Usando Dojos de Programação para o Ensino de Desenvolvimento Dirigido por Testes

Ramiro Batista Luz[1,2], **Adolfo Neto**[2]

[1]Departamento Acadêmico de Informática
Universidade Tecnológica Federal do Paraná (UTFPR)
Av. Sete de Setembro, 3165
Curitiba - Paraná - Brasil - CEP: 80230-901

[2]Câmara Municipal de Curitiba
Rua Barão do Rio Branco, 720
Curitiba - Paraná - Brasil - CEP: 80010-902

ramiroluz@gmail.com, adolfo@utfpr.edu.br

Abstract. *Coding dojo is a dynamic and collaborative activity inspired em martial arts that follows a discipline in a cheerful and pleasant environment. Agile development techniques are utilized during the coding dojo, as test driven development, pair programming and baby steps, explained later in this article. This article presents coding dojo characteristics that help the agile development technique's teaching for heterogeneous groups.*

Resumo. *Dojo de programação é uma atividade dinâmica e colaborativa inspirada em artes marciais que segue uma disciplina num ambiente de ensino agradável e divertido. Técnicas de desenvolvimento ágil são utilizadas durante o dojo de programação, dentre elas o desenvolvimento guiado por testes, programação em pares, passos de bebê que serão explicadas neste artigo. Apresentamos nesse artigo características do dojo de programação que favorecem o ensino de técnicas de desenvolvimento ágil para grupos heterogênicos.*

1. Introdução

A palavra dojo, em uma tradução literal do japonês, significa ”lugar do caminho”, originalmente usada para designar o espaço de meditação dos monges budistas. Uma tradução livre significa ”lugar onde se estuda a vida”, mas hoje em dia o termo é mais conhecido como lugar onde se pratica artes marciais, uma sala dentro de uma academia com o chão coberto por tapetes.

Dojo de programação, em inglês *Coding Dojo*, é uma atividade que vem sendo usada por programadores para praticar métodos ágeis, especialmente desenvolvimento dirigido por testes [Beck and Andres 2004], programação pareada [Williams and Kessler 2002] e passos de bebê [Beck and Andres 2004]. Durante a atividade de *dojo* de programação [Sato et al. 2008b], é possível aprimorar diversas habilidades necessárias para o dia a dia de equipes que utilizam metodologias ágeis como, o desenvolvimento dirigido por testes, a comunicação de ideias reforçada pelas regras do

dojo de programação e essencial no dia a dia de times que adotaram métodos ágeis e a colaboração com o time buscando um objetivo comum característica marcante da atividade.

Até o momento pudemos perceber que dojo de programação é uma atividade dinâmica e que favorece o aprendizado, incluindo e incentivando os alunos a participar do processo de aprendizado de forma colaborativa [Bossavit and Gaillot 2005] e [Sato et al. 2008a].

Acreditamos que a atividade potencializa o aprendizado de programação pelo aspecto prático que programar computadores requer e pelo exemplo. É comum em turmas da disciplina de programação de computadores algum aluno que não consegue dar os primeiros passos, eventualmente o que falta é apenas um detalhe, num dojo de programação este aluno observa outros alunos fazendo e percebe esse detalhe, em seguida ele mesmo irá praticar e em caso de dúvida conta com a ajuda dos companheiros da plateia.

O desenvolvimento dirigido por testes, em inglês *Test Driven Development* normalmente identificado pela sigla TDD, é um conjunto de técnicas que encoraja projetos simples e conjuntos de testes que inspiram confiança [Beck 2002]. Duas regras básicas devem ser seguidas, primeiro deve-se escrever um teste automatizado que falhe antes de escrever algum código e a outra regra diz que deve-se remover código duplicado. Essas regras geram o simbolismo do TDD que possui três termos, Vermelho, Verde e Refatoração, chamado de mantra do TDD [Beck 2002]. O significado é o seguinte, vermelho - escreva um pequeno teste que falhe, verde - faça o teste funcionar rapidamente cometendo qualquer pecado necessário nesse processo e refatoração - remova a duplicação criada para fazer o teste passar. As vantagens de usar esse método são *feedback* constante da situação do código, confiança para mudar pois as funcionalidades prontas possuem testes, caso uma alteração afete algo pronto o teste irá verificar se aquilo que estava pronto continua funcionando e há um limite indicando que algo está pronto, quando os testes estiverem passando não há necessidade de continuar desenvolvendo [Beck 2002].

A programação em par é uma técnica caracterizada por uma dupla sentada lado a lado compartilhando o mesmo computador e desenvolvendo a solução colaborativamente. É um diálogo e programação simultânea entre duas pessoas [Beck and Andres 2004]. Essa técnica permite que os programadores mantenham a atenção de um e do outro na tarefa, refinem as ideias do sistema, esclarecimento das ideias, quando alguém não consegue seguir em frente o outro toma iniciativa e ambos se mantem alinhados com as práticas do time [Beck and Andres 2004].

Durante o desenvolvimento há uma tentação de fazer grandes mudanças, entretanto, fazer mudanças grandes de uma só vez é perigoso. O impacto de fazer pequenos passos é muito menor do que a perda causada pelo recuo da equipe ao abortar grandes mudanças. Os passos de bebê, *Baby Steps* em inglês, é o nome dado à técnica de fazer pequenas mudanças na direção da solução do problema [Beck and Andres 2004].

Para realização de *dojos* de programação é necessário um projetor, um computador e um quadro para escrever ou desenhar algo relacionado à atividade, além dos participantes e cadeiras para todos ficarem confortáveis. Opcionalmente é feito um lanche após a atividade, oportunizando um momento de socialização e descontração onde discussões podem ser continuadas.

De forma simplificada pode-se dizer que durante um encontro de *dojo* de programação, programadores se reúnem para resolver desafios de programação. Os desafios devem ser resolvidos utilizando técnicas de métodos ágeis. Em um único computador os participantes se revezam em pares, usando TDD e passos de bebê, para resolver um desafio, a cada turno, que pode ser um limite de tempo ou outra regra como por exemplo, sempre que um teste passar, o participante que está usando o teclado vai para a plateia, o participante que estava ao lado assume o teclado e uma pessoa da plateia senta ao lado para formar o par. A dupla deve tomar o cuidado de explicar o que está sendo feito para a plateia. A tela do computador é projetada para que todos vejam o que está acontecendo.

2. Estado da Arte

Nos últimos anos alguns estudos foram realizados para avaliar a qualidade de *software* produzido com metodologias ágeis [Vodde and Koskela 2007], [Siniaalto and Abrahamsson 2007], [Janzen and Saiedian 2008] e [Pancur et al. 2003]. Outros avaliaram a aderência à pratica [Bravo 2010] e até aspectos subjetivos como relacionamento aluno-professor [Dubinski 2003]. Dentre eles [Bravo 2010], [Bravo 2011] utilizaram *dojos* de programação para treinar os participantes de uma pesquisa. A forma de avaliação utilizada foram questionários e ferramentas desenvolvidas para registrar estatísticas relacionadas com a metodologia e que auxiliam a aplicação da prática com lembretes para troca de pares. Segundo Mariana Bravo [Bravo 2011], existem indícios de que dojo de programação é uma boa técnica para ensino de práticas ágeis, especialmente para aqueles que não a conhecem.

No Brasil temos diversos grupos que se encontram regularmente para prática de *dojos* de programação[1]. Um dos grupos que mais se destaca é o grupo DojoSP, que se encontra na USP. Integrantes do grupo escreveram um artigo [Sato et al. 2008b] sobre o ambiente de aprendizado dos *dojos* de programação que eles participam. Chama a atenção o item 4 "*Dojo and Learning*", que relaciona o aprendizado e a prática de atividades esportivas e artísticas com *dojos* de programação.

Em [Wellington 2005] e [Dubinski 2003] foram avaliados estudantes de graduação durante o aprendizado de TDD mas sem a utilização da técnica de *dojos* de programação. Em [Bravo 2011] foi feita uma avaliação de percepção de aprendizado de TDD onde foi aplicada a técnica de *dojos* de programação. Os participantes responderam questionários *on-line* sobre a percepção de aprendizado voluntariamente. Foram encontrados indícios de que a prática é uma boa técnica de ensino aos que pouco conhecem ou

[1]Existe uma aplicação que mapeia esses grupos, apesar de estar um pouco desatualizada ela mostra a distribuição geográfica de grupos, `http://dojomap.heroku.com/` com código fonte disponível em `https://github.com/horaextra/dojomap`.

tem pouca experiência com TDD, não ficando claro o efeito sobre participantes mais experientes. Foi apontada como ameaça à pesquisa o fato dos participantes participarem de encontros de *dojos* de programação voluntariamente, sendo assim, predispostos a treinar as práticas propostas.

Atualmente existem muitos encontros no Brasil para prática de *dojos* de programação. Alguns ocorrem em universidades outros em empresas. A prática de *dojos* de programação também vem sendo explorada por empresas, como pode ser observado no artigo de funcionários da Nokia [Vodde and Koskela 2007].

Em [Siniaalto and Abrahamsson 2007] foram avaliados projetos desenvolvidos usando TDD e foram obtidos indícios de que TDD não produz sistemas com alta coesão quando utilizados por desenvolvedores amadores. Também foi observado que TDD não produz código coeso automaticamente. O simples fato de usar TDD não é sinônimo de qualidade: é necessário treinamento. Por isso a sugestão de usar técnicas de *dojos* de programação. Outras abordagens de ensino de TDD já foram estudadas [Wellington 2005], [Dubinski 2003] e [Pancur et al. 2003] cada uma seguindo uma estratégia e com observações diferentes.

Dojo de programação, pelas suas características pode ser considerado uma atividade que se enquadra em categorias de ensino como *Active Learning*, definida como método instrutivo que engaja o aluno no processo de aprendizado [Prince 2004], *Collaborative Learning* método instrutivo no qual os estudantes trabalham juntos em pequenos grupos em torno de um objetivo comum [Prince 2004], *Cooperative Learning* forma estruturada de trabalho em grupo onde os estudantes perseguem um objetivo comum enquanto são avaliados individualmente [Prince 2004] e *Problem Based Learning* um método instrutivo onde um problema relevante é introduzido no inicio do ciclo instrucional e usado para fornecer contexto e motivação para o aprendizado que segue [Prince 2004].

3. Metodologia

Durante o período da pesquisa, ainda em andamento, realizamos encontros de dojo de programação, sempre com o cuidado de registrar a ata dos encontros com a opinião dos participantes que destacaram pontos positivos e negativos de acordo com sua percepção. Em um dos encontros, alunos de graduação responderam um questionário experimental para determinar o perfil dos participantes. Disponibilizamos um questionário eletrônico respondido por participantes de dojo de programação de diversas regiões do Brasil e com diferentes níveis de experiência. Ainda devemos realizar ao menos um experimento com alunos de mestrado. à seguir mais detalhes sobre a realização da pesquisa.

3.1. Opinião dos participantes

Os participantes de *dojo* de programação alegam que a participação em encontros de *dojo* de programação gera alguns benefícios como a troca de experiências, aprendizado de linguagem de programação que não conhecia, aprendizado de TDD, entendimento de como funciona TDD, dá origem a discussões interessantes, conhecimento

de novos ambientes de desenvolvimento, conhecer ou relembrar bibliotecas de linguagens de programação e conhecer ou relembrar técnicas de programação, além de fazer novos amigos são alguns exemplos registrados nas atas dos encontros [DojoPR 2012]. Essas opiniões ajudaram a identificar os aspectos mais valorizados pelos participantes.

3.2. Questionário experimental

O questionário experimental foi respondido por dez participantes, aplicado para conhecer o perfil dos participantes de dojo de programação. A média de idade foi de 24,6. Um participante era mulher e 9 homens. Participaram 5 estudantes do 3ºperíodo e 5 profissionais. Outras questões, como a frequência de encontros as linguagens preferidas e hábitos como programar nas horas vagas e participação em comunidades de programadores indicam que os participantes na sua maioria possuem interesse adicional em programação. Isso mostrou a necessidade de tomar cuidado na realização da pesquisa pois participantes voluntários já são favoráveis à aprender programação, uma ameaça adicional à validade da pesquisa.

3.3. Questionário eletrônico

Foi disponibilizado um questionário eletrônico, `http://va.mu/XcTM`, respondido por voluntários participantes de dojo de programação de diversas regiões do país. Além do perfil dos participantes o questionário solicitou a opinião dos participantes com relação à prática de dojo de programação e seus efeitos no aprendizado de técnicas de desenvolvimento ágil. A escala de Likert foi usada para as questões de opinião.

3.4. Entrevista

Elizabeth Leddy, consultora de software, integrante do time de desenvolvimento do Plone, sistema de gerenciamento de conteúdo livre, concedeu uma entrevista não estruturada, o autor achou interessante entrevistá-la devido à experiência realizando dojos de programação em suas aulas. Elizabeth é voluntária num espaço hacker mais conhecidos pelo termo em inglês *hackerspace*, chamado Noisebridge [Appelbaum 2010] em São Francisco, Califórnia [Schneeweisz 2010], onde ensina programação com a linguagem Python [Leddy] para pessoas que querem mudar de carreira e desenvolver software para internet, as aulas já estão acontecendo há mais de um ano. Já foram realizadas cerca de três turmas, aproximadamente 50% dos participantes continuam desde o começo e os outros 50% variam a cada nova turma. Elizabeth relatou que, no começo, durante as aulas alguns alunos entendiam rapidamente o assunto e alguns não entendiam, mas ao invés de tirarem as dúvidas eles ficavam quietos. Após conhecer o dojo de programação ela experimentou usar durante as aulas, comentou que a atividade se tornou extremamente popular rapidamente. O tópico ensinado é particularmente complexo, ensinar desenvolvimento web para pessoas que não sabem programar e que querem mudar de carreira. Esse perfil de pessoas precisa aprender conceitos práticos para que possam aplicar em projetos pessoais, num ritmo razoável, sem pressa. O dojo de programação ajudou-os com isso pois a aula segue no ritmo da pessoa que está usando o computador, chamado de piloto. Outro ponto relevante é que as pessoas interagem mais, perguntam mais coisas uns aos outros e criam laços que se estendem para o dia a dia após as aulas, alguns alunos começaram projetos pessoais juntos. A estrutura das aulas é semanal e organizada de forma que a cada semana seja intercalada uma aula para apresentar conceitos e uma

aula prática com dojo de programação. Segundo Elizabeth, o dojo de programação torna a aula mais dinâmica, eventualmente quando a dupla que está usando o computador não consegue resolver alguma coisa é convidado um aluno que tem um pouco mais de conhecimento e o que acontece é que esse aluno, pela confiança e segurança no assunto, fala bastante e ajuda o grupo a caminhar, tornando o ambiente participativo.

3.5. Experimento prático

A última fase planejada da pesquisa é um experimento prático. De acordo com as respostas obtidas nas fases anteriores será realizado um treinamento com uma turma de alunos da disciplina de métodos ágeis. A turma será dividida em dois grupos, um grupo terá 20 horas-aula expositivas de TDD com teoria e prática intercaladas. O outro grupo participará de 20 horas de dojo de programação. O mesmo conteúdo será apresentado aos dois grupos, divididos aleatoriamente, pelo mesmo instrutor. Após essas 20 horas, ambos os grupos devem desenvolver um projeto semelhante. O código do projeto será submetido à um avaliador neutro de forma cega. Além dessa avaliação o código deve ser analisado por ferramenta de análise de cobertura de testes, que examina quais partes do programa foram executadas pelo conjunto de testes.

4. Resultados

O questionário eletrônico ajudou a identificar o perfil dos participantes de acordo com a atividade, representada no gráfico 1.

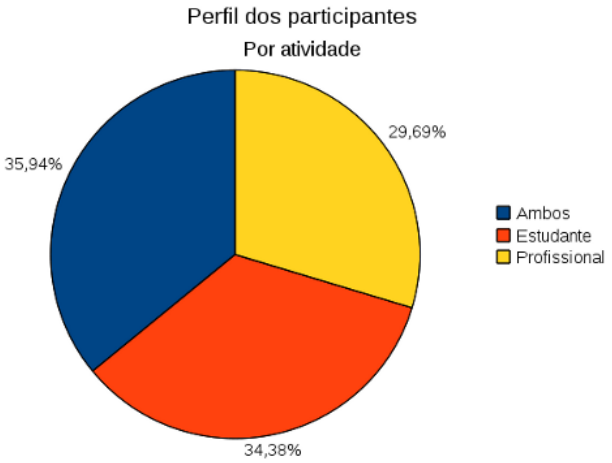

Figura 1. Perfil dos participantes de acordo com a atividade

O perfil dos participantes de acordo com a experiência com passos de bebê, representada no gráfico 2.

Figura 2. Perfil dos participantes de acordo com a experiência com passos de bebê

O perfil dos participantes de acordo com a experiência em anos da participação em dojos de programação, representada no gráfico 3.

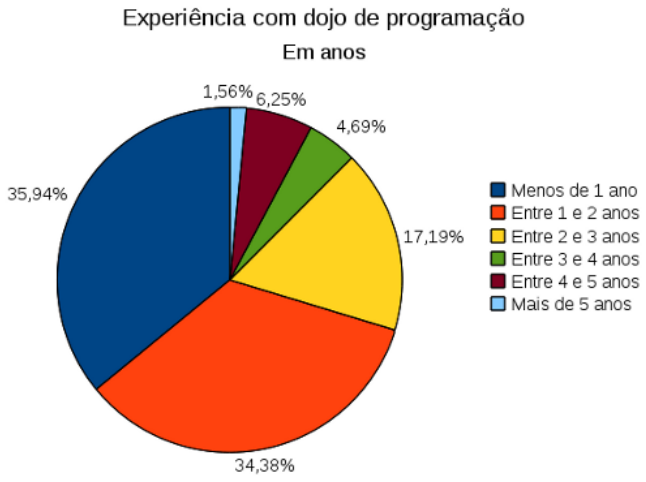

Figura 3. Perfil dos participantes de acordo com a experiência em dojo

O perfil dos participantes de acordo com a experiência em programação em par, representada no gráfico 4.

Figura 4. Perfil dos participantes em programação em par

O perfil dos participantes de acordo com a experiência em TDD, representada no gráfico 5.

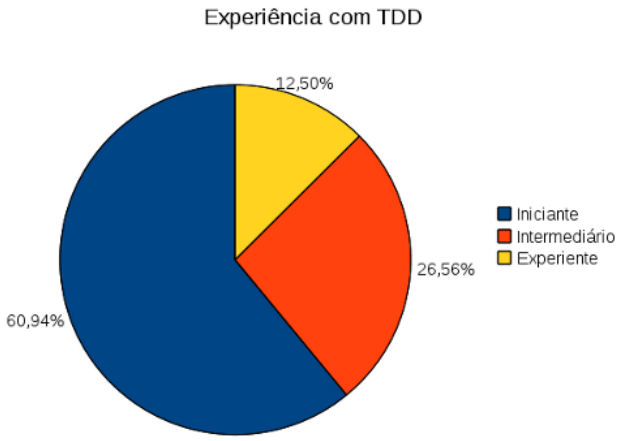

Figura 5. Perfil dos participantes de acordo com a experiência em TDD

As questões do questionário foram apresentadas usando a escala de Likert, variando de -2, -1, 0, 1, 2. Com uma participação de 64 voluntários a maior concordância resulta em 128 e maior discordância em -128. De acordo com as respostas do questionário eletrônico existe uma preferencia moderada, 44 na escala de Likert, para encontros em intervalos pequenos, semanais ao invés de quinzenais. As respostas indicaram uma concordância leve, 29 na escala de Likert, para o conhecimento prévio da linguagem de programação usada. Houve uma discordância moderada , -16 na escala de Likert, para a afirmação de que os participantes precisam ser experientes. A afirmação de que utilização de programação em par atrapalha o andamento da atividade teve uma discordância forte, -91 na escala de Likert. Assim como a afirmação de que passos de bebê atrapalha a atividade, que resultou em -73 na escala de Likert. A utilização de TDD ajuda

no andamento da atividade teve uma forte concordância, 73 na escala de Likert. Houve uma concordância leve, 18 na escala de Likert, de que laboratórios de informática são adequados à realização da atividade. Os participantes concordaram moderadamente que o dojo deva ocorrer num local fixo, 31 na escala de Likert. Houve uma concordância alta de que é fundamental o uso de um quadro ou *flip-chart* para escrever, 61 na escala de Likert. Os participantes discordam de forma moderada que não é necessário possuir um projetor para a dupla comunicar as ideias, -63 na escala de Likert. O dojo de programação favorece o aprendizado de novas linguagens de programação obteve uma forte concordância, 94 na escala de Likert. Os participantes discordam de que o dojo dificulta o aprendizado de TDD, -77 na escala de Likert. O dojo de programação favorece o nivelamento dos participantes por conta da programação em par, recebeu uma concordância moderada, 67 na escala de Likert. Forte concordância para a afirmação de que praticar passos de bebê ensina a resolver problemas de forma gradual, 92 na escala de Likert. Durante a atividade de dojo de programação aprende-se bastante devido à troca de experiências, obteve um alto grau de concordância, 105 na escala de Likert. Uma discordância moderada, 54 na escala de Likert, para a afirmação que é preferível aprender métodos ágeis através do ensino tradicional, com exposição teórica e exercícios individuais.

5. Análise dos resultados

Os resultados do questionário eletrônico indicam que os participantes concordam que o dojo de programação ajuda o aprendizado de métodos ágeis, as questões relacionadas à programação em par, passos de bebê e TDD receberam valores altos na escala de Likert. Outro ponto considerado forte foi a troca de experiencia entre os participantes. Esses resultados direcionaram o planejamento da última etapa da pesquisa, o experimento será limitado à avaliação de TDD, onde usaremos ferramentas de estatísticas de cobertura de testes e avaliação de código por avaliadores independentes.

6. Conclusão

No momento podemos dizer que o dojo de programação favorece a participação incluindo os programadores na ambiente de aprendizado. O dojo de programação favorece a socialização dos programadores, segundo constatamos nas entrevistas. O próximo passo é avaliar o ensino de desenvolvimento dirigido por testes usando dojo de programação a fim de obter software com maiores taxas de cobertura de testes.

Referências

Appelbaum, J. (2010). Noisebridge. https://www.noisebridge.net/wiki/Noisebridge.

Beck (2002). *Test driven development: by example*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Beck, K. and Andres, C. (2004). *Extreme programming explained: embrace change (2nd edition)*. Addison-Wesley Professional.

Bossavit, L. and Gaillot, E. (2005). The coder's dojo – a different way to teach and learn programming. In Baumeister, H., Marchesi, M., and Holcombe, M., editors, *Extreme Programming and Agile Processes in Software Engineering*, volume 3556 of *Lecture Notes in Computer Science*, pages 1156–1158. Springer Berlin / Heidelberg.

Bravo, Mariana and Goldman, A. (2010). Reinforcing the Learning of Agile Practices Using Coding Dojos. In Aalst, Will and Mylopoulos, J., editor, *Agile Processes in Software Engineering and Extreme Programming*, volume 48 of *Lecture Notes in Business Information Processing*, pages 379–380. Springer Berlin Heidelberg.

Bravo, M. V. (2011). Abordagens para o ensino de práticas de programação extrema. Master's thesis, Instituto de Matemática e Estatística da Universidade de São Paulo.

DojoPR, G. (2012). Home dojo-parana/dojo-parana wiki. https://github.com/dojo-parana/dojo-parana/wiki/.

Dubinski, Y. (2003). eXtreme Programming as a Framework for Student-Project Coaching in Computer Science Capstone Courses. In *Software: Science, Technology and Engineering, 2003. SwSTE '03. Proceedings. IEEE International Conference on*.

Janzen, D. and Saiedian, H. (2008). Does test-driven development really improve software design quality? *IEEE Software*, 25:77–84.

Leddy, E. PyClass - noisebridge. https://www.noisebridge.net/wiki/PyClass.

Pancur, M., Ciglaric, M., Trampus, M., and Vidmar, T. (2003). Towards empirical evaluation of test-driven development in a university environment. In *The IEEE region 8 EUROCON 2003. computer as a tool.*, pages 83–86, Ljubljana, Slovenia.

Prince, M. (2004). Does active learning work? A review of the research. *JOURNAL OF ENGINEERING EDUCATION*, 93(3).

Sato, D., Corbucci, H., and Bravo, M. (2008a). Coding dojo: An environment for learning and sharing agile practices. In *Agile, 2008. AGILE '08. Conference*, pages 459 –464.

Sato, D. T., Corbucci, H., and Bravo, M. V. (2008b). Coding dojo: an environment for learning and sharing agile practices. *AGILE Conference*, 0:459–464.

Schneeweisz, E. (2010). San francisco - HackerspaceWiki. http://hackerspaces.org/wiki/San_Francisco.

Siniaalto, M. and Abrahamsson, P. (2007). A comparative case study on the impact of test-driven development on program design and test coverage. In *Proceedings of the first international symposium on empirical software engineering and measurement*, ESEM '07, pages 275–284, Washington, DC, USA. IEEE Computer Society.

Vodde, B. and Koskela, L. (2007). Learning test-driven development by counting lines. *IEEE Software*, 24:74–79.

Wellington, C. A. (2005). Managing a Project Course Using Extreme Programming. *Frontiers in Education, 2005. FIE '05. Proceedings 35th Annual Conference*.

Williams, L. and Kessler, R. (2002). *Pair Programming Illuminated*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.