Scrum Solo

Processo de software para desenvolvimento individual

Scrum solo

Software process for individual development

Tiago Pagotto
Laboratório de Inovação - LABINOV
Universidade Tecnológica Federal do Paraná - UTFPR
Cornélio Procópio, Brasil
pagotto@alunos.utfpr.edu.br

José Augusto Fabri, Alexandre L´Erario e José Antonio Gonçalves
Laboratório de Inovação - LABINOV
Universidade Tecnológica Federal do Paraná - UTFPR
Cornélio Procópio, Brasil
{fabri,alerario}@utfpr.edu.br

Resumo – **No Brasil existem muitas microempresas de software com um único desenvolvedor (desenvolvedor solo). Em pesquisa realizada pela Secretaria de Política de Informática do Ministério de Ciência e Tecnologia foi apontado que cerca de 60% das empresas de software consolidadas no mercado, iniciaram suas atividades com somente um desenvolvedor. Em sua maioria, estes desenvolvedores individuais não utilizam, formalmente, um processo de software. Neste ponto, este trabalho tem como objetivo apresentar um processo que possa ser usado de maneira formal por estes desenvolvedores solo – o Scrum Solo. Inicialmente, o procedimento definido foi experimentado no desenvolvimento de software para o Laboratório de Inovação da Universidade Tecnológica Federal do Paraná. Posteriormente, vários alunos desfrutaram do Scrum Solo para a elaboração de seus trabalhos de conclusão de curso. Atualmente, muitas empresas que trabalham com o desenvolvimento individual se beneficiam do processo retratado neste documento.**

Palavras Chave – Metodologia Ágil; Scrum; Personal Software Process (PSP); Scrum Solo.

Abstract — **In Brazil there are several small software companies with only one developer (solo developer). In a survey conducted by Secretaria de Política de Informática of Ministério de Ciência e Tecnologia has been indicated that about 60% of market consolidated software companies started their activities with a single developer. The majority of these developers does not formally make use of a software process. At this point, this work aims to present a formal software process for solo developers – the Scrum Solo. Initially, the defined procedure was experienced in software development for the Laboratório de Inovação of Universidade Tecnológica Federal do Paraná. Afterwards, several students made use of Scrum Solo to elaborate their completion of course works. Nowadays, plenty software companies working with individual development benefit from the process described in this document.**

Keywords: Agile; Scrum; Personal Software Process (PSP); Scrum Solo.

I. INTRODUÇÃO

No Brasil, atualmente existem muitos desenvolvedores de software que trabalham individualmente (solo). Estes representam cerca de 10% da força produtiva do setor de software (fonte: ministério do planejamento do Brasil). Aliado a este número, o Brasil forma anualmente cerca de 40 mil profissionais na área de Tencologia da Informação (fonte: ministério da educação do Brasil), estes profissionais carecem de um processo de produção para o desenvolvimento de seus trabalhos de conclusão de curso.

O Scrum Solo, criado em 2012 pelos autores deste trabalho, se caracteriza como um processo iterativo e incremental que une as boas práticas delineadas pelo Personal Software Process (PSP) e pelo Scrum. O processo proposto neste trabalho vem suprir a ausência de um processo para desenvolvedores solo, fato este apontado no parágrafo anterior.

O objetivo deste artigo é justamente descrever o processo e apresentar sua aplicação no desenvolvimento de parte de um pequeno projeto de software.

Adicionalmente, o processo proposto neste trabalho é amplamente utilizando pelos alunos dos cursos de Engenharia da Computação e Tecnologia em Análise e Desenvolvimento de Sistemas da UTFPR - campus Cornélio Procópio. Além destes, no Paraná, muitas empresas que possuem um único desenvolvedor também utilizam o Scrum Solo.

Para atingir o objetivo traçado, o trabalho foi segmentado da seguinte maneira: as seções II e III apresentam uma visão geral do Scrum e do PSP; já a IV descreve o Scrum Solo; na V é retratada uma aplicação do Scrum Solo em parte de um pequeno projeto de software; e, por fim, as seções VI e VII abordam, respectivamente, os resultados e as considerações finais.

II. SCRUM

O Scrum é um *framework* ágil (vide Figura 1) para gerenciamento de projetos que se destaca por sua abordagem enxuta (*lean*) de desenvolvimento [2]. Inicialmente, ele foi utilizado em empresas de fabricação de automóveis e produtos de consumo. Essas empresas notaram que projetos usando equipes pequenas e multidisciplinares (*crossfunctional*) produziram os melhores resultados, e associaram estas equipes altamente eficazes a formação Scrum Rugby (utilizada para

reinicio do jogo em certos casos). Jeff Sutherland, John Scumniotales e Jeff MacKenna documentaram, criaram e implementaram o Scrum na empresa Easel Corporation em 1993, incorporando estilos de gerenciamento observados por Takeuchi e Nonaka [3]. Em 1995, Ken Schwaber formalizou a definição de Scrum e ajudou a implantá-lo no desenvolvimento de software em todo mundo [4].

Por ser um modelo iterativo e incremental, o Scrum divide o projeto em vários *sprints* (ciclos curtos de desenvolvimento) consecutivos que ocorrerão de acordo com a prioridade do *product owner* (proprietário do produto). Cada período de *sprint* é definido, geralmente, entre duas e quatro semanas. Durante esse tempo, o *scrum team* (analista e programadores) se dedica ao máximo para ter um pequeno conjunto de funcionalidades codificadas e testadas [4].

A primeira atividade de cada *sprint* é uma reunião de planejamento (*daily scrum meetings*) do mesmo, na qual *product owner* e *scrum team* conversam sobre as prioridades do *product backlog* (lista de requisitos ou funcionalidades a serem desenvolvidas no software) e, assim, definem o *sprint backlog* (lista de requisitos ou funcionalidades que serão implementadas na *sprint*). Ao finalizar um *sprint*, é realizada outra reunião na qual ocorre a revisão e a demonstração das funcionalidades produzidas; com isso, obtem-se um *feedback* do *product owner*, o que pode levar a alterações nas funcionalidades recém-entregues e incrementações no *product backlog* [4].

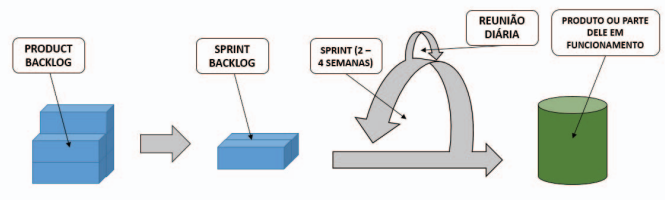

Figura 1. Fluxo do Scrum

Esta seção apresentou uma visão geral do Scrum, a próxima irá caracterizar as boas práticas do *Personal Software Process*.

III. *Personal Software Process* (PSP)

O PSP é um processo de melhoria projetado para ajudar os desenvolvedores a controlar, administrar e aperfeiçoar sua competência para produzir software de qualidade. O propósito do PSP é ajudar o desenvolvedor a melhorar a sua forma de trabalho, entendendo sua própria performance e sabendo onde e como melhorá-la. A filosofia por trás do PSP é que a competência de uma organização para construir softwares de determinado tamanho e grau de complexidade decorre, em parte, da habilidade individual de seus engenheiros. O PSP se baseia no princípio do conhecimento, avaliação e melhorias contínuas do processo individual [3].

Este processo define cinco atividades distintas, são elas [6] [7]:

A. **Planejamento**: isolamento dos requisitos e, baseando-se neles, os desenvolvedores estimam o tamanho, o tempo e os recursos necessários. Toda a métrica é registrada em planilhas ou gabaritos. Finalmente, as tarefas de desenvolvimento são identificadas e um cronograma de projeto é criado;

B. **Projeto de alto nível**: são desenvolvidas especificações externas para cada requisito a ser construído. Protótipos são construídos quando existe incerteza. Todo tempo e esforço para a construção do projeto em alto nível devem ser registrados e monitorados;

C. **Revisão do projeto de alto nível**: métodos de verificação formal são aplicados para descobrir possíveis erros no projeto. Vale ressaltar novamente, que todo tempo e esforço para a construção da revisão do projeto devem ser registrados e monitorados;

D. **Desenvolvimento**: o projeto é refinado e reavaliado. O código é gerado, revisado, compilado e testado. As métricas de registro e monitoramento do tempo e esforço também são mantidas no desenvolvimento;

E. **Pós conclusão**: usando as medidas e as métricas coletadas em todas as atividades anteriores, a efetividade do processo é determinada por meio de uma análise estatística. Essas medidas e métricas devem fornecer diretrizes para a modificação do processo de modo a aperfeiçoar sua efetividade.

Adicionalmente, deve-se ressaltar que o PSP é o resultado de uma adaptação do CMMI (Capability Maturity Model – Integration) para o nível individual e é dividido em sete níveis de maturidade, sendo estes [7]:

- **Nível 0 (PSP0)**: o desenvolvedor irá institucionalizar o seu processo de produção de software e registrar o tempo de desenvolvimento dos artefatos gerados.
- **Nível 0.1 (PSP0.1)**: o desenvolvedor adota um padrão de código e estabelece uma métrica de software.
- **Nível 1 (PSP1)**: o desenvolvedor utiliza as métricas para estimar o tamanho e a complexidade do projeto.
- **Nível 1.1 (PSP1.1)**: o desenvolvedor irá planejar e executar o teste utilizando o seu relatório utilizando, para isso, métricas e a base de experiência.
- **Nível 2 (PSP2)**: o desenvolvedor tem a possibilidade de promover a revisão do código e do *design*.
- **Nível 2.1 (PSP2.1)**: o desenvolvedor tem condições de construir seus guias e *templates*.
- **Nível 3 (PSP3)**: possibilita o refinamento cíclico do processo.

Feita a formalização do Scrum e do PSP, a seção seguinte trata a temática deste artigo, o framework Scrum aplicado à um único desenvolvedor.

IV. Scrum Solo

Como não é possível produzir softwares de maneira ad-hoc, o Scrum Solo [1] surge como uma customização do processo Scrum voltada para o desenvolvimento individual de software.

[1] O processo Scrum Solo pode ser visualizado na íntegra no seguinte endereço: http://scrumsolo.wordpress.com.

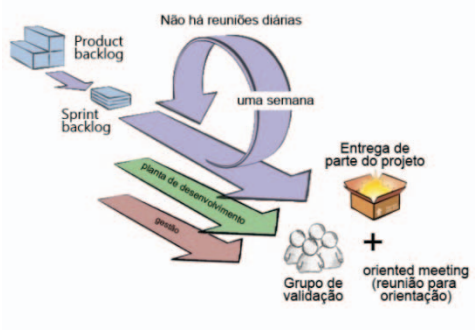

Figura 2. Fluxo do processo Scrum Solo

Ao analisar a Figura 2, é possível perceber que o Scrum Solo possui características semelhantes ao Scrum propriamente dito. O *product backlog* e o *sprint backlog* ocorrem de maneira idêntica em ambos os frameworks. No Scrum Solo, é proposto que os *sprints* tenham durações reduzidas a uma semana e não existam reuniões diárias; ao final de cada *sprint*, de forma equivalente ao Scrum, deve ser entregue, pelo desenvolvedor, um protótipo do software com novas funcionalidades e, podem existir, quando necessário, reuniões de orientação entre o grupo de validação (clientes e usuários finais) e o desenvolvedor [1].

É importante evidenciar também que o fato do Scrum Solo integrar as boas práticas do *framework* Scrum com as prerrogativas delineadas pelo PSP garante qualidade e agilidade na produção do software (vide figura 3) [6].

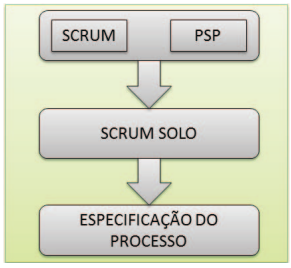

Figura 3. Escopo do Scrum Solo

A seguir, será descrita a especificação do Scrum Solo relacionando as atividades, atores e artefatos:

A. **Atividades do processo** [1] [6]

- ***Requeriment***: objetiva definir o escopo do produto, caracterizar o cliente do produto e definir a *product backlog*. As informações coletadas com o cliente e orientador (pessoa que possui grande conhecimento sobre o processo de software) são usadas como entradas e, os artefatos gerados nessa atividade são: escopo, *product backlog* e protótipo do software (vide Figura 4). Percebe-se que todos os artefatos gerados são armazenados em um repositório de dados. É interessante que este repositório permaneça em uma nuvem. Nota-se que o protótipo do software e o *product backlog* dessa atividade estão diretamente ligados ao projeto de alto nível (atividade do PSP – item B).

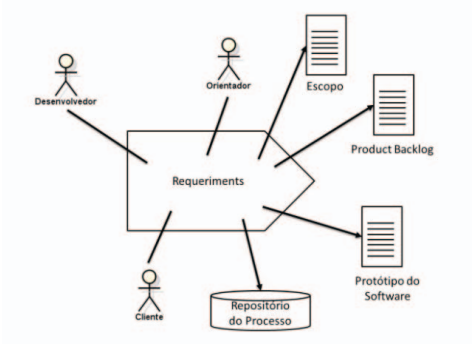

Figura 4. Atividade de Requisitos do Scrum Solo

- ***Sprint***: objetiva desenvolver o conjunto de itens selecionados a partir da *product backlog* em duração máxima de uma semana. O *product backlog* e o protótipo do software são usados como entrada e, os artefatos gerados nessa atividade são: *sprint backlog*, produto, ata e planta de desenvolvimento (vide Figura 5). Nota-se que o *sprint* e a ata dessa atividade estão diretamente ligados ao desenvolvimento (atividade do PSP – item D).

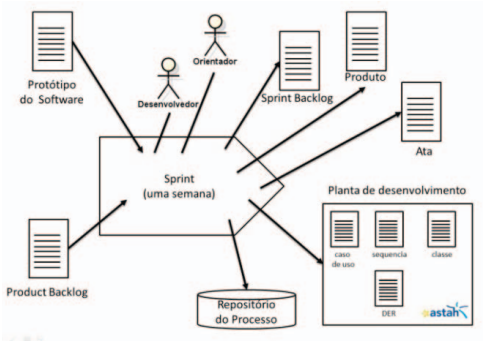

Figura 5. Atividade de *Sprint* do Scrum Solo

- ***Deployment***: objetiva disponibilizar o produto para uso do cliente. A planta de desenvolvimento é usada como entrada e, os artefatos gerados nessa atividade são: produto e ata de validação (vide Figura 6).

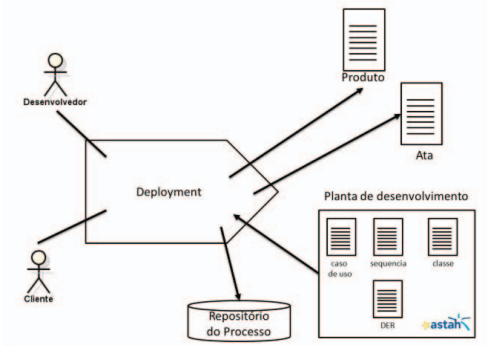

Figura 6. Atividade de Entrega do Scrum Solo

- ***Management***: objetiva planejar, monitorar e controlar o desenvolvimento do produto. O *product backlog* é usado como entrada e, os artefatos gerados nessa atividade são: estrutura

analítica do projeto (EAP), cronograma, planilha de custo e planilha de controle (vide Figura 7).

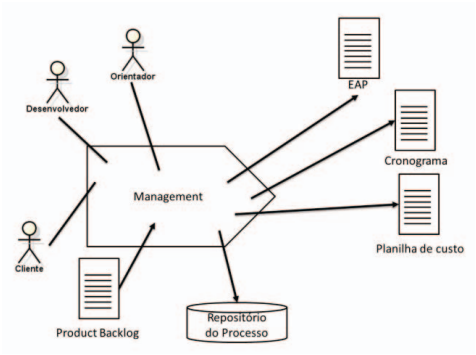

Figura 7. Atividade de Gestão do Scrum Solo

Nota-se que o cronograma e a planilha de custo dessa atividade estão diretamente ligados ao planejamento, à revisão do projeto de alto nível e à pós-conclusão (atividades do PSP – itens A, C e E, respectivamente).

B. **Atores do processo** [1]

- ***Product owner***: caracterizado como proprietário do produto. Ele irá interagir diretamente com o produto e pode ser caracterizado como uma única pessoa, por exemplo: gerente, contador, secretária; ou, como um grupo de usuários, por exemplo: pessoas que necessitam de um aplicativo para verificar a disponibilidade de ônibus em todo o país.
- **Desenvolvedor individual**: responsável por executar o processo e construir o produto.
- **Orientador**: caracterizado como um consultor que conhece a fundo o processo. Este possui uma visão ampla da tecnologia utilizada para o desenvolvimento do produto e do escopo do projeto.
- **Grupo de Validação**: possíveis usuários do produto gerado. Este deve participar da validação do produto, a qual deve constar na ata (artefato do processo).

C. **Artefatos do processo**

- **Escopo**: caracteriza o escopo do processo e os aspectos inerentes ao mapeamento dos problemas do *product owner*. Descreve os principais pontos do software (aplicados na solução de problemas), o perfil do cliente e os itens da *product backlog* (requisitos funcionais) – vide artefato na nuvem do processo[2].
- **Protótipo de Software**: coleciona as telas para acesso e manipulação de dados do software, além das interfaces dos relatórios. Deve-se ressaltar que todos os arquivos inseridos devem estar no formato .PNG e que é importante apontar o nome do item da *product backlog* que representa a tela ou a interface de relatório – vide artefato na nuvem do processo. Este artefato se relaciona diretamente com a atividade B do PSP – projeto de alto nível.
- ***Product backlog***: caracterizado como uma lista de funcionalidades que devem ser implementadas no software. Deve ser limitada ao código da funcionalidade, a descrição, a data da inserção e a data de seleção para a *sprint backlog*. Nota-se que a data da inserção só será preenchida quando a funcionalidade for selecionada – vide artefato na nuvem do processo.
- **Repositório do processo**: caracterizado como um serviço de armazenamento de arquivos na nuvem, este objetiva armazenar todos os artefatos gerados durante a execução do processo. Este repositório deve ser organizado em diretórios e o conjunto de regras para a organização dos diretórios é de responsabilidade do desenvolvedor.
- ***Sprint backlog***: armazena o conjunto de funcionalidades que devem ser implementadas durante aquele *sprint*. O *sprint backlog* armazena o código da funcionalidade (mesmo da *product backlog*) e o link para acesso a planta de especificação daquela funcionalidade. Esta planta é caracterizada pelos diagramas de: casos de uso, sequência, classes e entidade e relacionamento. É válido destacar que todos os diagramas devem ser armazenados no repositório do processo. O referido artefato possui data de inserção da funcionalidade no *sprint backlog*, tempo de construção (orçado ou previsto e realizado ou real), data de validação e, por fim, se a funcionalidade é fruto de um retrabalho – vide artefato na nuvem do processo.
- **Produto ou parte dele em funcionamento**: versão do produto que possibilite ao cliente obter um retorno sobre o investimento feito na compra do software.
- **Ata**: objetiva registrar a validação da implantação de uma funcionalidade. É utilizada no *sprint* e na entrega. No *sprint*, a funcionalidade é validada pelo orientador; já na entrega, a mesma é validada pelo cliente – vide artefato na nuvem do processo. Este artefato se relaciona diretametne com a atividade E do PSP – pós conclusão.
- **Planta de desenvolvimento**: objetiva aglutinar os artefatos que foram utilizados na especificação das funcionalidades. O Scrum Solo sugere que sejam utilizados os diagramas de: casos de uso, sequência, classes e entidade e relacionamento. Porém, de acordo com a especificidade do projeto, esta planta pode ser customizada. Uma ferramenta que possibilita a construção dos referidos diagramas é o Astah.
- **Estrutura Analítica do Projeto (EAP)**: organograma que objetiva apresentar o escopo do projeto. No topo deste organograma é encontrado o nome do projeto, logo abaixo, as atividades e, posteriormente os pacotes de trabalho. Estes últimos caracterizam formalmente todo o trabalho que será feito durante a execução do projeto – vide modelo na nuvem do processo. Este artefato se relaciona com a atividade A do PSP – planejamento.
- **Cronograma**: organiza sequencialmente os pacotes de trabalho dentro de um espaço de tempo determinado. Nele, pode-se apontar o responsável pela execução de

[2] A nuvem que contém os artefatos do processo pode ser acessada através do endereço: https://goo.gl/pf7NU2. A partir deste ponto, todos os artefatos apresentados e referenciados neste item podem ser encontrados na subpasta "TEMPLATES" com seus respectivos indicadores.

cada atividade. O Scrum Solo sugere que se utilize o cronograma no formato de diagrama de Gantt – vide exemplo na nuvem do processo. Este artefato se relaciona com a atividade A do PSP – planejamento.

- **Planilha de custo**: mapea o custo efetivo gerado durante a execução do projeto. Ao final, tem-se, de forma consistente, a visão de uma comparação entre o orçamento realizado previamente e os custos e tempos reais gastos no projeto – vide artefato na nuvem do processo. Este artefato se relaciona com a atividade A do PSP – planejamento.

V. APLICAÇÃO DO SCRUM SOLO

Nesta seção iremos aplicar o Scrum Solo no desenvolvimento de parte de um pequeno projeto de software. O principal cliente deste projeto é Laboratório de Inovação[3] da Universidade Tecnológica Federal do Paraná – campus Cornélio Procópio.

O software a ser construído com o apoio do Scrum Solo tem como objetivo criar um mecanismo de agendamento que pode ser feito a partir de qualquer dispositivo. O referido agendamento é feito por empresas constantemente atendidas pelo laboratório.

Inicia-se a construção do software pela atividade *Requeriments*. Conforme relatado, essa atividade requer a construção dos seguintes artefatos: Escopo, *Product Backlog* e Protótipo de Software (vide Figura 4). Estão envolvidos diretamente nesta atividade: desenvolvedor, orientador e *product owner*.

A execução da atividade de *Requeriment* requer uma interação direta entre desenvolvedor e cliente. O primeiro artefato gerado desta interação é o escopo do projeto. Ambos os itens, interação e escopo, podem ser visualizados na nuvem que contém o repositório do projeto[4] (local de armazenamento de todos os artefatos).

Ao analisar o artefato gerado é possível perceber claramente algumas funcionalidades ou requisitos do software (no Scrum Solo estes requisitos também são chamados de itens): cadastrar empresas; agendar atendimentos; realizar atendimentos; relatar atendimentos realizados; e, relatar atendimentos a realizar. Estes itens irão compor a *product backlog*, que pode ser visualizada no repositório deste projeto.

A *product backlog* possui os campos: *ID* do item; descrição (caracteriza o nome do item); data da inserção do requisito na *Product Backlog;* e, data da seleção deste requisito para compor a *sprint backlog*.

Após o preenchimento da primeira versão da p*roduct backlog*, é necessário delinear o primeiro protótipo do produto. Para realizá-lo é preciso interagir diretamente com o cliente. É sugerida a utilização do artefato "Protótipo do Software" mapeado anteriormente na nuvem e uma caneta comum para compor a primeira versão do documento. Através de um vídeo[5] pode ser visualizado como os autores deste trabalho realizaram a composição da interface do primeiro item da *product backlog* – Cadastrar Empresas.

Após a execução da atividade *Requeriment*, o Scrum Solo prevê que uma *Sprint* seja configurada. Paralelamente, deve-se iniciar a gestão do projeto por meio da execução da atividade de *Management*. Pode-se perceber esse paralelismo na Figura 2.

A execução da atividade *Management* irá delinear os seguintes artefatos: Estrutura Analítica do Projeto (EAP), Cronograma Inicial e Planilha de Custo. É indispensável que todos estes artefatos também sejam validados pelo cliente.

A Estrutura Analítica deste projeto também pode ser visualizada na nuvem. Nota-se que a EAP possui cinco pacotes de trabalho: cadastrar clientes; agendar atendimento; realizar atendimentos; relatar atendimentos realizados; e, relatar atendimentos a realizar. Estes pacotes são os mesmos documentados na *product backlog*.

De posse da EAP é necessário construir o cronograma de trabalho, para isto, como apresentado anteriormente, o Scrum Solo propõe ao desenvolvedor que ele utilize um diagrama de Gantt para representá-lo. O cronograma de trabalho também pode ser visualizado no repositório deste projeto.

Vale salientar que o cronograma desenvolvido deve ser consistente com a EAP (todos os pacotes de trabalho definidos no cronograma devem constar também na EAP). Observa-se também, que os pacotes de trabalho "cadastrar empresas" e "agendar atendimentos" serão desenvolvidos durante a primeira *sprint*, ou seja, na primeira semana. Este fato demonstra a consistência entre o cronograma e o Scrum Solo que prevê que as *sprints* devem durar uma semana.

O último artefato a ser construído na atividade *Management* (vide Figura 7) é a planilha de custo, a qual também deve retratar todos os itens que foram inseridos na *product backlog*, na EAP e no cronograma. A planilha de custo, apresentada no repositório deste projeto, possui o tempo e o custo orçado (em minutos) para os itens da primeira *sprint*. Ao final da *sprint*, após a reunião de orientação (realizada na presença do orientador), é necessário informar o tempo real destinado ao desenvolvimento do item da *product backlog*. Deve-se destacar que a unidade de tempo (minutos) configurada no Scrum Solo pode ser alterada de acordo com a necessidade do desenvolvedor.

É importante enfatizar que o Scrum Solo caracteriza-se pela iteratividade e incrementalidade, dentro deste contexto, os artefatos gerados pelas atividades *Requeriment* (executada sistematicamente ao longo do desenvolvimento do produto) e *Management* serão incrementados ou alterados constantemente. Estes devem ser atualizados e versionados no Repositório do Projeto.

---

[3] O Laboratório de Inovação (LABINOV) da Universidade Tecnológica Federal do Paraná, campus Cornélio Procópio, surgiu em 2014 com a missão de oportunizar o crescimento de empresas e projetos inovadores.

[4] A nuvem que contém o repositório do projeto pode ser acessada através do endereço: https://goo.gl/pf7NU2. A partir deste ponto, todos os artefatos produzidos e referenciados nesta aplicação podem ser encontrados e visualizados na subpasta "EXEMPLO DE APLICAÇÃO" com seus respectivos indicadores.

[5] Vídeo disponível na nuvem do processo na subpasta "EXEMPLO DE APLICAÇÃO" e também no endereço: https://youtu.be/N8xmN25vDO0.

Após a execução da atividade de *Managment*, será executada a s*print* (vide Figura 5). Para o software caracterizado neste trabalho foram gerados o seguintes artefatos: *Sprint Backlog*, Planta de Desenvolvimento, Produto ou parte dele em funcionamento e Ata de Validação. A maior parte destes artefatos referidos anteriormente podem ser encontrados na nuvem do projeto.

Ao analisar a *sprint backlog* é possivel verificar a presença dos seguintes itens: Cadastrar Empresas e Agendar Atendimentos. Estes itens fazem parte do cronograma apresentado na atividade de *Management*, provendo assim uma consistência entre os artefatos gerados. "Cadastrar Empresas" também foi especificado por meio da planta de desenvolvimento, o link desta especificação também está presente na s*print backlog*. A data de inserção e o tempo de construção (em minutos) orçado também fazem parte do escopo. Observa-se que os tempos orçados para os itens equivalem com os da planilha de custo gerada na atividade de *Management*. Precisa-se relevar que a data de validação do produto e o tempo realizado devem ser inseridos no artefato ao final da *sprint* (neste caso o tempo realizado é de 250 minutos e data de validação ocorreu no dia 06/01/2016). O campo "retrabalho" tem como objetivo denotar se o item gerado durante a *sprint* não foi validado e deve sofrer alguma alteração.

O Scrum Solo prevê que a planta de desenvolvimento do software seja desenvolvida com a ferramenta Astah [6], amplamente utilizada neste trabalho.

A Ata de Validação demonstra formalmente se o item foi aceito pelo orientador no final da *sprint.*

Por fim, a atividade de *Deployment* (vide Figura 6) tem como objetivo validar todo trabalho de desenvolvimento junto ao Grupo de Validação. Esta atividade é de suma importância, pois é por meio dela que o software entra em funcionamento.

VI. Resultados

Conforme relatado na introdução deste trabalho, o Scrum Solo foi criado em 2012. Vários desenvolvedores solo vêm utilizando o referido processo com êxito. Os resultados quantitativos e qualitativos do sucesso deste modelo são apresentados nos itens a seguir:

- 55 alunos dos cursos de Engenharia da Computação e Análise e Desenvolvimento de Sistemas da Universidade Tecnológica Federal do Paraná utilizaram o Scrum Solo nos anos de 2012, 2013 e 2014 e obtiveram sucesso no desenvolvimento de seus produtos.
- Um mapeamento efetuado pelos autores deste trabalho detectou 8 alunos e ex-alunos dos cursos de Análise e Desenvolvimento de Sistemas da Universidade Tecnológica Federal do Paraná que utilizam o Scrum Solo no desenvolvimento de produtos caracterizados como software. Estes alunos relatam que o processo apresentado neste artigo supre as necessidades eminentes na produção de software. Segundo os ex-alunos, o processo atende as expectativas inerentes a gestão de projetos do cliente e do desenvolvedor.

VII. Considerações Finais

O processo descrito neste trabalho une algumas práticas do *Personal Software Process* (PSP) ao Scrum. O PSP se caracteriza como um processo de software que apresenta o que deve ser feito para se ter um produto de qualidade. O Scrum se apresenta como um processo iterativo e incremental que tem como objetivo sistematizar o desenvolvimento de um sofware por meio da execução de uma série de *sprints*. A união destes dois processos geraram o Scrum Solo e, este último une as boas práticas apresentadas pelo PSP à sistematização produtiva garantida através do Scrum. O Scrum Solo transpõe a barreira do que deve ser feito e apresenta como fazer, fato este abordado na seção V.

Na seção V é perceptivel que os artefatos gerados com o Scrum Solo são armazenados em um repositório na nuvem. É crucial que estes itens armazenados sejam colecionados em uma página acessível ao cliente, permitindo sua visualização.

O Scrum Solo apresenta formalmente o Astah para automatizar a construção da planta do software. Os autores deste trabalho julgam que esta ferramenta reune todos os diagramas e artefatos necessários para a construção de um software..

Por fim, é importante realçar que o Scrum Solo encontra-se em constante evolução. Esta pode ser feita por qualquer desenvolvedor e reportada aos autores deste trabalho.

Agradecimentos

Os autores deste trabalho agradecem ao apoio, delineado para apresentação deste trabalho, da Fundação Araucária, da Secretaria de Estado da Ciência, Tecnologia e Ensino Superior do Paraná (SETI).

Referências Bibliográficas

[1] J. A. Fabri, A. L´Erario e T. Pagotto, "SCRUM SOLO", disponível em: https://www.scrumsolo.wordpress.com/, acessado em: 18 de fevereiro de 2016.

[2] M. Cohn , "MOUNTAIN GOAT SOFTWARE", disponível em: https://www.mountaingoatsoftware.com/agile/scrum, acessado em: 24 de fevereiro de 2016.

[3] J. Sutherland, A. Viktorov, J. Blount e N. Puntikov, "Distributed SCRUM: Agile Project Management with Outsourced Development Teams", Hawaii International Conference on Software Systems (HICSS'40), 2007.

[4] K. Schwaber, "Agile Project Development with SCRUM", capítulos 1-2, Microsoft Press, 2004.

[5] W. S. Humphrey, "PSP: a Self-Improvement Process for Software", Addison-Wesley Professional, 2005.

[6] R. Guoping, D. Shao e H. Zhang, "SCRUM-PSP: Embracing Process Agility and Discipline", 17th Asia Pacific Software Engineering Conference (APSEC'10), 2010.

[7] W. S. Humphrey, "The Personal Software Process (PSP)", Carnegie Mellon University, 2000.

[6] A ferramenta Astah está disponível no endereço: http://www.astah.net/.