Um Mapeamento Sistemático Sobre o Uso de Metodologias Ágeis no Processo de Experimentação Científica

Augusto Consulmagnos Romeiro, Daniel de Oliveira

Instituto de Computação - Universidade Federal Fluminense (UFF), Brasil

{augusto.romeiro,danielcmo}@ic.uff.br

Resumo. *Muitos experimentos científicos são baseados em simulações computacionais. Assim, os cientistas de diversas áreas, como a biologia e a química, desenvolvem softwares como parte fundamental de sua pesquisa, seja diretamente escrevendo programas ou desenvolvendo novos workflows científicos. Assim como na área comercial, todo o desenvolvimento em projetos científicos deve ser estruturado, documentado e realizado de forma a reduzir o tempo de desenvolvimento e os custos envolvidos. No processo de experimentação científica os requisitos são instáveis e podem mudar constantemente durante a execução do projeto, por isso, as metodologias ágeis já consagradas para o desenvolvimento de software comercial se mostram interessantes. O propósito desse artigo é mapear sistematicamente as abordagens existentes que aplicam metodologias ágeis no contexto científico de forma que possamos fornecer uma visão geral da aplicação de tais metodologias no ambiente científico e apresentar as lacunas e oportunidades para melhorias neste domínio.*

1. Introdução

É de conhecimento comum que o desenvolvimento de *software* é uma atividade complexa e que envolve muitos riscos, principalmente no que tange o orçamento, o consumo de tempo e a qualidade do produto final [Pressman e Maxim 2014]. De forma a reduzir esses riscos associados ao desenvolvimento de *software* foram desenvolvidas metodologias mais tradicionais como o desenvolvimento em cascata e o RUP [Pressman e Maxim 2014]. No desenvolvimento em cascata, o processo de desenvolvimento do *software* segue uma sequência de fases bem definidas, sendo que cada fase depende da conclusão da fase anterior para ser iniciada. Já no RUP o processo é iterativo por natureza e incremental, porém ainda é linear em grande parte já que possui "mini-cascatas" dentro de cada iteração. O problema dessas metodologias é que elas assumem certos fatos que não se aplicam ao desenvolvimento de *software* real [Teles 2004].

Para suprir essa lacuna foram propostas as metodologias ágeis [Teles 2004]. Nas metodologias ágeis, o desenvolvimento do *software* segue o modelo iterativo e em espiral onde todas as fases do modelo em cascata são executadas *n* vezes ao longo do desenvolvimento (*n* ciclos), onde ao final de cada ciclo temos um produto funcional (e não um protótipo). Ao final do último ciclo o *software* completo está pronto para ser colocado em produção. As metodologias ágeis possuem várias vantagens como por exemplo: o cliente/equipe de desenvolvimento aprende ao longo do desenvolvimento, uma vez que já pode utilizar o sistema, o tempo de "espera" do cliente por um produto é reduzido *etc*. O uso de metodologias ágeis se mostrou interessante para o desenvolvimento de *software* já que o mesmo é uma tarefa não-linear onde tradicionalmente ocorrem implementações, correções de erro e refatorações sucessivas,

já que o aprendizado adquirido durante o desenvolvimento torna possível aos desenvolvedores perceberem maneiras mais eficientes de desenvolverem o *software*.

Entretanto, não é somente na área comercial que *softwares* são desenvolvidos e cujo trabalho também é não-linear. A área científica tem se tornado cada vez mais dependente do desenvolvimento de abordagens computacionais, principalmente em áreas que desenvolvem experimentos *in silico* [Mattoso *et al.* 2010, Travassos e Barros 2003]. Muitos dos cientistas de diversas áreas, como a biologia e a química, também desenvolvem *softwares* como parte fundamental de sua pesquisa, seja diretamente escrevendo programas ou criando novos *softwares* por meio da composição de *softwares* existentes (*e.g.,* na forma de um *workflow* científico). Tal como os desenvolvedores comerciais, o código desenvolvido (ou o *workflow* modelado), a execução do mesmo bem como a análise dos resultados precisam ser compreensíveis, passíveis de reprodução, livres de erros e bem documentados. Assim, é intuitivo que pensemos que as metodologias ágeis também possam ser aplicadas no domínio científico.

Entretanto, quando lidamos com experimentos científicos algumas características o diferem do desenvolvimento de *software* tradicional. Por exemplo, o desenvolvimento do *software* científico é apenas uma parte do ciclo de vida do experimento científico definido por Mattoso *et al.* (2010). Sua execução com um conjunto de parâmetros (a execução do experimento propriamente dita) e a análise dos resultados gerados são também "entregáveis" do processo e devem ser considerados pela metodologia. Alguns trabalhos já foram propostos na literatura para a aplicação de metodologias ágeis no processo de experimentação científica. Entretanto, cada um desses trabalhos cobre somente parte do ciclo de vida do experimento, fazendo com que tenhamos dificuldade em visualizar o cenário completo. Assim, faz-se necessária a execução de um mapeamento sistemático como os executados pela área de Engenharia de *Software* para que se realize um levantamento criterioso das abordagens existentes. Este artigo tem, então, como objetivos: (i) fornecer uma visão da aplicação de metodologias ágeis no processo científico; e (ii) apresentar lacunas e oportunidades para melhorias neste domínio.

Este artigo está organizado em 3 seções além da introdução. Na Seção 2 apresentamos uma descrição de metodologias ágeis e listamos as práticas mais comuns nessas metodologias que servirão para balizar a comparação entre as abordagens. Na Seção 3 apresentamos o planejamento, a execução e a análise do mapeamento sistemático da literatura, e, finalmente, na Seção 4 concluímos este artigo.

2. Metodologias Ágeis e suas Principais Práticas

A proposta principal das metodologias ágeis é fazer com que as pessoas desempenhem um papel fundamental no desenvolvimento de *software*, sendo para isso essencial que em todos os times de desenvolvimento ocorra uma boa comunicação entre os desenvolvedores. Além disso, é primordial que cada desenvolvedor esteja comprometido com a qualidade do *software* a ser desenvolvido. Nessas metodologias é valorizada a entrega de um produto funcional e adequado ao que o cliente realmente deseja. Existem diversas metodologias ágeis propostas como o XP, o SCRUM, o Crystal e o *Dynamic Systems Development Method* (DSDM) [Teles 2004]. Embora cada um tenha uma característica própria no que tange suas práticas, eles compartilham inúmeras características em comum, incluindo o desenvolvimento iterativo, e um foco

na comunicação interativa e na redução do esforço empregado em artefatos intermediários [Cohen *et al.* 2004], que podem ser aproveitadas no ambiente científico.

De forma a comparar as abordagens existentes que aplicam metodologias ágeis no ambiente científico, adaptamos a listagem de práticas proposta por [Sletholt *et al.* 2012] que são comuns a todas as metodologias e que podem ser aplicadas em projetos de experimentação científica e as apresentamos a seguir: (1) Existe um papel dedicado para priorizar as funcionalidades, (2) Existe um papel de um facilitador do processo de experimentação/desenvolvimento, (3) Existe uma reunião de planejamento do ciclo para criar uma lista de tarefas, (4) É feito um planejamento para estimar as tarefas durante o planejamento do ciclo, (5) Existe um ciclo com tempo definido e produzindo um incremento do produto/entregável, (6) Existe compromisso mútuo das tarefas do ciclo entre o coordenador e o time de desenvolvimento, (7) É feita uma curta reunião diária para resolver os problemas atuais, (8) Os membros da equipe se voluntariariam para fazer as tarefas, (9) É feito um gráfico de *Burndown* para monitorar o progresso das tarefas do ciclo, (10) É feita uma reunião para revisar o trabalho que foi completado, (11) É feita uma reunião para aprender com o ciclo anterior, (12) É realizado um planejamento para gerar incrementos do produto/entregável, (13) São escritas *user stories*, (14) O time tem um espaço aberto para trabalho, (15) É definido um ritmo de trabalho, (16) A velocidade do projeto deve poder ser medida, (17) As pessoas são conectadas para não haver perda de conhecimento, (18) O coordenador deve estar sempre disponível, (19) O desenvolvimento de programas/métodos e a execução do experimento deve ser padronizada, (20) Os testes devem ser desenvolvidos, (21) Toda produção de programas/*workflows* deve ser feita em pares (*pair programming*), (22) Cada par deve realizar a integração dos produtos desenvolvidos de cada vez, (23) Deve ser feita a integração contínua, (24) Deve-se configurar um computador/ambiente para realizar a integração, (25) É utilizada a propriedade coletiva, (26) É fomentado o uso de arquitetura simples, (27) Escolha metáforas para o projeto, (28) São usadas sessões de *brainstorm* para criar arquiteturas, (29) São usados testes de conceito para reduzir os riscos de uma solução (*Spike Solutions*), (30) Nenhuma funcionalidade é adicionada no início do projeto sem avaliação da equipe, (31) É feita a refatoração sempre que possível, (32) Todo código deve estar coberto por testes, (33) Todos os testes unitários devem passar antes de ser entregue um incremento, (34) Quando é achado um *bug* é criado um teste para ele, e (35) Testes de aceitação são executados com frequência e a pontuação é publicada. Uma vez definidas as práticas, podemos então comparar as abordagens existentes que aplicam metodologias ágeis no processo de experimentação científica.

3. Aplicação do Mapeamento Sistemático Proposto

O mapeamento sistemático é uma das formas de conseguirmos diferenciar, determinar e explicar conteúdos envolvendo uma área de pesquisa tentando responder a uma questão [Kitchenham 2004]. A ideia do mapeamento sistemático é apresentar uma forma de avaliação que seja justa envolvendo um tópico de pesquisa, onde a revisão deve ser confiável e caso necessário, deve haver a possibilidade de se realizar uma auditoria nos resultados. Então, para isto um mapeamento sistemático contém um protocolo de como o mesmo foi feito para poder permitir que possa ser repetido por outros pesquisadores. Os estudos individuais que são analisados em um mapeamento sistemático são chamados de estudos primários. O mapeamento em si é um estudo secundário.

Como previsto em [Kitchenham 2004] devemos executar três fases principais: o planejamento, a execução e a análise dos resultados. Na fase de planejamento é definido o protocolo que iremos seguir. Para isto devemos ter um objetivo bem definido em nossa pesquisa. Dessa forma, elaboramos duas perguntas que esta pesquisa deve responder após a análise dos resultados: (i) Existe alguma abordagem que aplique métodos ágeis no processo de experimentação científica? e (ii) Quais práticas ágeis estão envolvidas nestes estudos? Assim, nossa estratégia de busca consistiu em identificar abordagens relacionadas aos conceitos de *e-Science* e métodos ágeis. Para tal, definimos a seguinte *string* de busca:

(e-science OR eScience OR "scientific software") AND (agile development OR agile methods OR agile practices OR agile software development OR Scrum OR Agile Software)

Conectamos os termos principais usando o operador lógico AND e as variações possíveis, usando o operador lógico OR. Para executar a *string* apresentada, consideramos os motores de busca de artigos científicos apresentados na Tabela 1. Esses motores de busca foram selecionados pois atendem aos critérios previamente definidos. Foram considerados motores de busca onde suas publicações são atualizadas com regularidade, o texto se encontra disponível para download e análise e os textos são revisados de acordo com o processo de *peer-review*.

Tabela 1 Motores de Busca utilizados no mapeamento sistemático

Bases de Dados	URL
ACM	`http://dl.acm.org`
IEEExplore	`http://ieeexplore.ieee.org`
Scopus	`http://www.scopus.com`
Google Acadêmico	`https://scholar.google.com.br`

No caso da busca no Google Acadêmico, traduzimos os termos da *string* de busca para que resultados em Português também fossem retornados. Muitos dos artigos encontrados ao efetuar a busca eram irrelevantes para esta pesquisa, então definimos dois critérios para incluir o artigo em nossa pesquisa. Os critérios para a inclusão foram que o estudo do artigo tivesse o foco em métodos ágeis e *e-Science* e caso dois artigos apresentassem o mesmo tema, consideraríamos o mais recente. Criamos também alguns critérios de exclusão de artigos: os artigos devem estar disponíveis na internet, os artigos devem estar em formato eletrônico, os artigos devem estar escritos em inglês ou português e os artigos devem contemplar a execução de estudos experimentais utilizando metodologias ágeis em projetos científicos.

O mapeamento sistemático foi realizado entre Março e Abril de 2015. Na Tabela 2 apresentamos a quantidade total de resultados obtidos utilizando a máquina de busca e a quantidade de artigos considerados após a filtragem realizada pelo título e resumo. Dentre os 56 artigos filtrados após a leitura do título e resumo, selecionamos somente 8 artigos como relevantes para esta pesquisa de acordo com o conteúdo do artigo. A seguir discutimos cada um desses artigos de forma a compararmos as abordagens existentes.

Tabela 2 Resultado das buscas nas máquinas de busca de artigos científicos

Mecanismos de busca	Resultados	Resultados com filtro
ACM	272	5
IEEExplore	11	5
Scopus	21	15
Google Acadêmico	2540	31
Total	2844	56

A Tabela 3 apresenta a lista de projetos descritos nesses estudos experimentais primários. Para cada um dos projetos listamos quais práticas o mesmo atende (S) ou não (*N*). Nos casos em que não há informação no artigo, representamos com "-". Nanthaamornphong *et al.* (2013) apresentam um estudo de caso que investiga o nível de apoio que os métodos ágeis fornecem para o desenvolvimento de *software* científico. O estudo de caso investiga o desenvolvimento de *software* para o projeto CLiiME (*Community Laser-induced Incandescence Modeling Environment*).

As conclusões de Nanthaamornphong *et al.* são as seguintes: (i) práticas ágeis fornecem métodos para enfrentar os riscos no início do ciclo de vida do experimento, (ii) os métodos ágeis parecem ser adequados para projetos de *software* científicos, (iii) os métodos ágeis ajudam desenvolvedores a evitar problemas relacionados com a qualidade de *software* dirigindo o desenvolvimento com nível suficiente de qualidade. O projeto apresentado nesse artigo é identificado como o Projeto 1 na Tabela 3.

Kane *et al.* (2006) apresentam experiências de múltiplos projetos onde métodos ágeis foram aplicados no cenário científico. Um total de seis projetos, todos incorporando práticas ágeis, são examinados pelos autores. Os métodos ágeis foram consideradas muito adequados para esse tipo de projeto de desenvolvimento de *software*. Nesta linha de desenvolvimento de *software* biomédico, o *software* tem de ser receptivo a mudanças, tanto as mudanças no domínio científico quanto as mudanças específicas feitas pelo coordenador do projeto. Resultados e características similares foram relatados em todos os projetos analisados, que fortalecem a o uso de métodos ágeis como por exemplo: todos os projetos tinham times pequenos (uma equipe com 2 a 5 membros). Os projetos discutidos nesse artigo são referenciados como os Projetos de 2 a 7 na Tabela 3.

Pitt-Francis *et al.* (2009) apresentam como métodos ágeis foram utilizados para o desenvolvimento de uma biblioteca para aplicações biomédicas. Os autores utilizaram a programação em pares (*i.e. pair programming*) e integração contínua, além da frequente refatoração do código desenvolvido. Porém, os autores não justificam porque aplicam determinadas práticas em seu projeto de cunho biológico. O projeto chamado Chaste é feito de forma incremental mas não é descrita a forma da organização dos times nem como é feito o planejamento do projeto. O projeto apresentado neste artigo é referenciado como Projeto 8 na Tabela 3.

Easterbrook e Johns (2009) apresentam um estudo de caso que mostra o desenvolvimento de aplicações cientificas pelos pesquisadores do *Met Office Hadley Centre* e que são baseadas em métodos ágeis. O artigo reforça que no processo utilizado a ênfase foi dada a verificação e validação das atividades, porém os autores não definem

quais dos métodos ágeis foram efetivamente utilizados, somente algumas práticas. O projeto apresentado nesse artigo na Tabela 3 é o Projeto número 9.

Wood e Kleb (2003) apresentam uma tentativa de inclusão de práticas do XP em um projeto de pesquisa da NASA. O projeto em que o XP foi aplicado possuía uma equipe reduzida (2-3 pessoas) e foi concluído em apenas 12 semanas. Das práticas existentes para o XP, a única que não foi utilizada por não haver viabilidade foi a programação em pares. O projeto apresentado nesse artigo na Tabela 3 é o Projeto de número 10. Kane (2003), Maxville (2009) e Ackroyd *et al.* (2008) discutem a aplicação do SCRUM em projetos científicos. Os três artigos escolheram o SCRUM devido à necessidade de serem mais sensíveis às mudanças de requisitos das aplicações. Todos os autores relatam experiências positivas com a implementação ágil como a prática incremental, com foco em testes e requisitos. Os projetos apresentados nestes artigos na Tabela 3 são os projetos 11, 12 e 13, respectivamente.

Tabela 3 Comparação dos estudos primários selecionados

Prática	*Projetos*									
	1	3	4	2,5,6,7	8	9	10	11	12	13
1	-	-	-	-	-	N	N	-	-	S
2,4	-	-	-	-	-	N	N	-	-	-
3	S	-	-	-	-	-	N	-	S	S
5	S	S	S	S	S	-	S	S	S	S
6	-	-	-	-	-	N	N	-	S	-
7	N	-	-	S	S	N	S	S	S	N
8	-	S	N	S	-	S	S	-	-	-
9	-	-	-	-	S	N	S	-	-	-
10	-	-	-	-	-	-	N	-	-	-
11	-	-	-	-	-	-	S	-	-	-
12,23	S	S	S	S	S	S	S	S	S	-
13	S	S	S	S	S	N	-	S	-	-
14	N	S	S	S	-	S	S	-	-	-
15,22,27	N	-	-	-	-	-	S	-	-	-
16	S	S	S	S	S	N	S	S	-	-
17	N	-	-	-	S	N	-	S	-	-
18	S	S	S	S	S	S	S	S	-	-
19	S	-	-	-	-	S	S	S	-	S
20,34	S	-	-	-	S	-	-	-	-	-
21	N	N	N	N	N	-	S	-	-	N
24	S	-	-	-	-	-	-	-	S	-
25	S	S	-	S	S	S	S	S	-	N
26	S	-	-	-	-	-	S	-	-	N
28	N	-	-	-	-	-	-	-	-	-
29,33	S	-	-	-	-	-	-	-	-	-
30	S	-	-	-	-	S	-	-	-	-
31	S	S	S	S	S	-	S	S	-	S
32	S	-	-	-	S	-	S	-	-	-
35	S	-	-	-	S	S	-	-	-	-

Apesar de todos os projetos apresentados descreverem vantagens no uso de métodos ágeis, esses projetos focaram somente no desenvolvimento do *software* científico, sem englobar outras etapas da experimentação científica como a execução do experimento e a análise dos dados obtidos. Por exemplo, no projeto 1, os métodos ágeis

foram empregados somente até a entrega do *software* do CLiiME. Entretanto, o processo de investigação científica continuou com a execução do referido *software* e a análise dos resultados. Essas duas etapas não são consideradas em nenhum dos trabalhos, evidenciando que existe uma lacuna a ser preenchida na aplicação de metodologias ágeis no ambiente científico. Acreditamos que as metodologias ágeis podem ser bastante úteis também para apoiar o planejamento das execuções dos experimentos (escolhas de valores de parâmetros, de *datasets* de entrada *etc*.) e também no planejamento de sua análise (como os times vão dividir as tarefas de análise de resultados, como chegarão a conclusões conjuntas *etc.*), já que em ambas as atividades existe todo um processo (muitas vezes rígido) envolvido.

4. Conclusões

O processo de experimentação científica evoluiu muito nos últimos anos. Muitos dos projetos científicos atuais envolvem a execução (muitas vezes em paralelo) de complexas simulações computacionais em ambientes distribuídos. Tais simulações são desempenhadas por um programa ou uma série de programas que são desenvolvidos ou adaptados no contexto desses projetos. Assim, se tornou uma prática comum que os cientistas de diversas áreas também desenvolvam e excutem *softwares* como parte fundamental de sua pesquisa. Assim, um projeto científico possui uma etapa clara de desenvolvimento do *software* a ser utilizado, uma etapa de execução desse *software* e outra etapa de análise de dados conforme explicitado por Mattoso *et al.* (2010). Esse processo deve ser estruturado, passível de reprodução e principalmente otimizado, para que se poupem recursos e o mesmo termine no tempo planejado. Assim, metodologias se fazem necessárias. Como o processo envolve o desenvolvimento de *software* científico, podemos nos beneficiar de metodologias já empregadas na engenharia de *software*, bastando adaptar tais metodologias para o novo contexto. Como os requisitos em projetos científicos são instáveis, as metodologias ágeis são intuitivamente as mais recomendadas. Porém, não é claro como tais metodologias são/já foram aplicadas em cenários científicos. Este artigo tem como objetivo realizar um mapeamento sistemático da literatura para que possamos fornecer uma visão da aplicação de metodologias ágeis no processo científico e também apresentar lacunas e oportunidades para melhorias neste domínio. Ao fim do mapeamento foram selecionados 13 projetos descritos em 8 artigos científicos. Tais projetos apresentam estudos de caso interessantes tanto da aplicação do XP quanto do SCRUM, porém todos eles se limitam a etapa do desenvolvimento do *software*, sem se preocupar com as etapas de execução e análise do experimento. Acreditamos que as metodologias ágeis podem auxiliar os cientistas nessas etapas também e se fazem muito importantes para que todo o processo se torne estruturado.

Agradecimentos. Os autores gostariam de agradecer ao CNPq (478878/2013-3), a FAPERJ (E-26/111.370/2013) e a CAPES pelo financiamento parcial deste trabalho.

Referências Bibliográficas

[Ackroyd *et al.* 2008] Ackroyd, K. S., Kinder, S. H., Mant, G. R., Miller, M. C., Ramsdale, C. A., and Stephenson, P. C. (2008). Scientific *software* development at a research facility. IEEE *software*, 25(4):44–51.

[Cohen *et al.* 2004] Cohen, D., Lindvall, M., and Costa, P. (2004). An introduction to agile methods. Advances in computers, 62:1–66.

[Easterbrook e Johns 2009] Easterbrook, S. M. and Johns, T. C. (2009). Engineering the *software* for understanding climate change. Computing in science & engineering, 11(6):64–74.

[Kane 2003] Kane, D. (2003). Introducing agile development into bioinformatics: an experience report. In Agile Development Conference, 2003. ADC 2003. Proceedings of the, pages 132–139. IEEE.

[Kane *et al.* 2006] Kane, D. W., Hohman, M. M., Cerami, E. G., McCormick, M. W., Kuhlmman, K. F., and Byrd, J. A. (2006). Agile methods in biomedical *software* development: a multi-site experience report. BMC Bioinformatics, 7(1):273.

[Kitchenham 2004] Kitchenham, B. (2004). Procedures for performing systematic reviews. Keele, UK, Keele University, 33(2004):1–26.

[Mattoso *et al.* 2010] M. Mattoso, C. Werner, G. H. Travassos, V. Braganholo, E. Ogasawara, D. D. Oliveira, S. M. S. D. Cruz, W. Martinho, and L. Murta, "Towards supporting the life cycle of large scale scientific experiments," International Journal of Business Process Integration and Management, vol. 5, no. 1, p. 79, 2010.

[Maxville 2009] Maxville, V. (2009). Preparing scientists for scalable *software* develop- ment. In *Software* Engineering for Computational Science and Engineering, 2009. SECSE'09. ICSE Workshop on, pages 80–85. IEEE.

[Nanthaamornphong *et al.* 2013] Nanthaamornphong, A., Morris, K., Rouson, D. W., and Michelsen, H. A. (2013). A case study: Agile development in the community laser- induced incandescence modeling environment (cliime). In Proceedings of the 5th In- ternational Workshop on *Software* Engineering for Computational Science and Engi- neering, pages 9–18. IEEE Press.

[Pitt-Francis *et al.* 2009] Pitt-Francis, J., Pathmanathan, P., Bernabeu, M. O., Bordas, R., Cooper, J., Fletcher, A. G., Mirams, G. R., Murray, P., Osborne, J. M., Walter, A., *et al.* (2009). Chaste: a test-driven approach to *software* development for biological modelling. Computer Physics Communications, 180(12):2452–2471.

[Pressman e Maxim 2014] Pressman, R., Maxim, B (2014) *Software* Engineering: A Practitioner's Approach. McGraw-Hill Science/Engineering/Math; 8 edition (January 23, 2014)

[Sletholt *et al.* 2012] Sletholt, M. T., Hannay, J. E., Pfahl, D., and Langtangen, H. P. (2012). What do we know about scientific *software* development's agile practices? Computing in Science & Engineering, 14(2):24–37.

[Teles 2004] Teles, V. Extreme *Programming*: Aprenda como encantar seus usuários desenvolvendo *software* com agilidade e alta qualidade. Novatec. ISBN: 85-7522-047-0;

[Travassos e Barros 2003] G. H. Travassos and M. O. Barros, "Contributions of In Virtuo and In Silico Experiments for the Future of Empirical Studies in *Software* Engineering," in 2nd Workshop on Empirical *Software* Engineering the Future of Empirical Studies in *Software* Engineering, Rome, Italy, 2003, pp. 117–130

[Wood e Kleb 2003] Wood, W. A. and Kleb, W. L. (2003). Exploring xp for scientific research. *Software*, IEEE, 20(3):30–36.