **Software evolution visualization: A systematic mapping study.** Information and Software Technology, Volume 55, Issue 11, November 2013, Pages 1860-1883, ISSN 0950-5849.

R. L. Novais, C. Lima, G. de F. Carneiro, P. R. M. S. Junior, and M. Mendonça. **An interactive differential and temporal approach to visually analyze software evolution.** In Visualizing Software for Understanding and Analysis (VISSOFT), 2011 6th IEEE International Workshop on, pages 1–4. 2011a.

R. L. Novais, C. Lima, G. de F. Carneiro, P. R. M. S. Junior, and M. Mendonça. **On the use of software visualization to analyze software evolution - an interactive differential approach.** In R. Zhang, J. Cordeiro, X. Li, Z. Zhang, and J. Zhang, editors, ICEIS (3), pages 15–24. SciTePress. 2011c.

R. L. Novais, C. Nunes, C. Lima, E. Cirilo, F. Dantas, A. Garcia, and M. Mendonca. **On the proactive and interactive visualization for feature evolution comprehension: An industrial investigation.** In Software Engineering (ICSE), 2012 34th International Conference on, pages 1044–1053. 2012a.

R. L. Novais, G. de F. Carneiro, P. R. M. S. Junior, and M. Mendonça. **On the use of software visualization to analyze software evolution: An interactive differential approach.** In R. Zhang, J. Zhang, Z. Zhang, J. Filipe, and J. Cordeiro, editors, Enterprise Information Systems, volume 102 of Lecture Notes in Business Information Processing, pages 241–255. Springer Berlin Heidelberg. 2012b.

R. L. Novais, M. Mendonça, **A multiple strategy software evolution visualization infrastructure.** Information and Software Technology. submitted. 2013.

R. L. Novais, M. Mendonça, D. Maron, I. Machado, C. Lima. **On the use of a multiple-visualization approach to manage software bugs.** In Software Visualization (WBVS), 2011 1nd Brazilian Workshop on, pages 1–8. 2011b.

R. L. Novais, P. R. M. S. Junior, and M. Mendonça. **Timeline matrix: an on demand view for software evolution analysis.** In Software Visualization (WBVS), 2012 2nd Brazilian Workshop on, pages 1–8. 2012c.

S. Chidamber, C. Kemerer. **A metrics suite for object oriented design.** Software Engineering. IEEE Transactions on, 20(6), 476–493. 1994.

S. Diehl. Software Visualization: **Visualizing the Structure, Behaviour, and Evolution of Software.** Springer-Verlag New York, Inc., Secaucus, NJ, USA. 2007.

T. Fritz and G. C. Murphy. **Using Information Fragments to Answer the Questions Developers Ask.** Department of Computer Science, University of British Columbia. Vancouver, BC, Canada, 2010.

Tu Qiang, M.W. Godfrey. **An integrated approach for studying architectural evolution.** Program Comprehension. Proceedings. 10th International Workshop on , vol., no., pp.127,136, 2002.

10. AGRADECIMENTOS

Agradecemos ao IFBA (http://www.ifba.edu.br) e a FAPESB (http://www.fapesb.ba.gov.br/) pela bolsa de iniciação científica fornecida para a realização do projeto de iniciação científica da aluna Creidiane Brito. Agradecemos também ao grupo de pesquisa GIA – Grupo de Informática Aplicada (http://www.gia.ifba.edu.br), por fornecer a estrutura física para realização do projeto.