Synergia-Ágil: O desafio de implantar métodos ágeis em uma organização com processo tradicional maduro

Eduardo Borges[1], Raquel Lara[1], Eduardo Habib Bechelane Maia[1], Clarindo Isaias Pereira da Silva e Padua[1], Wilson de Pádua Paula Filho[1]
[1]Synergia - Universidade Federal de Minas Gerais
Av. Antônio Carlos 6627 - CEP 31270-010 - Belo Horizonte – MG - Brasil
{eborges,raqlara,habib,clarindo}@dcc.ufmg.br;
wppf@ieee.org

Abstract. As metodologias ágeis propõem uma nova abordagem para o desenvolvimento de software onde os gastos excessivos com formalismo são eliminados, ao mesmo tempo em que priorizam o bom relacionamento com as pessoas que participam do projeto, a adaptação às mudanças e as atividades prioritárias para o usuário final. Este trabalho relata uma experiência de desenvolvimento e evolução de um processo que combina práticas do Scrum e do XP em uma organização que utilizava o processo tradicional, mostrando as dificuldades e os benefícios obtidos na migração de um processo tradicional para o ágil.

1 Introdução

Desde 2000, o Synergia - Laboratório de Engenharia de Software do Departamento de Ciência da Computação da Universidade Federal de Minas Gerais (UFMG) - oferece serviços de desenvolvimento de sistemas para órgãos públicos, fundações e empresas privadas. Por estar inserido na Universidade, o laboratório tem a cultura de forte valorização de processos, métodos e boas práticas da Engenharia de Software. Tal característica favoreceu sua consolidação, ao longo dos anos, como referência em qualidade no desenvolvimento de sistemas de informação.

A maturidade do seu processo de desenvolvimento, denominado Praxis-Synergia, foi alcançada após anos de investimento em práticas de monitoramento e melhoria contínua. Trata-se de uma versão personalizada do processo Praxis (PRocesso para Aplicativos eXtensíveis Interativos) [1], adaptada para se tornar mais adequada ao contexto dos projetos desenvolvidos no laboratório – projetos de médio e grande porte onde era necessário garantir altos níveis de qualidade e robustez.

Devido a esse contexto, o Praxis-Synergia se desenvolveu e evoluiu a partir de um forte investimento em qualidade, focado principalmente em três aspectos: (1) formalismo nas especificações de requisitos, geralmente produzidas em uma etapa anterior ao restante do desenvolvimento do produto[1]; (2) Modelagem e documentação detalhadas e (3) alto investimento em qualidade através de revisões, inspeções e testes.

O foco em tais práticas era o estado da arte dos processos de desenvolvimento nos anos 2000 e, de fato, trouxe resultados excelentes no controle dos projetos, qualidade do produto final, conformidade com os requisitos e diminuição da taxa de defeitos.

[1] Embora o processo oficial oferecesse a abordagem iterativa e incremental, a maior parte dos contratos de projetos exigia uma especificação prévia dos requisitos.

1.1 Com um processo já maduro, por que mudar?

A necessidade de mudança surgiu quando pôde-se verificar a ocorrência de problemas inerentes a processos com essas características.

Os projetos de desenvolvimento tinham em sua maioria mais de um ano de duração até a entrega da primeira versão operacional do produto. Nesse período, eram inevitáveis as alterações em requisitos em virtude de mudanças e prioridades dos clientes. Essas alterações oneravam o projeto, já que envolviam retrabalho não só no código-fonte, mas em todos os artefatos, que eram bastante detalhados e complexos.

Ao mesmo tempo em que garantia um alto nível de qualidade para o produto final, o grande investimento em revisões, inspeções e testes elevava o custo do projeto. Como o laboratório sempre teve a cultura de priorizar a qualidade dos produtos em detrimento do prazo e do custo, esse investimento se tornou cada vez maior com a evolução do processo. Em sua versão mais recente, eram previstas seis inspeções formais sobre cada requisito, além de validações formais realizadas pelo cliente e diversos tipos de testes, fazendo com que cerca de 50% do custo do projeto fosse gasto com essas apreciações. Apesar de contribuir para a qualidade dos produtos, isso com frequência inviabilizava o início de novos projetos por serem considerados caros.

Outro problema percebido no processo tradicional[2] foi o *overhead* de esforço e custo representado pelas sucessivas trocas de responsáveis, na equipe, pelo desenvolvimento de um mesmo requisito. Tarefas de especificação e análise, desenho, implementação e testes eram desempenhadas por profissionais diferentes e a própria organização da equipe técnica era feita tomando como base essa divisão de papéis.

A partir do momento em que foram detectados esses problemas, foi verificada a necessidade de um novo processo que buscasse solucioná-los e que fosse seguido de forma consistente em todos os projetos do laboratório. Após acompanhar, nos últimos anos, o desenvolvimento e a consolidação das metodologias ágeis, optou-se por criar o novo processo baseado em tais práticas e aplicá-lo em projetos piloto, com a expectativa de que elas evitassem os problemas aqui descritos.

1.2 As metodologias ágeis

As metodologias ágeis surgiram visando melhorar o desempenho de projetos, tirando o foco do processo e dando ênfase à contribuição dos participantes do projeto. Surgiram quando, em 2001, 17 gurus do desenvolvimento de software criaram o "Manifesto para o Desenvolvimento Ágil de Software" [3], que definiu os princípios básicos que os métodos ágeis deveriam valorizar:

- **Indivíduos e interações entre eles**, mais que processos e ferramentas;
- **Software funcionando**, mais que documentação compreensiva e detalhada;
- **Colaboração com os clientes**, mais que negociação de contratos;
- **Adaptação às mudanças**, mais que seguir o plano inicial.

[2]Neste artigo, utilizamos o termo "tradicional" para se referir a processos de desenvolvimento de software baseados em paradigmas anteriores ao das metodologias ágeis, incluindo os processos baseados nos modelos em "cascata" e derivados do Processo Unificado [1].

[3] Disponível em: http://www.agilemanifesto.org/

As metodologias ágeis pregam, portanto, o desenvolvimento de sistemas com base na colaboração entre os envolvidos, trabalho em equipe, comunicação informal e adaptação às mudanças que surgirem durante o desenvolvimento do projeto. Assim como no modelo iterativo e incremental, o software é dividido em versões, mas no caso dos processos ágeis, uma versão nova deve ser produzida de uma a quatro semanas, passando por todas as etapas desde especificação até a fase de testes.

1.3 O processo Synergia-Ágil

O desenvolvimento do novo processo, batizado de Synergia-Ágil, teve a preocupação de evitar a falácia de adotar o formato dos métodos ágeis sem incorporar o conteúdo do Manifesto Ágil. Segundo Stuart Reid[4], na realidade existem poucas empresas que utilizam processos realmente ágeis e que seguem todos os princípios do manifesto.

O desenvolvimento do Synergia-Ágil baseou-se principalmente em duas das metodologias ágeis: o Scrum [2] e o XP [3]. A combinação dessas duas metodologias foi escolhida porque, conforme Fitzgerald et al [4], enquanto as práticas do Scrum são mais voltadas para o planejamento e o gerenciamento do projeto, as práticas do XP são mais voltadas para a parte técnica do desenvolvimento do software. Além disso, pesquisas feitas em [9] e [10] mostram que o Scrum e sua combinação com o XP constituem as duas metodologias ágeis mais difundidas nas organizações atualmente.

Durante os dois anos em que foi desenvolvido, o novo processo foi aplicado em dois projetos piloto. A experiência adquirida nesses dois projetos foi fundamental para definir e amadurecer o processo e antecipar problemas e dificuldades que certamente surgiriam durante sua implantação. As seções 2 e 3 descrevem as experiências desses projetos, enquanto a seção 4 descreve as principais lições aprendidas e como estas contribuíram para evoluir o processo. Finalmente, a seção 5 resume as conclusões obtidas.

2 Primeiro piloto - explorando as metodologias ágeis

O primeiro projeto a utilizar metodologias ágeis no laboratório foi desenvolvido durante o seu primeiro Programa de Residência em Software. Tal programa mobilizou uma equipe de 10 estudantes com carga horária de 20 horas semanais com o objetivo de desenvolver uma pequena aplicação de otimização de rotas de transporte.

A proposta principal do programa era a de identificar talentos profissionais para o laboratório e obter melhor clareza dos benefícios da prática de residir estudantes nas suas dependências, haja vista que essa técnica já foi amplamente registrada na literatura em diferentes empresas ([5] e [6]). Como o projeto que seria desenvolvido apresentava também condições ideais para testar e aprimorar o novo processo, o laboratório decidiu utilizá-lo como um primeiro piloto para o Synergia-Ágil. Foram condições motivadoras o porte pequeno do produto que seria desenvolvido, o fato de que seria desenvolvido como projeto interno do laboratório e a existência de incertezas quanto à viabilidade de alguns de seus requisitos.

O processo Synergia-Ágil ainda não estava completamente descrito, mas o projeto já utilizou o Scrum para a parte gerencial e algumas práticas XP para a parte técnica.

[4] Disponível em: http://www.mynewsdesk.com/uk/pressroom/testing-solutions-group/pressreleases/download/resource_attached_pdf_document/773486

O programa teve duração de seis meses e o software desenvolvido consistia em uma aplicação *front-end* para algoritmos de otimização de rotas.

A primeira fase do programa teve a duração de cinco semanas e foi dedicada para treinamento dos residentes. Foram oferecidos treinamentos em Scrum, engenharia de requisitos, introdução a Unified Modeling Language (UML), introdução às ferramentas de desenvolvimento, desenvolvimento dirigido a testes (Test Drive Development - TDD), testes de unidade, programação orientada a objetos, *framework* de persistência e *framework* de interface de usuário para web.

Na segunda fase, de três semanas, ocorreu a modelagem inicial dos requisitos.

A terceira fase foi a de desenvolvimento do produto e teve duração de quatro meses. Essa fase foi mentoreada por um programador Java experiente e seguiu todos os eventos Scrum e um subconjunto das práticas XP (enumeradas na seção 4).

Além do mentor e dos 10 residentes, a equipe do programa contou com uma engenheira de processos envolvida diretamente na personalização e documentação do Synergia-Ágil, no papel de scrum master; uma analista de requisitos no papel de dona do produto; um engenheiro de usabilidade para consultoria e criação do guia de estilo do projeto e um testador estagiário que não fez parte do time de desenvolvimento, mas dedicou horas no final de cada sprint para testar o incremento produzido.

A última fase teve duração de dois dias, quando os residentes receberam feedback sobre seu desempenho e responderam a um questionário de avaliação do programa.

2.1 Resultados e conclusões

O desenvolvimento do produto foi feito em quatro Sprints de um mês cada, nas quais se pôde acompanhar a evolução do time e obter lições importantes para a evolução do processo Synergia-Ágil. Os principais resultados estão elencados abaixo:

- **Papel de Scrum Master indefinido no início.** Inicialmente, o mentor Java também era o Scrum Master. Porém, ele acabou não conseguindo desempenhar o papel de Scrum Master, comprometendo a execução do processo. A solução foi inserir a partir da segunda Sprint, uma engenheira de processos, que entendia bem a metodologia Scrum, com o papel de Scrum Master.
- **A produtividade foi baixa na primeira sprint.** Percebeu-se que a inexperiência e baixa maturidade profissional do time (formado por alunos sem experiência profissional anterior) fez com que o trabalho autogerenciável não funcionasse bem. A solução foi dar atribuições gerenciais à Scrum Master, que precisou interferir na organização do trabalho e motivar o trabalho em equipe, até que o time conseguisse desenvolver o conceito de equipe e determinar o que poderiam entregar.
- **Testador externo ao time.** Mesmo com o time testando o produto a cada sprint, o testador forneceu uma visão imparcial que contribuiu com a qualidade do produto.
- **Dois residentes tiveram a iniciativa de sair durante programa.** Um saiu durante a segunda sprint e declarou não conseguir acompanhar o ritmo dos colegas, sentindo que estava prejudicando o progresso do time. O outro saiu durante a terceira Sprint após ter apresentado pouco comprometimento com o projeto e dificuldade de trabalhar em equipe. Tais resultados reforçam a ideia de que um time ágil, quando bem integrado e comprometido, naturalmente seleciona quem faz parte dele.

- **Avaliação da metodologia Scrum.** A partir da segunda Sprint, todos os eventos previstos no Scrum foram realizados e os artefatos utilizados e atualizados. Isso deixou o time mais coordenado, comprometido e o processo de desenvolvimento mais transparente para todos.
- **Avaliação das práticas XP.** Percebeu-se que a programação em par facilita a troca de conhecimento e a percepção de defeitos em tempo de codificação, mas os residentes tiveram dificuldades de aplicar a prática com alguns integrantes que tinham dificuldade de trabalhar em equipe. As demais práticas: jogo do planejamento, metáfora comum, integração contínua, propriedade coletiva do código, semana de 20 horas, regras adaptáveis e prototipação foram aplicadas com sucesso. O desenvolvimento dirigido por testes não se aplicou na prática, já que o teste de unidade foi aplicado somente a partir da terceira sprint.
- **Avaliação de qualidade.** Ao final do programa, um arquiteto do laboratório fez uma avaliação do código do produto. Na sua avaliação qualitativa, considerou que o produto possuía baixo reuso e falta de padronização. Além disso, considerou que o produto teria problemas de desempenho caso a base de dados crescesse. Em sua visão, o produto tinha problemas de qualidade que seriam graves para um software comercial, mas aceitáveis para o contexto e os objetivos do programa.
- **Avaliação do programa de residência.** O questionário de avaliação do programa foi respondido por nove dos 10 residentes e o resultado da pesquisa mostrou que os residentes cresceram tecnicamente adquirindo conhecimentos das diversas áreas da Engenharia de Software. A autoavaliação dos residentes mostrou que eles sentiram que amadureceram profissionalmente, sendo capazes de integrar uma equipe de desenvolvimento de software pós-programa. De fato, o laboratório contratou cinco dos residentes para fazerem parte do seu quadro de colaboradores.

3 Segundo piloto - consolidando o processo

A segunda experiência de utilização de processo ágil na organização foi no projeto de desenvolvimento de uma plataforma colaborativa de criação e compartilhamento de conteúdo. O projeto foi escolhido como segundo piloto por reunir, assim como o anterior, características favoráveis a isso. Neste caso, as principais motivações foram:

- Era um projeto interno desenvolvido com recursos próprios, onde era possível assumir mais os riscos decorrentes da implantação de um novo processo;
- O produto desenvolvido não tinha grande complexidade e seu porte era relativamente pequeno (cerca de 260 Pontos de Função);
- O projeto tinha características que dificultavam a adoção do processo tradicional, como a dificuldade em detalhar os requisitos *a priori* sem o feedback do uso do produto, e também a adoção de uma nova tecnologia cujas possibilidades e limitações só seriam conhecidas no decorrer do projeto, dificultando que o planejamento do escopo e a estimativa de produtividade fossem realizados da forma tradicional.

O novo projeto, entretanto, tinha duas diferenças importantes em relação ao anterior. A primeira foi que o time de desenvolvimento formado tinha experiência com o processo tradicional de desenvolvimento adotado na organização. Outra diferença foi

que, apesar de ser desenvolvido com recursos próprios, o produto resultante do projeto seria cedido para uso por um órgão do Governo de Minas Gerais, que não só ofereceu apoio institucional como também acompanhou o projeto e participou do planejamento e retrospectiva das Sprints, auxiliando o Dono do Produto na definição de prioridades e na aceitação dos resultados apresentados.

O projeto teve duração de nove meses, onde ocorreram treze Sprints de desenvolvimento que resultaram na primeira entrega do produto em ambiente de produção. A equipe, a despeito de algumas alterações ao longo do projeto, manteve um tamanho médio de nove integrantes, sendo seis componentes do time de desenvolvimento, um Scrum Master, um Dono do Produto e um consultor técnico (arquiteto de software). Assim como no primeiro piloto, foram adotados todos os eventos Scrum e parte das práticas XP, conforme detalhado na seção 4.

3.1 Resultados e conclusões

Mesmo com a experiência do projeto anterior, o segundo piloto enfrentou dificuldades na assimilação e na aplicação das práticas ágeis. Parte disso se deveu ao próprio processo que estava pouco maduro e tinha pontos ainda não definidos. Outra parte foi decorrente da dificuldade da equipe em assimilar o novo paradigma, já que a maior parte dela tinha experiência com o processo tradicional da organização.

Dificuldade 1: Trocar especificações pela colaboração com o Dono do Produto.

Um problema reportado ao longo da primeira metade do projeto foi a dificuldade do time em abandonar as especificações formais de requisitos do processo tradicional e substituí-la pela colaboração contínua com o Dono do Produto. Nas primeiras Sprints, muitas histórias de usuário eram rejeitadas durante a reunião de revisão, a maior parte por detalhes menores que teriam sido evitados se os requisitos fossem mais discutidos com o Dono do Produto. A equipe chegou a argumentar que não havia sido especificado em nenhum lugar que a funcionalidade deveria funcionar de forma diferente daquela que foi desenvolvida, demonstrando que os princípios do Manifesto Ágil não haviam sido bem assimilados. Tal dificuldade só foi contornada com a inclusão, no "conceito de pronto" das histórias de usuário, da exigência de que a história fosse validada pelo Dono do Produto antes da reunião revisão. Isso forçou a equipe a interagir mais com o Dono do Produto, até o ponto em que ela mesma percebeu como tal interação é fundamental para o sucesso do projeto.

Dificuldade 2: Desenvolver a arquitetura ao longo do projeto (e não a priori).

O time de desenvolvimento estava habituado a ter uma etapa específica de definição de arquitetura antes do início do desenvolvimento das funcionalidades e se mostrou resistente à ideia do paradigma ágil de que a arquitetura fosse evoluída e consolidada ao longo das Sprints, na medida em que as funcionalidades fossem desenvolvidas e o código fosse continuamente refatorado. Tal resistência foi agravada por dois fatores: o uso de uma nova tecnologia, pouco conhecida pelo time, e a limitação de disponibilidade de tempo que o arquiteto de software mais experiente podia dedicar ao projeto. A preocupação não era infundada e, de fato, houve muito esforço consumido em correção de problemas de arquitetura do produto. A partir dessa experiência, foi inserida no processo uma atividade no começo do projeto, em que um Arquiteto de Software experiente ajuda o time a propor uma arquitetura inicial.

Dificuldade 3: Desenvolver em pequenos incrementos.

No processo tradicional, a equipe estava habituada a que o desenvolvimento de cada funcionalidade levasse vários meses desde sua especificação até a entrega ao cliente. A transição para o desenvolvimento iterativo e incremental trouxe desafios.

O primeiro deles foi dimensionar o escopo que caberia em uma Sprint e planejá-la de forma a entregar as funcionalidades do produto com qualidade ao final da Sprint. Nas primeiras, houve otimismo nas estimativas e o escopo não foi totalmente cumprido. Tal otimismo era mantido também durante a sprint: mesmo com o gráfico de *burndown*[5] mostrando atraso na sprint, o time optava por manter o escopo acreditando que os problemas encontrados seriam solucionados e que o atraso seria compensado, o que não ocorria. Tal comportamento fica claro ao observar o gráfico de *burndown* de uma das primeiras sprints do projeto, mostrado na Figura 1-a.

O problema, recorrente nas primeiras sprints, foi contornado durante o projeto e o resultado pode ser observado na Figura1-b, que mostra o gráfico de *burndown* da última Sprint do projeto, que teve seu escopo completamente cumprido. Tal melhora foi decorrente de uma série de ações tomadas ao longo do projeto, entre elas:

- **Introdução do uso de *story points*** [7] para apoiar o dimensionamento do escopo das Sprints, reduzindo a subjetividade das estimativas. Naturalmente, houve alguma dificuldade no início para assimilação da técnica, mas ela começou a trazer resultados rápidos e a velocidade (número de *story points* por Sprint) passou a ser um indicador confiável para dimensionar melhor o escopo de cada Sprint.
- **Redução do tamanho das Sprints** de quatro para duas semanas chegando a haver Sprints de uma semana ao final do projeto. Isso facilitou o planejamento das tarefas (é mais fácil planejar uma ou duas semanas do que um mês) e ajudou a fortalecer no time o conceito do desenvolvimento em pequenos incrementos.
- **Maior rigor no aceite das histórias de usuário** pelo Dono do Produto. No início do projeto, histórias eram consideradas aceitas ainda que necessitassem ajustes, o que dava uma falsa impressão de produtividade e gerava um *overhead* em cada Sprint com tarefas de correção e melhoria de histórias de Sprints anteriores.

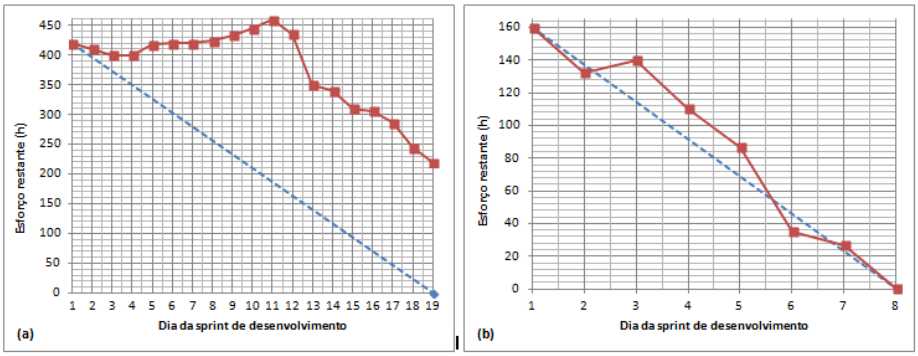

Fig. 1. Comparação do gráfico de burndown entre as sprints 4 e 13

[5] *Burndown* é uma representação gráfica utilizada pelas equipes Scrum para demonstrar diariamente o progresso do projeto em desenvolvimento.

Dificuldade 4: Instabilidade de escopo e incerteza de prazo.

No início do projeto, o planejamento era focado no curto prazo (próxima sprint) e não havia restrições claras de escopo e prazo para a conclusão da primeira versão operacional do produto. Foram necessárias várias Sprints até que o Dono do Produto adquirisse uma visão clara do que seria o produto mínimo viável[6] [8], considerado suficiente para a primeira entrega em produção. Isso fica visível no burnup da entrega (Figura 2), que mostra um aumento do escopo no decorrer do projeto. Foi necessário aumentar o envolvimento do Dono do Produto, em termos da carga horária dedicada ao projeto, para que ele conseguisse desenvolver sua visão de escopo e de prioridades. Só após esse aumento, entre as Sprints 8 e 9, o escopo ficou mais estável e foi possível fazer um planejamento mais confiável de quando seria a entrega do produto.

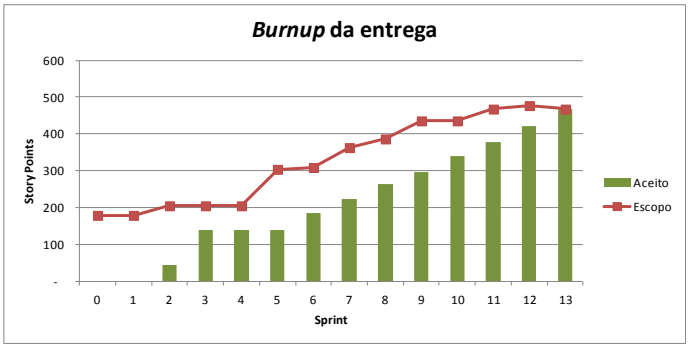

Fig. 2. Gráfico de *burnup* da entrega do produto.

4 A evolução do processo e as lições aprendidas

Paralelamente à execução dos projetos piloto, a equipe de processos do laboratório dedicou-se a estruturar e documentar o Synergia-Ágil utilizando como feedback a experiência obtida nos dois projetos. O processo foi gradualmente evoluindo do esboço inicial adotado no primeiro piloto (versão alfa) a uma versão mais estruturada (versão beta, adotada no segundo piloto), até chegar à primeira versão que o laboratório considerou madura o suficiente para ser adotada em projetos comerciais (versão 1.0).

Em todas essas versões, o processo era documentado em uma ferramenta de modelagem de processos e publicado para a equipe envolvida. A partir do segundo piloto, a equipe de processos avaliava, a cada Sprint, quais as lições aprendidas no dia-a-dia do projeto e, quando consideradas aplicáveis, eram incorporadas à definição do processo.

Das práticas e eventos recomendados no Scrum e XP, a parte do Scrum foi a primeira a ser totalmente incorporada, o que ocorreu desde o primeiro piloto. Por corresponderem a eventos essenciais para a gestão do projeto, desde o início ficou claro aos envolvidos que não seria possível controlar o projeto sem eles.

Já as práticas do XP não foram implantadas em sua totalidade nos pilotos. A Tabela 1 mostra, das práticas XP encontradas na literatura [1] [2], aquelas que foram contempladas em cada versão do processo. Apesar de todos os envolvidos concordarem

[6] O menor produto, gerado no menor tempo possível, que possa agregar valor importante o suficiente para que seu principal cliente o adote.

com o benefício de tais práticas, a implantação de algumas trazia grandes impactos no fluxo de trabalho da equipe e por isso foi necessário adotar uma estratégia de implantação gradual. Na Tabela 1, a situação apresentada para as versões alfa e beta corresponde ao processo ao final de cada projeto. A versão 1.0 do processo incorpora todas as práticas, refletindo a expectativa de adotá-las integralmente nos projetos seguintes.

Prática	Versão			Prática	Versão		
	α	β	1.0		α	β	1.0
Liberações pequenas	✔	✔	✔	Semana de 40 horas	✔	✔	✔
Velocidade (Story Points)	-	✔	✔	Plano de liberações	-	-	✔
Programação em pares	✔	-	✔	Refatoramento	✔	✔	✔
Jogo do planejamento	✔	✔	✔	Cliente local	✔	✔	✔
Metáfora comum	✔	✔	✔	Espaço aberto	✔	✔	✔
Integração contínua	✔	✔	✔	Trocar pessoas com frequência	✔	✔	✔
Desenho simples	-	-	✔	Reunião diária em pé	✔	✔	✔
Propriedade coletiva do código	-	-	✔	Desenvolvimento dirigido por testes	-	-	✔
Regras adaptáveis	✔	✔	✔	Padrões de codificação	✔	✔	✔
Prototipação	✔	✔	✔	Otimizações para o fim	-	-	✔

Tabela 1. Práticas XP nas versões alfa (piloto 1), Beta (piloto 2) e 1.0 do processo

Um ponto a observar é que a prática da programação em pares foi adotada com sucesso no primeiro piloto, mas não no segundo. Houve resistência da equipe em mudar a forma de trabalho, por se tratar de colaboradores já habituados ao trabalho mais individual previsto no processo tradicional. Além disso, seus integrantes tinham horários de trabalho heterogêneos, o que dificultou ainda mais o trabalho em pares. Apesar disso, os benefícios observados com essa prática no piloto anterior foram claros o suficiente para que a decisão fosse por incorporá-la na versão 1.0 do processo.

Apesar das dificuldades enfrentadas e de não ter sido possível exercitar todas as práticas recomendadas pelo XP, os dois pilotos conseguiram demonstrar ao laboratório alguns benefícios dos métodos ágeis sobre a abordagem tradicional utilizada nos projetos desenvolvidos até então. Alguns dos benefícios observados foram:

- **Maior visibilidade sobre a situação e o progresso do projeto**, tanto para a equipe quanto para a gerência e alta direção, e consequente aumento da capacidade de tomar decisões e ações tempestivas para contornar problemas que surgiam.
- **Maior entusiasmo e comprometimento da equipe do projeto**. À medida que o processo funcionou melhor, a equipe ficou visivelmente mais comprometida.
- **Maior envolvimento dos *stakeholders*** estimulado pelo contato mais próximo com o projeto e com o produto nas entregas realizadas ao final de cada sprint.
- **Maior valor do produto final**, com menos esforço gasto em funcionalidades pouco relevantes e maior investimento nos requisitos-chave.

Não foi observado um aumento significativo de produtividade, mas o fato de ela não ter caído com o custo da mudança de processo já foi considerado um bom resultado.

5 Conclusão

A partir da experiência de desenvolvimento do Synergia-Ágil e sua aplicação nos primeiros projetos, uma conclusão clara foi a de que a transição para processos ágeis envolve uma mudança de paradigma na forma de conceber, desenvolver, controlar e entregar projetos de software. A aparente simplicidade dessas metodologias, consequência do minimalismo pregado por elas, pode ocultar a complexidade envolvida em sua implantação, mesmo em uma organização com uma cultura de forte de valorização do processo de desenvolvimento. O fato de que o projeto cuja equipe já estava habituada ao *modus operandi* do processo tradicional tenha enfrentado mais dificuldade em assumir o novo paradigma do que o projeto anterior, composto por uma equipe de profissionais iniciantes, ilustra o quanto essa transição pode ser difícil.

Apesar das dificuldades, os benefícios observados nos dois pilotos motivaram o laboratório a assumir, em seu último planejamento estratégico, a transição para o novo processo como uma das ações prioritárias. Podemos afirmar, portanto, que os dois pilotos obtiveram sucesso em seu objetivo maior de avaliar o novo paradigma e fornecer insumos para que o laboratório decidisse sobre sua adoção definitiva.

No momento em que este artigo foi produzido, três projetos em andamento já adotavam a versão 1.0 do Synergia-Ágil. Um deles era um projeto interno e outros dois já aplicavam o novo processo em projetos comerciais: o desenvolvimento de um novo produto para um órgão do Governo Federal e a manutenção evolutiva de um software de grande porte do Governo do estado de Minas Gerais. A experiência com esses novos projetos certamente trará melhorias para serem incorporadas ao processo.

Referências

1. PAULA FILHO, Wilson de Pádua. Engenharia de Software: fundamentos, métodos e padrões. 3. ed. Rio de Janeiro : LTC, 2009.
2. Mike Cohn. 2009. Succeeding with Agile: Software Development Using Scrum (1st ed.). Addison-Wesley Professional.
3. Kent Beck. 1999. Extreme Programming Explained: Embrace Change. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.
4. B. Fitzgerald, G. Hartnett, and K. Conboy, “Customising agile methods to software practices at intel shannon,” EJIS, vol. 15, no. 2, pp. 200–213, 2006
5. Fabri, J.A.; L'Erario, A.; Begosso, L.C.; Begosso, L.R.; de Lima, F.C., "Implementation of Software Residency at a graduation course", *Frontiers in Education Conference (FIE), 2010 IEEE* , vol., no., pp.F1H-1,F1H-6, 27-30 Oct. 2010
6. Trotskovsky, E.; Sabag, N., "Internship in Engineering Design in Hi-Tech Industries: Theory and Practice," *Transforming Engineering Education: Creating Interdisciplinary Skills for Complex Global Environments, 2010 IEEE* , vol., no., pp.1,16, 6-9 April 2010
7. Cohn, M.; Agile Estimation and Planning: AddisonWesley, 2005.
8. Ries, E.; The Lean Startup: How Today’s Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses; Crown Publishing Group, 2011.
9. VERSIONONE; 7th Annual State of Agile Development Survey, Versionone.com,2013
10. Melo, Claudia de O.; Santos, Viviane A.; Corbucci, Hugo; Katayama, Eduardo; Goldman, Alfredo; Kon , Fabio; “Métodos ágeis no Brasil: estado da prática em times e organizações”; RT-MAC-2012-03. Dep. de Ciência da Computação. IME-USP. Maio, 2012