Implementação de Metodologia de Desenvolvimento Ágil em Projetos com Time Alocado e Não Alocado

André Luís Monteiro da Silva[1], **Luiz Thadeu de A. Cavalheiro**[2]
Norton Trevisan Roman[1], **Marcos L. Chaim**[1]

[1]Escola de Artes, Ciências e Humanidades – Universidade de São Paulo
Av. Arlindo Béttio 1000, Ermelino Matarazzo – 03828-000 – São Paulo – SP

[2]Construtora Andrade Gutierrez S.A.
R. Dr. Geraldo Campos Moreira 375, Brooklin Novo – São Paulo – SP

{andremonteiro,norton,chaim}@usp.br, luiz.cavalheiro@agnet.com.br

Abstract. *This paper describes an experience in implementing the Scrum agile methodology in two distinct projects, one with its developing team hosted in the company's administrative head office, and the other with a team allocated in a different town. Results illustrate the advantages of Scrum over the traditional waterfall methodology, previously used in the company. As an additional result, it was also observed that the distance between the Product Owner and Scrum Master, related to the developing team, may have an impact in the progress of the project.*

Resumo. *Este artigo descreve uma experiência de implementação da metodologia de desenvolvimento ágil Scrum em dois projetos distintos, um com um time de desenvolvedores na própria sede administrativa da empresa, e outro com um time alocado em outra cidade. O resultado dessa experiência ilustra as vantagens do uso do Scrum sobre a metodologia em cascata tradicional, anteriormente utilizada na empresa. Como resultado adicional, foi também observado que a distância entre o Product Owner e o Scrum Master, em relação ao time de desenvolvimento, pode ter um impacto no bom andamento do projeto.*

1. Introdução

Até a virada do milênio, era amplamente difundida, no mercado de produção de *software*, a ideia de que o desenvolvimento profissional de sistemas deveria seguir o assim chamado Modelo em Cascata (*cf.* [Pressman 2001]). Tendo como principal base o planejamento, esse modelo segue uma sequência sistemática linear, indo desde o levantamento de requisitos até o suporte à manutenção, passando pela análise de projeto, codificação e testes com o sistema, conforme ilustra a Figura 1.

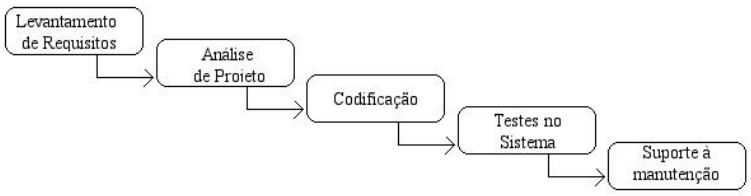

Figura 1. Modelo em cascata.

Embora bastante popular, há aproximadamente duas décadas a eficácia do modelo começou a ser questionada [Pressman 2001], com base em observações de que (i) projetos reais raramente obedecem a esse fluxo sequencial; (ii) é frequentemente difícil para o cliente apontar todos os requerimentos explicitamente; e (iii) o cliente deve ter paciência, pois só obterá uma versão funcional próximo ao fim do projeto. Assim, além de engessar o processo de produção de software, na medida em que uma fase só se inicia após a anterior ter sido completada, essa metodologia faz com que problemas não detectados no início possam ser desastrosos.

Como alternativa a esse modelo, juntamente com as críticas começaram a surgir metodologias consideradas mais "ágeis". Essas, por sua vez, acabaram tomando voz com o surgimento do *Manifesto para Desenvolvimento Ágil de Software* [Beck et al. 2001], que defendia basicamente uma mudança de foco na hora da produção de software. O foco, antes em processos e ferramentas, documentação abrangente, negociação de contratos e existência de um plano de curso, seria direcionado para a valorização de indivíduos e interações, software em funcionamento, colaboração com o cliente e resposta rápida a mudanças, sem contudo esquecer dos demais itens.

Apresentadas como princípios, essas mudanças serviram como base para diversas técnicas de Desenvolvimento Ágil de Software, como Programação Extrema [Kent Beck 2004], Desenvolvimento Guiado por Funcionalidades [Anderson 2004], Rational Unified Process (RUP) [Kruchten 2004] e Scrum [Pichler 2010], para citar alguns (para uma compilação das técnicas mais comuns consulte [Cohen et al. 2003]). Nesse artigo, descrevemos a experiência de implantação de uma dessas metodologias – Scrum – em dois times de desenvolvimento da Construtora Andrade Gutierrez: um alocado na sede da empresa, e outro externo.

Empresa do ramo de Engenharia e Construção, a Construtora Andrade Gutierrez possui uma área de Tecnologia da Informação, contando com um escritório de projetos responsável pela comunicação com os clientes (nesse caso, as demais áreas da empresa, como engenharia, controle, administração e recursos humanos, dentre outras). Além disso, a empresa dispõe de uma Fábrica de Software, cuja missão é gerenciar os processos do ciclo de vida de desenvolvimento de software da área de TI.

Todo o desenvolvimento de sistemas na Construtora Andrade Gutierrez é terceirizado, sendo sua gestão feita pela Fábrica. Dentro da empresa, há dois tipos de terceirização: a primeira ocorre quando os desenvolvedores estão alocados no próprio prédio da empresa, enquanto que a segunda ocorre com desenvolvedores alocados em outra cidade. Durantes muitos anos, a área de TI da Construtora Andrade Gutierrez utilizou a metodologia em cascata, cuja eficiência por muitas vezes era colocada em dúvida dentro da empresa. Por haver muita documentação e planejamento, existia um desperdício de dinheiro e tempo, o que ocasionava muitas discussões internas e com fornecedores.

Há aproximadamente um ano, contudo, a empresa decidiu migrar dessa metodologia para a de desenvolvimento ágil, pois o tempo utilizado para documentação seria melhor utilizado no desenvolvimento, dividindo o processo em partes funcionais a serem entregues. Dentre as metodologias existentes, Scrum foi escolhida pela proximidade de seus papéis com a atual formação da área de TI da Construtora Andrade Gutierrez (para mais detalhes, consulte a Seção 3), com resultados bastante satisfatórios.

A implantação da metodologia, no entanto, não foi algo fácil, havendo inclusive diferenças nos resultados obtidos com o time alocado na sede da empresa e o time externo. Nesse artigo descrevemos o caminho trilhado para implantação do Scrum na empresa, suas dificuldades e os passos seguidos para sua implementação com sucesso, fazendo um comparativo dos resultados de ambos os times.

O restante desse trabalho está organizado como segue. Na Seção 2 é feito um breve resumo da metodologia de desenvolvimento ágil Scrum, descrevendo seus artefatos e papéis. A Seção 3, por sua vez, descreve a metodologia usada para implementação do Scrum na empresa, indo desde a escolha dos projetos até a definição de papéis em cada time. Na Seção 4 são apresentados os resultados obtidos para ambos os times, bem como as dificuldades enfrentadas. Por fim, na Seção 5 apresentamos nossas considerações finais.

2. Scrum

De uso crescente nos últimos anos, o Scrum é sustentado por três pilares [Schwaber and Sutherland 2010]: transparência, inspeção e adaptação. A transparência possibilita que aqueles que gerenciam os resultados consigam visualizar todos os processos que podem afetá-los. Inspeção, por sua vez, indica a necessidade de inspeção frequente no trabalho, de forma que possíveis erros possam ser tratados assim que identificados, evitando maior prejuízo ao cliente. Por fim, por adaptação entende-se que as pessoas devem estar preparadas para mudanças no processo de desenvolvimento (normalmente identificadas durante a inspeção).

2.1. Papéis

A técnica do Scrum define três papéis básicos [Schwaber and Sutherland 2010, Pichler 2010]: Mestre do Scrum (*Scrum Master*), Dono do Produto (*Product Owner*) e Time. O *Scrum Master* tem como objetivo principal defender o Scrum, garantindo que a metodologia esteja funcionando de forma adequada e de acordo com as regras. Entender e desenvolver o time também são funções deste profissional, que deve estar sempre apto a remover os impedimentos que o time possa ter.

Novidade para a maioria das companhias [Pichler 2010], o *Product Owner* tem como principal função a priorização das funcionalidades de acordo com seu valor para o negócio. Ele é o dono do *Product backlog* (veremos mais adiante), garantindo assim o valor do trabalho do time. Crucial para o sucesso do projeto, é de suma importância que esse papel seja preenchido por uma pessoa bastante alinhada com as necessidades de negócio.

Por fim, o time compreende profissionais que devem conhecer profundamente a técnica utilizada para gerar a funcionalidade solicitada pelo *Product Owner*. Embora possam ser especialistas em alguma área específica, esses profissionais devem ter uma multidisciplinaridade, que será obtida através da troca de conhecimento com os outros membros do time. O time deve ser instruído sobre o que deve ser feito, mas não como fazê-lo.

2.2. Time-boxes

O Scrum emprega seis tipos de eventos com duração fixa – as chamadas *time-boxes* – para criar regularidade [Schwaber and Sutherland 2010]. Essas são: Reunião de Planejamento

da *Release*, *Sprint*, Reunião de Planejamento da *Sprint*, Revisão da *Sprint*, Retrospectiva da *Sprint* e Reunião Diária. A Reunião de Planejamento da *Release* acontece no início do projeto, servindo para que todos os representantes dos papéis do *Scrum* estejam juntos e possam definir as diretrizes para o projeto. O tempo desta reunião é bastante reduzido, se comparado ao tempo utilizado nas metodologias tradicionais, sendo que qualquer definição pode ser alterada em futuras reuniões de planejamento de *Sprint*.

Uma *Sprint*, por sua vez, é uma interação com duração fixa de, no máximo, quatro semanas. Nesse caso, cabe ao *Scrum Master* garantir que esse prazo não seja excedido. A Reunião de Planejamento da *Sprint* acontece para definir o que será feito na próxima *Sprint*, contando com a presença de todos os envolvidos. Essa reunião é dividida em duas etapas. Na primeira, o *Product Owner* estipula a sua ordem de prioridade, abrindo para discussão por parte dos demais envolvidos. Na segunda etapa, o time estipula as horas de desenvolvimento para os itens selecionados. Após a reunião, o *Product Owner* ordena os itens selecionados, de acordo com sua prioridade, para que sejam executados na Sprint, sempre respeitando as horas estipuladas para o desenvolvimento.

A Revisão da *Sprint* acontece antes da Reunião de Planejamento da próxima *Sprint* e logo após o time finalizar a *Sprint* que estava em andamento. Nela, o time demonstrará ao *Product Owner* a evolução de suas atividades e, consequentemente, a funcionalidade que o *Product Owner* havia combinado na Reunião de Planejamento. Nesse momento, são também discutidas ideias que ajudarão na definição da próxima *Sprint*, com base nas lições aprendidas nesse período de desenvolvimento.

Após a Reunião de Revisão da *Sprint*, porém antes da reunião de planejamento da próxima *Sprint*, os envolvidos deverão realizar a Retrospectiva. Nesta reunião o *Scrum Master* deve fazer com que o time reflita sobre todas as lições que foram aprendidas durante a *Sprint*. A ideia principal desta reunião é gerar uma melhora contínua através destas lições aprendidas.

Por fim, a Reunião Diária exige a participação apenas do time de desenvolvimento, tendo o *Scrum Master* presença opcional. Este encontro deve acontecer diariamente, com duração fixa de quinze minutos, sempre no mesmo local e horário para evitar problemas de comunicação. Durante a reunião, cada membro explica [Schwaber and Sutherland 2010] (i) o que ele realizou desde a última reunião; (ii) o que ele vai fazer antes da próxima reunião; e (iii) quais obstáculos estão em seu caminho. Muito embora o time seja o responsável por fazer esta reunião acontecer, cabe ao *Scrum Master* garantir que ela aconteça.

2.3. Artefatos do Scrum

São quatro os artefatos usados no *Scrum*: *Product Backlog*, *Burndown Release*, *Sprint Backlog* e *Burndown da Sprint*. O *Product Backlog* nada mais é que uma lista de requisitos para uma *release*. Embora qualquer pessoa possa adicionar itens ao *Product Backlog*, a responsabilidade por sua atualização é do *Product Owner*, que deve decidir a ordem de prioridade de cada requisito.

O gráfico *Burndown Release*, por sua vez, ilustra o andamento do *Product Backlog*. Assim, a partir do total de horas estimado, ele faz uma projeção do que deveria ter sido feito e do que foi feito realmente. Já o *Sprint Backlog* nasce após a reunião de planejamento da *Sprint*, com a finalidade de definir as tarefas a serem executadas dentro

dessa *Sprint*. Com base na projeção do tempo que o time tem e no que o ele desenvolve a cada dia, é gerado o gráfico *Burndown da Sprint*, de modo que o *Product Owner* possa acompanhar o andamento do Sprint, percebendo de maneira rápida qualquer atraso ou desvio.

A Figura 2 ilustra a interação dos artefatos e time-boxes do Scrum. Pode-se ver que inicialmente um *Product Backlog* é analisado, sendo dele formado um *Sprint Backlog*. Este *Sprint* passará por iterações de 2 a 4 semanas, no máximo, devendo todos os dias acontecer a reunião diária. Após esse período o time deve entregar, na Revisão da *Sprint*, uma parte de funcionalidade pronta para produção, que deve necessariamente ser um software útil para o cliente [Schwaber 1995].

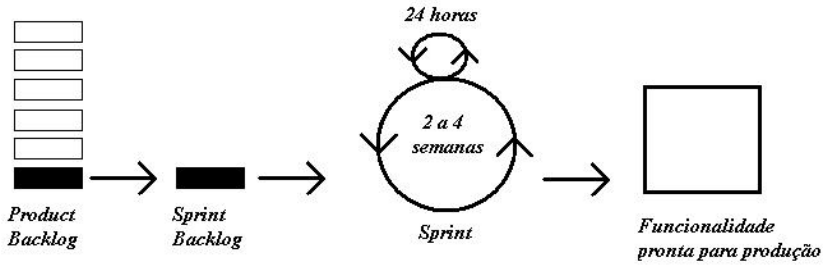

Figura 2. Modelo de funcionamento do *Scrum*.

3. Metodologia

Com a finalidade de testar o funcionamento do *Scrum*, foram escolhidos dois sistemas em desenvolvimento no setor de TI da Construtora Andrade Gutierrez. O primeiro deles – Sistema X[1] – registra as não conformidades que acontecem nas obras da construtora, viabilizando assim a tomada de ações preventivas ou corretivas sobre possíveis incidentes que estejam impactando o negócio. Por sua vez, o segundo sistema – Y – toma por base informações de outros sistemas desenvolvidos, gerando indicadores de TI (principalmente sobre funcionamento de servidores nas diversas obras da empresa), permitindo então o monitoramento de todas as operações dos demais sistemas da construtora.

No que diz respeito à introdução do *Scrum*, no entanto, a principal diferença entre os sistemas se dá na localização do time de desenvolvimento. Enquanto que o Sistema X possui um time alocado na sede administrativa da construtora, o Sistema Y possui seu time em outra cidade. Em ambos os casos, tanto o *Product Owner* quanto o *Scrum Master* permanecem na sede da empresa, ou seja, próximos ao time de desenvolvimento do Sistema X, porém afastados do time responsável pelo Sistema Y. Dessa forma, certos ritos do *Scrum*, como Reuniões de Revisão, por exemplo, no Sistema Y aconteciam somente através de vídeo conferência, e não presencialmente, como era o caso com o sistema X.

Outra diferença entre esses sistemas está em seu grupo de usuários finais, ou seja, seu *Product Owner*. No caso do Sistema X (de responsabilidade de outra área, que não a de TI) os usuários são, em sua maioria, pessoas que participam ativamente da construção da obra. Já no Sistema Y, os usuários são pessoas da área de TI. Dessa forma, enquanto o *Product Owner* do Sistema Y é um profissional da área, o do Sistema X é alguém externo.

[1]Os nomes dos sistemas foram omitidos a pedido da empresa.

Uma vez definidos os sistemas-alvo, foram feitos estudos sobre a antiga metodologia de desenvolvimento neles utilizada, de modo a permitir uma melhor compreensão de como se dava o relacionamento com o cliente, principalmente nas entregas de funcionalidades de software. Em seguida, toda a área de sistemas foi certificada como *Professional Scrum Master I*, por meio de estudos sobre o *Scrum*, a fim de explanar todos os ritos e artefatos necessários para que a metodologia seja implementada da melhor forma possível, de acordo com as adaptações da empresa. Os papéis do *Scrum* foram então preenchidos, dando início à sua implementação nos projetos.

Dentre as principais razões para a escolha do *Scrum* está sua sua flexibilidade em relação às metodologias tradicionais. Seu diferencial em relação às demais metodologias ágeis, no entanto, foi a proximidade de seus papéis com a atual formação da área de TI na Construtora Andrade Gutierrez, uma vez que já havia um time multidisciplinar para desenvolvimento e testes, assim como um analista de sistemas (que executou o papel de *Scrum Master*) e um analista de negócio (que assumiu o papel de *Product Owner*).

Compostos de oito *Sprints* de duas semanas, cada projeto teve uma duração de quatro meses. Nesse tempo, em cada sistema foram feitas oito Reuniões de Planejamento da *Sprint*, oito Reuniões de Revisão da *Sprint*, oito Reuniões de Retrospectiva da *Sprint* e oitenta e quatro[2] reuniões diárias de cerca de 15 minutos, além da Reunião de Planejamento da *Release*. Esses números foram necessários para garantir uma aderência total, por parte de todos os envolvidos, à metodologia do *Scrum*.

4. Resultados e Discussão

A Tabela 1 apresenta os resultados para o número de reuniões executadas em cada projeto. Nela, pode-se ver que apenas no Sistema X foi possível fazer todas as reuniões previstas pelo modelo. Já para o Sistema Y, a eficiência em relação à aderência da metodologia de desenvolvimento ágil foi inferior.

Tabela 1. Reuniões executadas em cada projeto.

	Número de Reuniões			
	Diárias	Planejamento	Revisão	Retrospectiva
Sistema X	84	8	8	8
Sistema Y	67	8	6	4

Embora haja diferenças em relação às demais reuniões, pode-se notar, nessa tabela, que as reuniões de planejamento – ocorridas antes do início de cada uma das oito *Sprints* – foram todas executadas em ambos os sistemas, indicando uma maior preocupação por parte dos envolvidos com esse tipo de questão. As reuniões diárias, de planejamento e de retrospectiva, por outro lado, sofreram uma redução, no Sistema Y, de 20% (nas diárias) a 50% (nas de retrospectiva).

Essa diferença, ainda que não estatisticamente significativa ($t(df = 3) = 1.4982$, $p = 0.231$), pode ter sido devida ao fato de, no Sistema X, o time de desenvolvimento estar alocado na sede administrativa e, portanto, trabalhando todos os dias ao lado de seu *Scrum Master* e *Product Owner*. No Sistema Y, por outro lado, não havia esta proximidade, o que se mostrou um problema ao executar algumas reuniões.

[2]Quatro meses de 21 dias úteis cada.

Com relação aos Sprints executados, a Figura 3 apresenta o gráfico *Burndown Release* para o Sistema X. Nela, é feita uma comparação entre a projeção do tempo previsto para produção (linha tracejada) e o que foi efetivamente produzido (linha contínua). No eixo vertical está o tempo restante até o fim do projeto, enquanto que no eixo horizontal é registrado o número de *Sprints* executados (num total de oito). É interessante notar que, logo após o terceiro *Sprint* o time ficou adiantado em relação ao tempo previsto, alinhando-se novamente a este somente ao final do projeto.

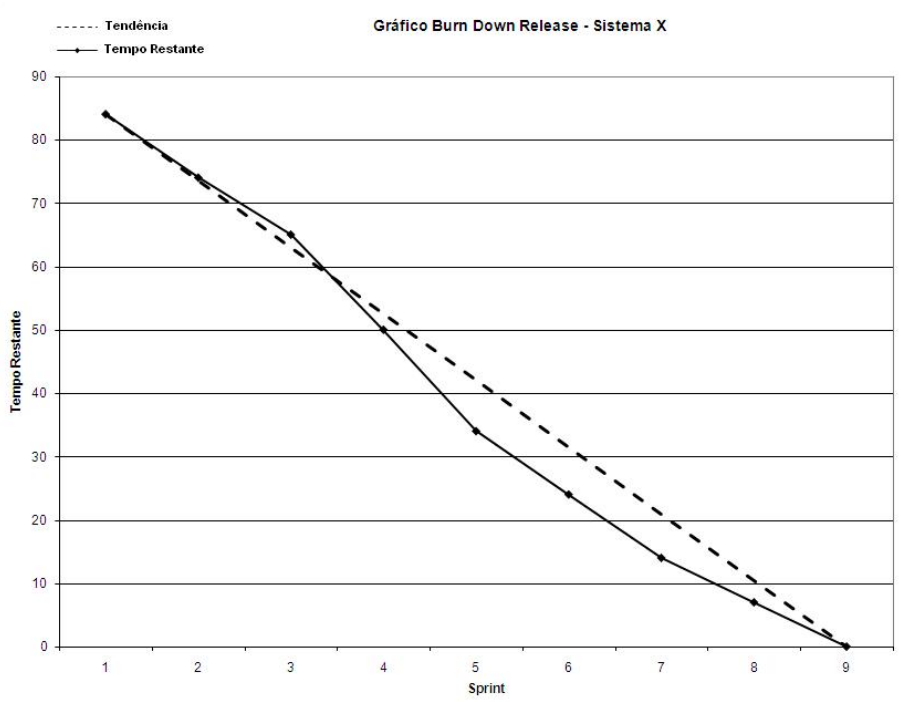

Figura 3. *Burndown Release* para o Sistema X.

O gráfico *Burndown Release* para o Sistema Y, por sua vez, pode ser visto na Figura 4. Ao contrário do Sistema X, podemos notar que, nesse sistema, logo na segunda *Sprint* o time começou a acumular atrasos, atingindo um pico na sétima *Sprint*. Isso fez com que os desenvolvedores tivessem que trabalhar bastante no final do projeto, havendo inclusive a necessidade de horas-extras, de modo a chegar ao resultado desejado no final do projeto.

Assim, mais uma vez verificamos uma diferença entre o time alocado na sede da empresa e o time externo, com resultados piores para o time externo. Essa diferença, por sua vez, pode ser devida ao fato de, no início do projeto, o papel de *Product Owner* ser compartilhado por duas pessoas, fazendo com que houvesse uma demora em definir algumas atividades para o time. Ainda que o time tenha ocupado seu tempo ocioso em tarefas de outros projetos, reduzindo assim o prejuízo para a empresa, a demora na definição de um *Product Owner* único para esse time, aliado à distância entre o time de desenvolvimento e seu *Scrum Master* e *Project Owner*, podem ter contribuído consideravelmente tanto para a existência de atrasos, quanto para o reduzido número de reuniões diárias.

Nessa mesma linha, uma última barreira encontrada quando da implementação do *Scrum* em ambos os projetos foi a dificuldade de conscientização do *Product Owner*

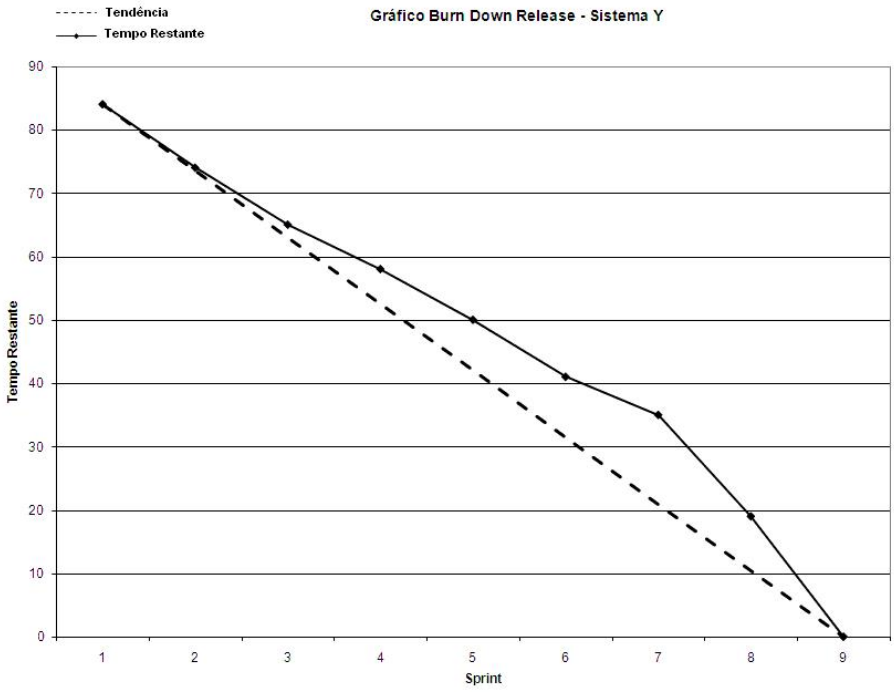

Figura 4. *Burndown Release* para o Sistema Y.

de suas obrigações e responsabilidades (algo já esperado, dada a novidade do conceito [Pichler 2010]). Uma alternativa para resolver esse problema, seria certificar o *Product Owner* como *Professional Scrum Product Owner*, por exemplo, para que este veja mais claramente a importância de seu papel. Esta certificação possibilitaria, por sua vez, que o *Product Owner* entendesse, desde o início do projeto, que o seu papel é fundamental nesta nova metodologia, que suas escolhas e sua priorização são os fatores que dão valor a todo o trabalho que é feito, e que ele é quem deve saber o que é prioritário para o negócio, tendo toda a responsabilidade com relação ao retorno sobre o investimento.

5. Conclusão

Nesse artigo apresentamos os resultados da experiência de implementação de *Scrum* em dois times de desenvolvimento (um alocado na sede e outro externo) da Construtora Andrade Gutierrez. Pudemos notar que, em ambos os projetos, o *Scrum* foi fundamental para uma entrega pontual e funcional para o cliente. Além disso, o cliente também demonstrou estar bastante satisfeito em relação à flexibilidade que o *Scrum* provém.

Anterior à implementação do *Scrum*, o cliente muitas vezes solicitava inovações urgentes que impactavam o trabalho atual da equipe de desenvolvimento, fazendo com que muito trabalho fosse perdido. Depois do *Scrum*, o cliente, agora *Product Owner*, passou a sentir-se parte da equipe, acompanhando gráficos de *Burndown*, e conseguindo perceber quando algo seria ou não entregue, podendo então intervir na *Sprint*, caso julgasse necessário. Esta proximidade, por sua vez, fez com que o time atuasse de maneira mais produtiva, uma vez que suas dúvidas eram prontamente sanadas pelo *Product Owner*. Talvez esse tenha sido um dos motivos pelos quais os resultados tenham apontado um desempenho melhor por parte do time que se encontrava mais próximo fisicamente do *Product Owner*.

Vale também lembrar que, no Sistema X, que possuía um time de desenvolvedores alocado na sede da construtora, todas as reuniões previstas pela metodologia foram executadas, o que possibilitou um maior controle da produtividade e andamento da equipe. Já no Sistema Y, onde o time de desenvolvedores não estava alocado na construtora, algumas reuniões não aconteceram. Os encontros marcados por vídeo-conferência eram, muitas vezes, cancelados e, pelo time estar em outra cidade, havia uma maior dificuldade para que estas fossem remarcadas. Desta forma, a dificuldade de comunicação ocasionada pela distância certamente contribuiu negativamente para a implementação do *Scrum*, ainda que não tenha sido fator impeditivo para o sucesso do projeto.

Podemos então dizer que o *Scrum* tem várias características marcantes que o diferem das metodologias antigas como, por exemplo, o desenvolvimento em cascata, antes utilizado na Construtora Andrade Gutierrez, e que mostrou-se muito ineficiente. Os fatores que mais influenciaram no sucesso dos projetos têm relação à concepção de proximidade com cliente e excelência na comunicação. Nesse sentido, e ainda que não afete de maneira drástica o resultado, é aconselhável que o *Scrum Master* e o *Project Owner* estejam na mesma localização geográfica do time de desenvolvimento. Além disso, também notamos a necessidade de tornar o papel *Project Owner* claro e oficial, certificando-o como *Professional Scrum Product Owner*.

Por fim, é importante notar que, independente do sistema, com desenvolvedores alocados na sede ou não, o comprometimento de um time de *Scrum* é o diferencial para o sucesso de um projeto, pois cada atraso que é percebido em um gráfico de *Burndown* é compensado em uma próxima iteração, fazendo com que mesmo equipes que estejam desalinhadas em relação ao cronograma consigam entregar o resultado com qualidade e no prazo determinado. Dessa forma, é importante manter a equipe inteira comprometida e comunicativa, a fim de resolver os conflitos de forma ágil e eficiente.

6. Agradecimentos

Agradecemos à Construtora Andrade Gutierrez pelo apoio à divulgação dessa experiência.

Referências

Anderson, D. J. (2004). Feature-driven development: towards a toc, lean and six sigma solution for software engineering. In *TOC ICO World Conference*, Miami, EUA.

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J., and Thomas, D. (2001). Manifesto para desenvolvimento Ágil de software. http://agilemanifesto.org/iso/ptbr/. Acesso em 10/01/2012.

Cohen, D., Lindvall, M., and Costa, P. (2003). Agile software development a dacs state-of-the-art report. Technical report, Fraunhofer Center for Experimental Software Engineering Maryland and The University of Maryland.

Kent Beck, C. A. (2004). *Extreme Programming Explained: Embrace Change*. Addison Wesley Professional, 2 edition. ISBN: 0-321-27865-8.

Kruchten, P. (2004). *The rational unified process: an introduction*. Pearson Education, 3 edition. ISBN 0-321-19770-4.

Pichler, R. (2010). *Agile Product Management with Scrum: Creating Products that Customers Love*. Addison-Wesley.

Pressman, R. S. (2001). *Software Engineering – A Practitioner's Approach*. McGraw-Hill, 5 edition.

Schwaber, K. (1995). Scrum development process. In *Proceedings of the 10th Annual ACM Conference on Object Oriented Programming Systems, Languages, and Applications (OOPSLA'95)*, pages 117–134, Austin, EUA.

Schwaber, K. and Sutherland, J. (2010). Scrum guide. `http://www.Scrum.org/storage/Scrumguides/Scrum%20Guide%20-%20PTBR.pdf`. Acessado em 11/01/2012.