

FACULDADE FORTIUM

ALINE PINHEIRO DANTAS

**DESENVOLVIMENTO DE SISTEMAS NA
ADMINISTRAÇÃO PÚBLICA: BENEFÍCIOS DO *SCRUM***

**BRASÍLIA-DF
2015**

ALINE PINHEIRO DANTAS

**DESENVOLVIMENTO DE SISTEMAS NA
ADMINISTRAÇÃO PÚBLICA: BENEFÍCIOS DO *SCRUM***

Trabalho de Conclusão de Curso apresentado ao curso de Pós-Graduação em Gestão de TI na Administração Pública da Faculdade Fortium, como requisito parcial para obtenção do título de Especialista em Gestão de TI na Administração Pública.

Orientador: MSc. César Augusto Borges de Andrade.

**BRASÍLIA-DF
2015**

ALINE PINHEIRO DANTAS

**DESENVOLVIMENTO DE SISTEMAS NA ADMINISTRAÇÃO PÚBLICA:
BENEFÍCIOS DO *SCRUM***

Trabalho de Conclusão de Curso apresentado ao curso de Pós-Graduação em Gestão de TI na Administração Pública da Faculdade Fortium, como requisito parcial para obtenção do título de Especialista em Gestão de TI na Administração Pública.

Aprovado em ____/____/____.

Banca Examinadora

Presidente: Prof. MSc. César Augusto Borges de Andrade - Faculdade Fortium
(Orientador)

Membro Titular: Prof. MSc. Rômulo Ferreira dos Santos - Faculdade Fortium

Membro Suplente: Prof. MSc. Gleyson Azevedo da Silva - Faculdade Fortium

**BRASÍLIA-DF
2015**

Desenvolvimento de Sistemas na Administração Pública: Benefícios do Scrum

Aline P. Dantas¹

¹Faculdade FORTIUM - Especialização em Gestão de TI na Administração Pública
SGAS Quadra 616 Módulo 114, Asa Sul - Brasília/DF - CEP 70.200-76

alinepinheiro20@gmail.com

Resumo. *Algumas metodologias de desenvolvimento de software com o passar do tempo adaptaram-se às necessidades das equipes e dos respectivos clientes. A partir das constantes mudanças de requisitos e das solicitações de entrega rápida dos softwares, surgiu a abordagem de desenvolvimento ágil. Essa abordagem tem o Scrum como método de gestão e planejamento de projetos de software e é indicado para locais em que é difícil planejar o futuro, como na Administração Pública. Este artigo apresenta quais os benefícios que podem ser obtidos com a utilização do Scrum em órgãos públicos.*

Palavras-chave: *Scrum, desenvolvimento de software, metodologia ágil*

Abstract. *Some software development methodologies over time adapted to the needs of the teams and their clients. From the ever-changing requirements and rapid delivery of software requests came the agile development approach. This approach has the Scrum method as management and planning software projects and is suitable for places where it is difficult to plan for the future, as in Public Administration. This article presents what benefits can be obtained with the use of Scrum in government agencies.*

Keywords: *Scrum, software development, agile methodology*

1. INTRODUÇÃO

As metodologias e as formas de desenvolvimento de sistemas sempre foram constantemente aprimoradas, de modo que os problemas enfrentados no passado se transformaram em lições aprendidas. Para Henriques (2014), os principais problemas eram: prazos esgotados, orçamentos extrapolados e usuários insatisfeitos, além dos conflitos entre os clientes e os analistas de sistemas. Na tentativa de amenizar esses problemas, em 2001, um grupo de dezessete especialistas em processos de desenvolvimento de software criou o Manifesto para Desenvolvimento Ágil de Software, ou apenas, Manifesto Ágil. O manifesto apresentou seus métodos para desenvolvimento ágil de *softwares* já aplicados por esses profissionais. Para garantir o sucesso dos projetos, esse grupo argumenta que devem ser respeitadas quatro premissas, onde os itens da esquerda são mais valorizados que os da direita de acordo com Beck et al. (2001):

1. Indivíduos e interação entre eles mais do que processos e ferramentas: o relacionamento entre as pessoas e sua capacidade de solucionar problemas são mais importantes que processos e ferramentas de trabalho, que por si só não produzem *softwares*;

2. *Software* em funcionamento mais do que documentação abrangente: melhor ter alguma parte de algo que é possível ver e utilizar, do que ter apenas a promessa de um grande produto em papel que leva muito tempo para se concretizar;
3. Colaboração com o cliente mais do que negociação de contratos: ter uma boa comunicação com o cliente rende melhores resultados do que apenas manter as solicitações do produto em um contrato;
4. Responder a mudanças mais do que seguir um plano: o escopo frequentemente tem mudanças do decorrer do projeto, pois o cliente amadurece as ideias durante o desenvolvimento do produto. O planejamento deve suportar alterações do escopo e ser útil para verificar os itens previstos e os que foram entregues.

Além dessas quatro premissas, Bek et al. (2001) informa que o Manifesto Ágil também se sustenta em 12 princípios:

1. A maior prioridade é satisfazer o cliente através da entrega contínua e adiantada de software com valor agregado;
2. Mudanças nos requisitos são bem-vindas, mesmo tardiamente no desenvolvimento. Processos ágeis tiram vantagem das mudanças visando vantagem competitiva para o cliente;
3. Entregar frequentemente software funcionando, de poucas semanas a poucos meses, com preferência à menor escala de tempo;
4. Pessoas de negócio e desenvolvedores devem trabalhar diariamente em conjunto por todo o projeto;
5. Construa projetos em torno de indivíduos motivados. Dê a eles o ambiente e o suporte necessário e confie neles para fazer o trabalho;
6. O método mais eficiente e eficaz de transmitir informações para e entre uma equipe de desenvolvimento é através de conversa face a face;
7. Software funcionando é a medida primária de progresso;
8. Os processos ágeis promovem desenvolvimento sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante indefinidamente;
9. Contínua atenção à excelência técnica e bom design aumenta a agilidade;
10. Simplicidade - a arte de maximizar a quantidade de trabalho não realizado - é essencial;
11. As melhores arquiteturas, requisitos e designs emergem de equipes auto-organizáveis;
12. Em intervalos regulares, a equipe reflete sobre como se tornar mais eficaz e então refina e ajusta seu comportamento de acordo.

As Metodologias Ágeis do manifesto possuem os mesmos ideais e muitas vezes são utilizadas em conjunto, adaptando-se à organização. Um dos métodos é o *Scrum*, utilizado para gerenciar projetos no desenvolvimento ágil de sistemas. Ele foi criado com a finalidade de melhorar e acelerar o processo de desenvolvimento de *softwares*.

O Scrum é um framework criado no início dos anos 90 e muito utilizado hoje para o desenvolvimento de software. O Scrum é dito como sendo um framework, pois ele não prescreve nada, não é um processo e nem mesmo uma técnica e é perfeitamente adaptável. No Scrum podemos incluir diversas técnicas utilizadas em outros processos, dessa forma podemos sempre melhorá-lo. (MEDEIROS, 2011)

Segundo Schwaber (2009, p. 3), o *Scrum* é fundamentado na teoria de controle de processos empíricos, ou seja, utiliza-se o que já é conhecido para a tomada de decisões. Três pilares sustentam a implementação de controle desses processos: a transparência, a inspeção e a adaptação. A transparência garante que aspectos do processo que afetam o resultado serão visíveis para quem os gerencia. A inspeção ocorre nos aspectos do processo para que variações inaceitáveis sejam detectadas. Já a adaptação é realizada sempre que aspectos do processo estiverem fora dos limites aceitáveis.

Os métodos ágeis favorecem a motivação dos colaboradores, bem como dependem de pessoas motivadas para realizar um trabalho mais efetivo, de acordo com Gabardo (2014, p.2). O *Scrum* prevê que a equipe de desenvolvedores deve resolver sozinha os problemas que surgirem durante o desenvolvimento de um *software*, por isso devem ser profissionais sempre motivados através de desafios, aprendizado contínuo e mérito.

Prates (2014) ressalta que o *Scrum* começou a conquistar espaço na Administração Pública somente após 2010, pois não conheciam a compatibilidade dos princípios do *Scrum* com os da Administração. Contudo, perdas de prazos para a entrega dos sistemas, orçamentos extrapolados, equipes desmotivadas e clientes insatisfeitos, fez com que essa comunidade procurasse outras metodologias. E foi assim que o *Scrum* começou a ser utilizado na área governamental brasileira.

O objetivo deste trabalho é expor as vantagens que a metodologia *Scrum* pode oferecer à Administração Pública durante o processo de desenvolvimento de sistemas. Para isso, esse artigo foi organizado da seguinte maneira: após a introdução, é apresentada a metodologia *Scrum* na seção 2: Apresentação do *Scrum*. São listadas as vantagens da utilização do *Scrum* em órgãos públicos na seção 3: Vantagens para a Administração Pública. A seção 4 mostra os desafios e riscos a serem enfrentados: Desafios e riscos na implementação do *Scrum* em entidades públicas. Por fim tem-se a conclusão obtida.

2. APRESENTAÇÃO DO SCRUM

“O *Scrum* é uma metodologia ágil para gestão e planejamento de projetos de *software*.” (COMUNIDADE DE DESENVOLVIMENTO ÁGIL DO BRASIL, 2013). Uma das principais características, é a divisão do processo em pequenos ciclos de desenvolvimento chamados *Sprint*. Os *Sprints* têm funcionalidades pré-definidas que após finalizadas são entregues aos clientes. A figura 1 representa a execução do *Scrum*: o *Product Backlog* (lista de funcionalidades do produto) é criado; depois a *Sprint* é iniciada com a reunião de planejamento, em que é criado o *Sprint Backlog* (lista de funcionalidades a serem implementadas na *Sprint*); a *Sprint* terá a duração de 2 a 4 semanas, com reuniões diárias no mesmo horário: a *Daily Scrum Meeting*; ao final da *Sprint* será entregue ao cliente uma parte do produto potencialmente utilizável.

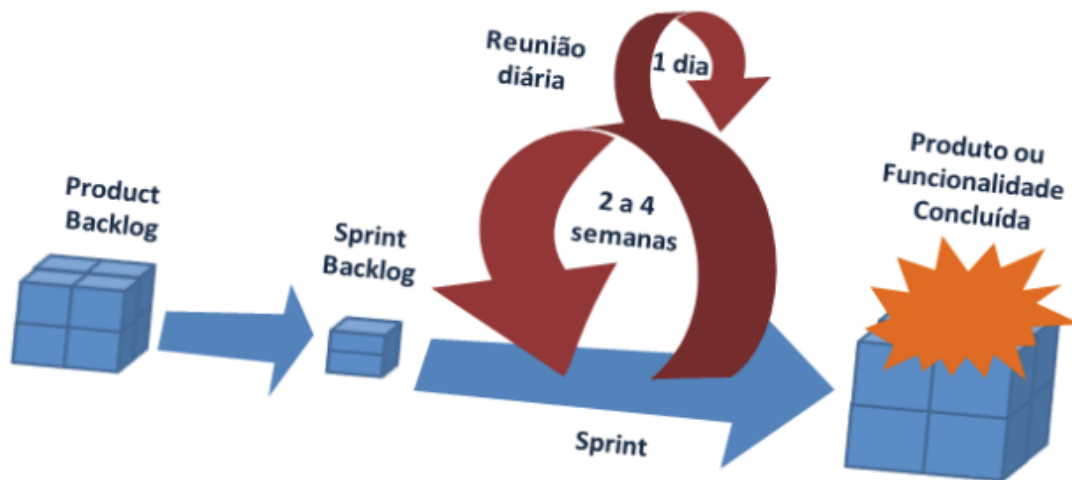


Figura 1. Práticas do Scrum

Fonte: (VIEIRA, 2014)

“O *framework Scrum* consiste em um conjunto formado por *Times Scrum* e seus papéis associados, Eventos com Duração Fixa (*Time-Boxes*), Artefatos e Regras.” (SCHWABER, 2009, p. 4)

2.1. PAPÉIS NO SCRUM

Cada *Time Scrum* possui três papéis: o *Scrum Master* (gerente do *Scrum*), o *Product Owner* (dono do produto) e a Equipe de Desenvolvimento.

“O *Scrum Master* é responsável por ajudar a todos os envolvidos a entender e abraçar os valores, princípios e práticas do *Scrum*.” (VIEIRA, 2014) Seguem suas responsabilidades:

1. Esclarece o andamento do projeto (contato/conexão);
2. É responsável por difundir os valores do *Scrum* e suas práticas;
3. Garante as reuniões de *Daily Scrum*;
4. Remove impedimentos;
5. Protege o time contra interferências externas;
6. Assegura que o time está totalmente funcional e produtivo; e
7. Possibilita uma cooperação estreita entre todos os papéis e funções.

“O *Product Owner* é a única pessoa responsável pelo gerenciamento do *Backlog* do Produto e por garantir o valor do trabalho realizado pelo Time.” (SCHWABER, 2009, p. 4) Ele mantém o *Backlog* do Produto visível a todos e deixa claro quais são os itens de maior prioridade, assim a equipe tem conhecimento em que se irá trabalhar. O *Product Owner* se compromete a não impor novos requisitos durante uma *Sprint*. Pode-se alterar requisitos, mas deve ser fora da *Sprint*, segundo a Comunidade Eclipse (2008). Seguem abaixo as responsabilidades do *Product Owner*:

1. Definir as funcionalidades do produto;
2. Decidir a data de liberação e conteúdo do *Release*;

3. Responder pela rentabilidade do produto (ROI);
4. Priorizar as funcionalidades de acordo com o valor de mercado;
5. Ajustar funcionalidades e prioridades a cada 30 dias, conforme necessário; e
6. Aceitar ou Rejeitar os resultados de trabalho.

A Equipe de Desenvolvimento realiza o trabalho necessário para transformar o *Product Backlog* em incrementos do produto potencialmente entregáveis ao final de cada *Sprint*. Organizadas de forma que o trabalho seja produtivo e flexível, as equipes são auto-organizáveis, interdisciplinares e trabalham em iterações. A própria equipe é que define como o trabalho será realizado e qual a melhor forma para alcançar o objetivo definido pelo *Product Owner*. Cada integrante terá conhecimentos em diferentes áreas para serem capazes de desenvolver o produto de acordo com os requisitos levantados. O conhecimento deve ser compartilhado por todos e continuamente aprimorado. Com isso, cria-se uma maior motivação da equipe, melhorando a eficiência e a eficácia. O trabalho em iterações deve-se ao uso das *Sprints*. O *Scrum* prevê que o número de pessoas na Equipe de Desenvolvimento deve ser de cinco a nove pessoas. Menos de cinco pessoas há uma menor interação, e com isso, menor produtividade. Mais de nove exige-se muita coordenação pois demanda muita complexidade.

2.2. EVENTOS DE DURAÇÃO FIXA

Para obter uma regularidade, o *Scrum* adota os Eventos com Duração Fixa. “Os Eventos com Duração Fixa (*Time-Boxes*) no *Scrum* são a Reunião de Planejamento da Versão para Entrega, a *Sprint*, a Reunião de Planejamento da *Sprint*, a Revisão da *Sprint*, a Retrospectiva da *Sprint* e a Reunião Diária.” (SCHWABER, 2009, p. 9)

A Reunião de Planejamento da Versão para Entrega cria um plano e metas em que a equipe *Scrum* e o resto da organização possam entender. Esse plano conterá a meta da versão, as prioridades do *Product Backlog*, os principais riscos, as características gerais e as funcionalidades da versão. Esse plano poderá ser verificado e alterado a cada *Sprint* pela organização, pois possivelmente as ideias serão amadurecidas e itens novos podem surgir, assim como itens definidos podem ser excluídos.

A *Sprint* é o evento mais importante do *Scrum*, tem duração fixa de até um mês (esse período servirá de base para mensurar os outros eventos) e realizada de forma iterativa até que o produto seja totalmente finalizado, conforme apresentado na figura 2. Cada *Sprint* tem como resultado uma parte do produto final potencialmente entregável, isto é, o incremento do produto deverá estar “pronto”. A definição de “pronto” deverá ser a mesma para todos envolvidos, pois para uns pode ser apenas um sistema codificado, para outros significa um sistema codificado, testado e homologado.

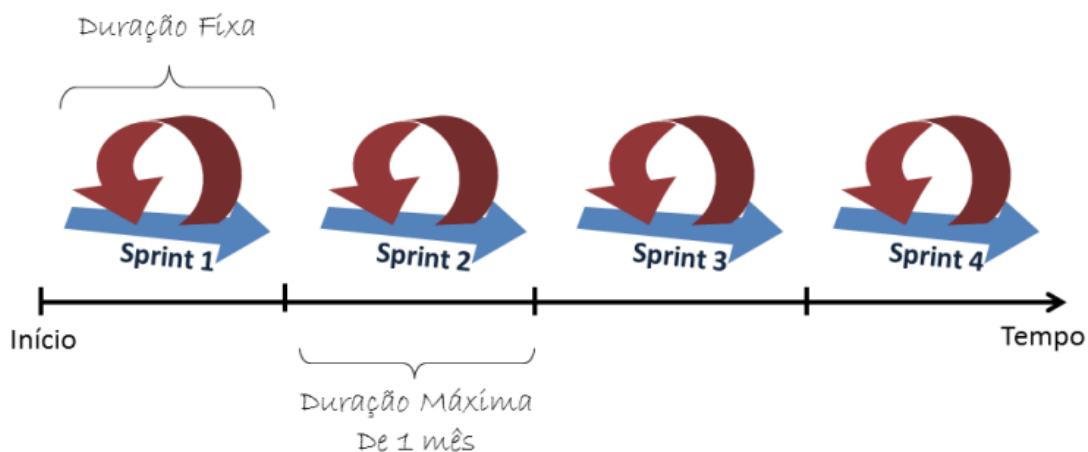


Figura 2. Sprints no decorrer do tempo

Fonte: (VIEIRA, 2014)

No início de cada *Sprint*, ocorre a Reunião de Planejamento da *Sprint*, onde serão definidas as atividades da iteração. Toda a equipe *Scrum* e interessados no desenvolvimento do produto participam dessa reunião, com duração de oito horas (proporcional ao período da *Sprint*). Na primeira metade da reunião, o *Product Owner* apresenta os itens prioritários do *Product Backlog* e a Equipe de Desenvolvimento seleciona os que ela se compromete a entregar, assim define-se o que será realizado na *Sprint*. Já na segunda metade da reunião, a Equipe deve projetar os itens selecionados de maneira a obter as tarefas necessárias para o desenvolvimento do incremento do produto. “Essas tarefas devem ser decompostas para que possam ser feitas em menos de um dia.” (SCHWABER, 2009, p. 9)

Ao final de cada *Sprint*, acontece a Revisão da *Sprint*, com a duração de quatro horas (proporcional ao período da *Sprint*). A Equipe de Desenvolvimento apresenta funcionalidade desenvolvida, o trabalho que foi realizado, os problemas encontrados e as soluções tomadas. Assim, essa revisão fornece uma importante entrada para as próximas *Sprints*.

Após a Revisão da *Sprint*, ocorre a Retrospectiva da *Sprint*. O *Scrum Master* incentiva a Equipe a revisar as práticas do *Scrum* que foram realizadas para serem melhor aproveitadas nas próximas *Sprints*. Com a duração de três horas (proporcional ao período da *Sprint*), essa reunião tem o objetivo de avaliar as relações pessoais, os processos e as ferramentas.

Durante a *Sprint* ocorre a Reunião Diária, a *Daily Scrum Meeting*, com toda a Equipe de Desenvolvimento, no mesmo local e horário. “Ela tem como objetivo disseminar conhecimento sobre o que foi feito no dia anterior, identificar impedimentos e priorizar o trabalho a ser realizado no dia que se inicia.” (COMUNIDADE DE DESENVOLVIMENTO ÁGIL DO BRASIL, 2013) Dessa forma, a equipe tem uma dimensão de tudo o que foi feito e o que ainda falta ser produzido. É uma reunião em que todos os integrantes da equipe assumem responsabilidades perante os demais.

2.3. ARTEFATOS

“Os artefatos do Scrum são projetados para fornecerem a transparência das informações necessárias para assegurar que o Time Scrum tenha sucesso na entrega do produto.” (VIEIRA,

2014) O *Scrum* utiliza quatro artefatos principais: o *Backlog* do Produto, o *Burndown* da Versão para Entrega, o *Backlog* da *Sprint* e o *Burndown* do *Backlog* da *Sprint*.

O *Backlog* do Produto é uma lista de requisitos do produto definida pelo *Product Owner* em conjunto com os demais interessados no produto. Os itens são priorizados por alguns fatores como valor, custo, conhecimento e risco. Os de maior valor são definidos no topo do *Backlog* do Produto, enquanto os de menor valor ficam direcionados para baixo. A figura 3 ilustra essa priorização das funcionalidades no *Backlog* do Produto. Esse *Backlog* é dinâmico, pois é alterado constantemente por mudanças nas regras de negócio ou por melhor compreensão da equipe *Scrum* sobre o produto.

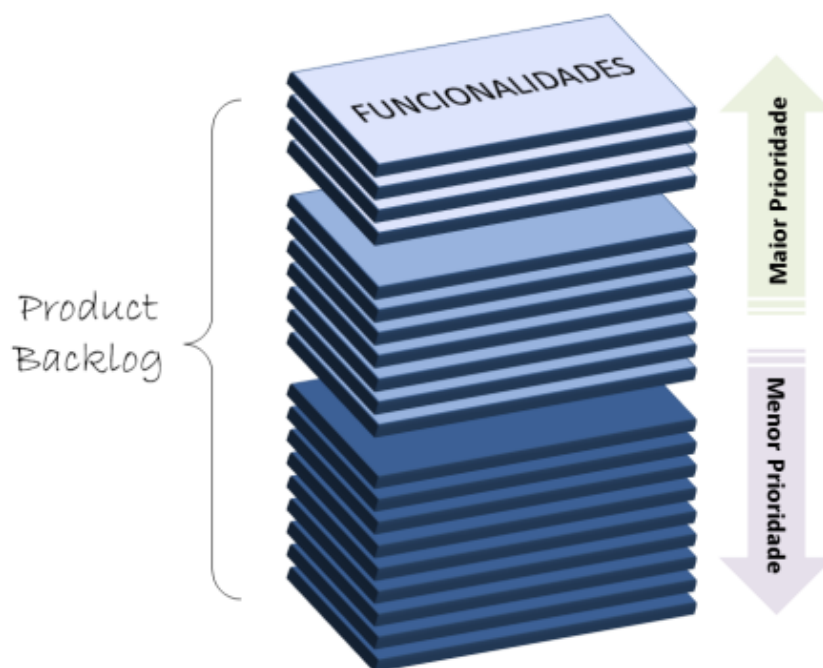


Figura 3. Priorização no Backlog de Produto

Fonte: (VIEIRA, 2014)

A figura 4 apresenta um exemplo de um projeto real. Cada item do *Backlog* do Produto tem uma priorização genérica (Muito Alta, Alta, etc.), a identificação, a descrição, a estimativa e o seu responsável. As estimativas são realizadas pela Equipe de Desenvolvimento, mas são muito imprecisas e somente são utilizadas para uma grosseira alocação de tarefas.

	Item #	Description	Est	By
Very High				
	1	Finish database versioning	16	KH
	2	Get rid of unneeded shared Java in database	8	KH
		- Add licensing	-	-
	3	Concurrent user licensing	16	TG
	4	Demo / Eval licensing	16	TG
		Analysis Manager		
	5	File formats we support are out of date	160	TG
	6	Round-trip Analyses	250	MC
High				
		- Enforce unique names	-	-
	7	In main application	24	KH
	8	In import	24	AM
		- Admin Program	-	-
	9	Delete users	4	JM
		- Analysis Manager	-	-
		When items are removed from an analysis, they should show up again in the pick list in lower 1/2 of the analysis tab	8	TG
	10	- Query	-	-
	11	Support for wildcards when searching	16	T&A
	12	Sorting of number attributes to handle negative numbers	16	T&A
	13	Horizontal scrolling	12	T&A
		- Population Genetics	-	-
	14	Frequency Manager	400	T&M
	15	Query Tool	400	T&M
	16	Additional Editors (which ones)	240	T&M
	17	Study Variable Manager	240	T&M
	18	Haplotypes	320	T&M
		- Add icons for v1.1 or 2.0	-	-
		- Pedigree Manager	-	-
	20	Validate Derived kindred	4	KH
Medium				
		- Explorer	-	-
		Launch tab synchronization (only show queries/analyses for logged in users)	8	T&A
	21			
	22	Delete settings (?)	4	T&A

Figura 4. Exemplo de Backlog do Produto
Fonte: (COMUNIDADE ECLIPSE, 2008)

Conforme Schwaber (2009, p. 18) o somatório das estimativas do trabalho restante do *Backlog do Produto* no decorrer do tempo é registrado em um gráfico chamado *Burndown* da Versão para Entrega. O trabalho restante é mensurado por unidades definidas previamente pela equipe *Scrum*. O tempo geralmente é medido em *Sprints*. A figura 5 apresenta um exemplo de um gráfico de *Burndown*, onde o eixo y é medido em pontos da *Sprint* e o eixo x são os dias da *Sprint*. São apresentadas três linhas, a azul é como a *Sprint* foi planejada, a vermelha é a *Sprint* atual e a verde é o ritmo diário.

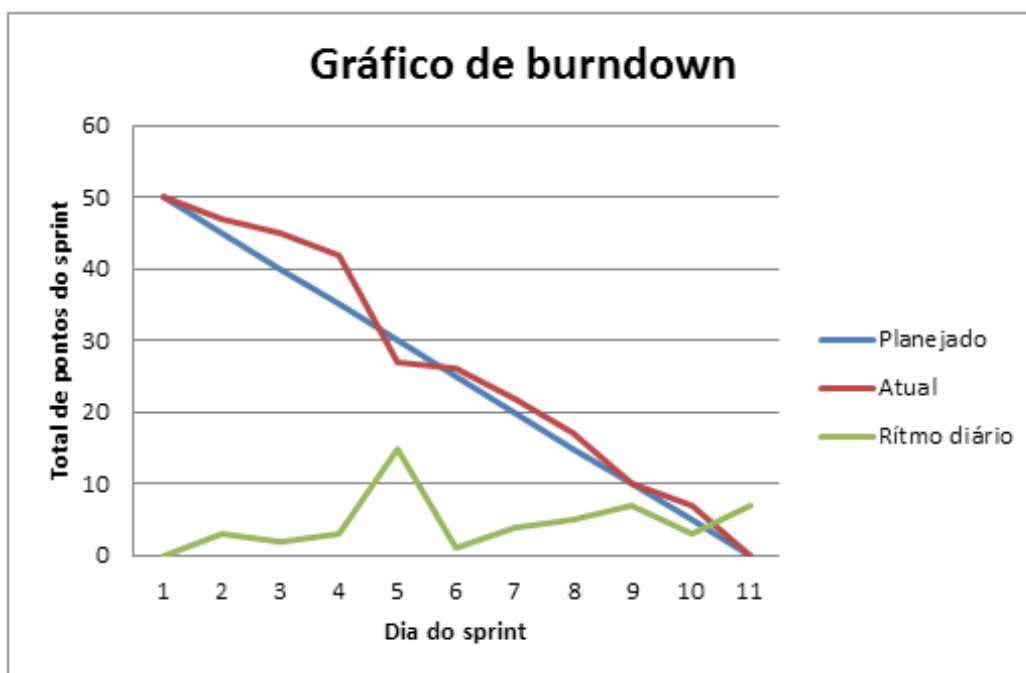


Figura 5. Gráfico de Burndown

Fonte: (CUNHA, 2011)

O *Backlog da Sprint* contém as tarefas que a equipe terá que desempenhar para que os itens do *Backlog do Produto* sejam transformados em um incremento “pronto”. Assim como o *Backlog* do Produto, o *Backlog da Sprint* também é dinâmico, pois durante uma *Sprint* pode ser que ele seja alterado para acompanhar as mudanças ocorridas. Tarefas podem surgir no meio do caminho e o tempo de duração das tarefas pode ser preciso aumentar ou diminuir.

O *Burndown* do *Backlog da Sprint* é um gráfico com a quantidade de trabalho restante do *Backlog da Sprint* ao longo do tempo. Esse gráfico é utilizado para a equipe monitorar o progresso diário do trabalho de uma *Sprint*.

As regras fazem o elo entre os eventos com duração fixa (*Time-Boxes*), os papéis e os artefatos do *Scrum*. Exemplo de regra do *Scrum*: em uma reunião diária só podem falar as pessoas que irão transformar o *Backlog* do Produto em um incremento potencialmente utilizável.

Além dos artefatos gerados do *Scrum*, existe uma ferramenta muito utilizada em organizações que adotaram as metodologias ágeis: o *Kanban*. *Kanban* é uma palavra japonesa e significa “cartão visual”. A adoção de *Scrum* pode ser útil para o início do uso *Kanban*. Ele é um quadro visível a todos e apresenta as tarefas a serem feitas, as que estão sendo trabalhadas e as que já foram finalizadas. De acordo com Ueda (2011), o *Kanban* contém princípios antigos e básicos para mostrar como está a situação do projeto em um olhar. Existem diversos modelos de *Kanban*, cada equipe personaliza o seu. O *Kanban* contém as seguintes práticas:

1. Visualizar o trabalho em andamento;
2. Limitar o trabalho em progresso;
3. Explicitar as políticas que estão sendo seguidas;
4. Medir e gerenciar o fluxo; e

5. Incentivar a melhoria contínua.

A figura 6 mostra um exemplo de *Kanban*, onde a primeira coluna apresenta os itens do *Backlog*, a segunda as tarefas a fazer, a terceira as tarefas em progresso e a última as tarefas feitas. Os itens são representados por *post-its* coloridos, onde cada cor representa um membro da Equipe de Desenvolvimento.















<i>Backlog Item</i>	A Fazer	Em Progresso	Feito
Motor	   	 	  
Chassis	  		
Porta	 		
Painel			

Figura 6. Exemplo de Kanban

Fonte: (CUNHA, 2011)

Com essa apresentação do funcionamento do *Scrum*, fica claro o que o projeto é trabalhado em pequenas iterações que são constantemente monitoradas e melhoradas. Cada iteração consiste nas atividades de captura de requisitos, análise, design, programação e testes, que finalizadas, entregam uma parte do produto ao cliente. A figura 7 apresenta um resumo de tudo que foi exposto sobre o *framework Scrum* na forma de um ciclo.

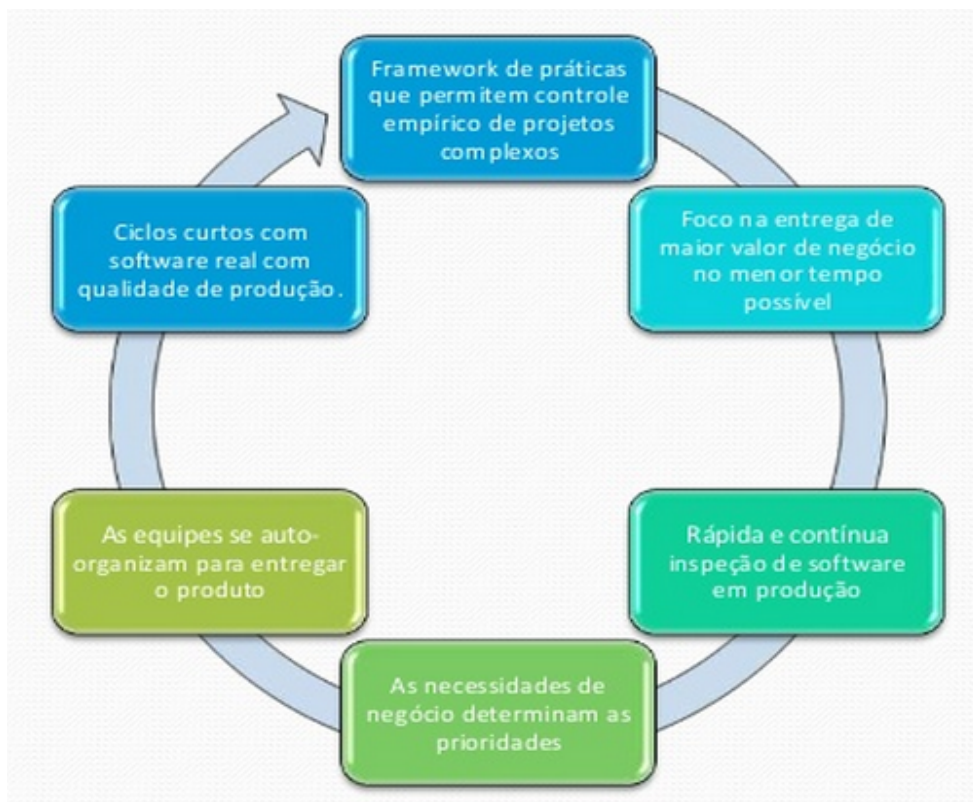


Figura 7. Ciclo do Scrum

Fonte: (MINDMASTER BRASIL, 2013)

3. VANTAGENS PARA A ADMINISTRAÇÃO PÚBLICA

O setor público tem muitos problemas no que diz respeito ao processo de desenvolvimento de *software*. Muitos desses problemas deve-se ao fato da adoção da metodologia tradicional para o desenvolvimento de *software*, principalmente em órgãos que utilizam contratos com fábricas de software (empresas contratadas através de licitações para desenvolver ou manter sistemas do órgão). Existe muita formalidade para o atendimento das solicitações relativas ao desenvolvimento dos sistemas nesses contratos, atrasando toda a real execução do trabalho. Por outro lado, a organização pode ter uma equipe própria de desenvolvedores, contendo apenas servidores públicos, que pode agilizar o processo de desenvolvimento. Entretanto se não houver regras, controles e processos pré-definidos, este cenário pode se tornar caótico, sem o mínimo de documentação e limitações quanto ao processo.

No primeiro caso são necessários muitos documentos para que efetivamente o desenvolvimento se inicie e, com a alta volatilidade dos requisitos na área pública, muitas vezes os documentos são alterados durante o trabalho do desenvolvedor. Isso gera retrabalho e aumento do cronograma. Já no segundo caso, o *software* é desenvolvido de forma desorganizada, sem o mínimo de documentação e planejamento, assim fica claro que o produto final não tem como ser condizente com o que o cliente deseja.

O *Scrum* apresenta diversas vantagens para a Administração Pública ao ser adotado para gerenciar o processo de desenvolvimento de *software* no lugar dos métodos tradicionais. A figura

8 apresenta os valores já mencionados dos métodos de desenvolvimento tradicionais e os do *Scrum*.



Figura 8. Valores dos Métodos Tradicionais e os do Scrum

Fonte: (MINDMASTER BRASIL, 2013)

A primeira vantagem estaria na definição clara de papéis do Time *Scrum*. O *Scrum Master* mantém o processo funcionando corretamente, solucionando o que surgir como empecilho para continuação do trabalho. O *Product Owner* é responsável por manter a comunicação constante entre os clientes e a Equipe de Desenvolvimento. Isso permite que a equipe não perca tempo com contatos repentinos dos clientes, o que acaba atrasando o desenvolvimento do produto. Outro ponto positivo é que o *Product Owner* não permite alterações de escopo na *Sprint* atual. Toda mudança é bem vinda, porém deverá ficar para as próximas iterações, ou seja, para as próximas *Sprints*. A Equipe de Desenvolvimento é incentivada a sempre buscar novos conhecimentos, pois no *Scrum* os membros devem ser interdisciplinares. Além disso, todos são mais comprometidos com o trabalho pois participam ativamente das atividades e cronogramas. Com isso, há um aumento na motivação da equipe, que sempre buscam cumprir as metas da *Sprint*.

Outro benefício para o setor público é a forma de planejamento do *Scrum*, composto pela reunião de planejamento da *Sprint*. Todos participam da reunião, realizada no início de cada *Sprint*. Primeiro são abordadas as funcionalidades de maior prioridade para o cliente, descritas pelo *Scrum Owner* e compreendidas pela Equipe de Desenvolvimento. O *Scrum* permite ainda que essa prioridade possa ser alterada pelo cliente, pois nem sempre ele tem definido o que é mais importante para o projeto. Essas funcionalidades serão as primeiras a serem desenvolvidas e consequentemente as primeiras a serem entregues ao cliente. A entrega rápida de pequenas partes do sistema solicitado gera uma satisfação maior do cliente. Finalizada uma *Sprint*, o cliente recebe um pedaço do *software* para homologação. Essa entrega faz com que o cliente sinta-se mais motivado a fazer parte do processo de construção do *software*, fortalecendo o apoio ao Time *Scrum*. Essa maior velocidade de entrega também evita que requisitos tornem-se obsoletos no meio do desenvolvimento como acontece muitas vezes no setor público. Na área

governamental ocorrem muitas alterações nas regras de negócio, pois elas dependem de legislações que sofrem um ritmo frequente de atualizações.

As funcionalidades são convertidas em tarefas técnicas pela Equipe de Desenvolvimento após a reunião de planejamento da *Sprint*, estas são repassadas para o *Sprint Backlog*. Durante a reunião, a Equipe de Desenvolvimento tem a autonomia para determinar o que será possível ou não executar na *Sprint*. Essa atribuição livra os desenvolvedores de serem alvos de imposição de prazos irreais de quem não domina o trabalho a ser produzido.

A melhor visualização do projeto por todos os envolvidos também é um outro ganho. Essa visualização é alcançada com as reuniões diárias dos desenvolvedores, com as reuniões ao final de cada *Sprint* e com os artefatos gerados. Nas reuniões diárias todos passam a conhecer todo o trabalho e assumem um compromisso perante os colegas do irá realizar. Os programadores sentem-se mais motivados a atingir as metas traçadas, têm uma visualização completa do projeto e passam a ter acesso a informações que nas metodologias tradicionais eram disponíveis apenas ao gerente de projetos. As reuniões ao final da *Sprint* apresentam o que foi alcançado, identifica o que funcionou bem, o que pode melhorar e o que pode ser feito para que essa melhora ocorra. Com isso, o processo é novamente verificado e continuamente melhorado com lições aprendidas. Os artefatos divulgam o andamento do projeto e ações proativas são tomadas afim de se evitar problemas maiores em caso de risco eminente. Esses artefatos devem ser visíveis por todos e atualizados diariamente caso necessário, gerando transparência ao processo de desenvolvimento.

Há também uma melhora na redução de erros no sistema, pois o *Scrum* prioriza a qualidade e a funcionalidade dos produtos, que são entregues somente após baterias de testes e ajustes dos erros encontrados.

Contudo, fica claro que o *Scrum* tem vários benefícios a oferecer para a Administração Pública, porém deve ficar claro que ele deve ser implantado em etapas e adaptado à política do ambiente.

4. DESAFIOS E RISCOS NA IMPLEMENTAÇÃO DO SCRUM EM ENTIDADES PÚBLICAS

Antes de mais nada, o *Scrum* deve ser aplicado na organização em pequenas partes, para não haver um choque de cultura, principalmente se o órgão adotar processos tradicionais. Deve-se avaliar o que será utilizado do *Scrum*, pois ele deve ser adaptado ao estilo de trabalho do local, assim não é necessário usar todo o *framework Scrum*, apenas o que for necessário de acordo com a cultura da entidade.

Para iniciar o uso do *Scrum*, recomenda-se escolher um projeto novo e importante, mas que não seja crítico. Projetos importantes geram o interesse das pessoas, porém os críticos não podem correr o risco de incertezas que podem vir com a nova metodologia. Selecionado o projeto deve-se definir o *Scrum Master* e o *Product Owner*. O *Scrum Master* deve ser uma pessoa que conheça o funcionamento do *Scrum*, pois será responsável por manter o processo rodando corretamente, por treinar a equipe e por resolver os impedimentos. Para o *Product Owner* será necessário sólido conhecimento dos requisitos do produto e também disponibilidade para a comunicação com a equipe e com os clientes.

Logo depois, deve-se realizar a reunião de planejamento da *Sprint* com os requisitos já definidos. Nessa reunião a Equipe de Desenvolvimento definirá as funcionalidades a serem atendidas na próxima *Sprint*. Será definido o prazo de entrega da *Sprint*, que será o menor possível afim de garantir a satisfação do cliente. A *Sprint* deve entregar um incremento do produto em que o cliente possa usar efetivamente. Qualquer alteração do cliente será repassada ao *Product Owner* que decidirá se aprova ou não, com isso o desenvolvedor fica protegido para continuar o trabalho dele.

Durante a *Sprint*, o *Scrum Master* manterá os eventos do *Scrum* em funcionamento com a equipe. Em especial, as reuniões diárias que nivelam os conhecimentos e relatam os problemas encontrados para que sejam solucionados. Além dos eventos, deve-se ter o mínimo de documentação para acompanhamento dos trabalhos, como o Gráfico de *Burndown* e o *KanBan*, que estarão visíveis a todos.

Com essa proposta de implantação, é importante frisar que a Administração Pública enfrentará desafios e possíveis riscos na alteração da metodologia de desenvolvimento de *software*. Existem dois cenários para uso do *Scrum* no desenvolvimento de *software* na setor público: apenas com servidores do órgão ou com servidores e empresas terceirizadas.

De acordo com Gabardo (2009), para usar o *Scrum* em equipes montadas somente com servidores do órgão público deve-se primeiramente investir na motivação das pessoas. O cargo de um servidor público tem suas atribuições definidas em lei, assim ele só pode fazer o que estiver escrito. Essa rigidez gera desmotivação, pois o crescimento profissional fica limitado. Uma saída para o servidor público insatisfeito com seu cargo seria fazer um novo concurso, mas como é difícil passar em um concurso público, o que acaba acontecendo é a acomodação no cargo atual, mesmo com a insatisfação. Uma dos limitadores é a falta de investimento em capacitação, o servidor depende de orçamento público aprovado em lei para realizar uma capacitação, que é um meio de motivar os profissionais. Outro agravante, são projetos descontinuados no meio da execução, seja por conta de políticas, ou por conta de falta de verba. São diversos fatores que podem levar o servidor público à insatisfação profissional, por isso para aplicar o *Scrum* neste cenário deve-se investir nas melhorias profissionais para estes servidores, de forma que a metodologia seja adotada com motivação e faça com que os projetos sejam finalizados com sucesso.

Com a utilização de métodos ágeis, as organizações atingem melhor eficiência e efetividade com melhores práticas para executar um trabalho. Assim aumentam o lucro para empresas privadas e economiza dinheiro público no caso de entidades públicas. A área pública permite remuneração variável de acordo com a avaliação de desempenho e o resultado. Essa avaliação será composta por critérios e fatores que reflitam a contribuição da equipe de trabalho para o cumprimento das metas intermediárias e globais do órgão ou entidade, e os resultados alcançados pela organização como um todo. Mas é sabido que a Administração Pública ainda encontra restrições em lei com relação à remuneração variável dos servidores públicos, já que para motivar uma equipe é importante o reconhecimento de um bom desempenho através de prêmios.

Diante do exposto, os servidores públicos que forem trabalhar com o *Scrum* precisam de políticas de remuneração, em que uma parte dela seja definida pela qualidade e resultado do trabalho desempenhado. Sem essa política definida, os servidores ficarão desmotivados em trabalhar no projeto em que o objetivo será o ganho de produtividade, pois eles não receberão nenhuma parte desse ganho.

Uma série de riscos podem prejudicar o andamento da implementação do *Scrum* na Administração Pública. O Acórdão 2314/2013 (BRASIL, 2013) realizou um estudo acerca da utilização de métodos ágeis na contratação para desenvolvimento de *software* pela Administração Pública Federal. Esse estudo foi feito através de pesquisas e visitas pela equipe de fiscalização do TCU a órgãos públicos que contrataram empresas para desenvolverem sistemas utilizando o *Scrum* como metodologia ágil de gestão de desenvolvimento de *software*, a partir disso foi elencada uma lista de riscos nesse tipo de contratação. A lista foi dividida em três grupos distintos: processos, pessoas e produtos.

4.1. RISCOS RELATIVOS A PROCESSOS

4.1.1. Contratação de desenvolvimento de *software* com adaptação de metodologia ágil que desvirtue sua essência

Caso alguma prática ou regra da metodologia ágil entre em conflito com normas e jurisprudências vigentes ou não atenda completamente à instituição contratante, pode ser que ela resolva adaptá-la à realidade dela. Essa prática pode ter as seguintes consequências:

- licitação com pouca competitividade, pois nem todas as empresas terão interesse em adaptar uma metodologia que elas já adotam com sucesso, colocando o negócio delas em risco;
- conflitos constantes com a contratada devido a interpretações erradas com relação à utilização dos processos, práticas e papéis definidos no instrumento convocatório;
- conflitos entre as pessoas da contratante por falha na atribuição de papéis;
- produtos entregues com baixa qualidade, caso a adaptação afete partes do processo que atendam a verificação da qualidade do produto;
- baixa eficiência e produtividade da equipe contratada, caso a adaptação elimine práticas que a torne eficiente.

4.1.2. Alteração da metodologia ágil adotada no instrumento convocatório no decorrer da execução contratual

Uma instituição inexperiente em métodos ágeis pode vir a alterar a metodologia selecionada durante a execução do contrato. Com isso, podem ocorrer problemas na área financeira, já que o valor definido pela empresa contratada pode aumentar devido aos custos que venham a surgir com as alterações.

4.1.3. Ausência de definição dos artefatos ou alteração dos artefatos exigidos da contratada no instrumento convocatório durante a execução contratual

Esse risco também pode ocorrer devido à inexperiência da instituição em métodos ágeis. Sem conhecer todos os produtos necessários para o desenvolvimento do software durante a licitação, a instituição pode deixar de solicitar alguns artefatos importantes. Dessa forma, os custos poderão ser elevados durante a execução do contrato e gerar conflitos com a empresa contratada.

4.1.4. Exigência de artefatos desnecessários ou que se tornam obsoletos rapidamente

Novamente pela inexperiência da instituição, principalmente na mudança de contratos de desenvolvimento de *software* tradicionais para contratos de metodologias ágeis, podem ocorrer solicitações de artefatos desnecessários. Esses artefatos tornam-se obsoletos rapidamente e oneram o ajuste contratual.

4.1.5. Utilização de contrato para desenvolvimento de software por metodologias tradicionais para desenvolvimento por métodos ágeis

Esse risco gera problemas entre a contratante e a fornecedora por conta da mudança do paradigma não previsto no instrumento convocatório. Muitos elementos podem não se adequar à utilização de métodos ágeis, como níveis de serviço e artefatos exigidos da contratada durante a execução contratual.

4.2. RISCOS RELATIVOS A PESSOAS

4.2.1. Falta de comprometimento ou colaboração insatisfatória do responsável indicado pela área de negócios (*Product Owner*) no desenvolvimento do software

Caso o *Product Owner* não se comprometa com suas atividades, o projeto poderá sofrer muitos problemas. As atividades do *Product Owner* são as relacionadas ao produto: definir a visão do produto, gerenciar suas funcionalidades, priorizá-las e, aprovar ou não os artefatos entregues em cada *Sprint*. Ele também deve estar disponível para dirimir possíveis dúvidas da Equipe de Desenvolvimento da empresa contratada. A ausência de comprometimento do *Product Owner* com o trabalho, pode gerar produtos sem a qualidade adequada ou que não atendam às necessidades dos clientes, atrasos no desenvolvimento e até o cancelamento do projeto.

4.2.2. Falta do conhecimento necessário do indicado pela área de negócios (*Product Owner*) para o desenvolvimento do software

A ausência do conhecimento do produto em sua totalidade pelo *Product Owner* pode levar a definições e priorizações inadequadas das funcionalidades do produto a ser desenvolvido. Com isso, será necessário mais trabalho para ajustar os produtos entregues, ou até mesmo refazê-lo desde o início, descartando todo o trabalho realizado, gerando mais custo para o contratante.

4.2.3. Excessiva dependência da visão do indicado pela área de negócios (*Product Owner*)

Se não houver interação do *Product Owner* com os futuros usuários do sistema em desenvolvimento, as expectativas destes podem não ser atendidas com relação ao *software* a ser construído. Outro problema causado pela excessiva dependência da visão do *Product Owner*, seriam os momentos de afastamento dele da instituição, como férias e licenças, em que o desenvolvimento do sistema poderia ficar suspenso, atrasando todo o processo.

4.2.4. Equipe da empresa contratada não ter expertise em desenvolvimento de *software* com métodos ágeis

A empresa contratada deve apresentar atestados de capacidade técnica exigidos na licitação que comprovem expertise em metodologias ágeis. E mesmo que estes atestados sejam verídicos, pode ser que os resultados na nova contratação não sejam os mesmos. Dessa forma podem haver atrasos e entregas de produtos com qualidade inferior ao solicitado.

4.2.5. Dificuldade de comunicação entre a equipe de desenvolvimento da contratada com o indicado pela área de negócios (*Product Owner*)

Além da disponibilidade do *Product Owner* para atender a equipe de desenvolvimento da empresa contratada, também deve existir comunicação eficaz entre eles. Um facilitador para atender esse requisito é alocar a equipe no mesmo local de trabalho da entidade contratante. A má comunicação pode gerar atrasos na entrega dos produtos.

4.3. RISCOS RELATIVOS A PRODUTOS

4.3.1. Alteração constante da lista de funcionalidades do produto

Durante o desenvolvimento de *software* as mudanças na lista de funcionalidades do produto são bem aceitas nas metodologias ágeis. Entretanto constantes alterações dessa lista realizadas de forma descontrolada podem levar a empresa contratada a extrapolar prazos e custos planejados no contrato.

4.3.2. Iniciação de novo ciclo sem que os produtos construídos na etapa anterior tenham sido validados

Com entregas de funcionalidades por ciclos, os métodos ágeis permitem que a equipe inicie um trabalho que tenha funcionalidades interdependentes das entregues no ciclo anterior. O problema ocorre quando o produto do ciclo anterior não é aprovado, assim as funcionalidades do ciclo atual ficam comprometidas, atrasando todo o projeto.

4.3.3. Falta de planejamento adequado do software a ser construído

O *Scrum* define o produto a ser desenvolvido através do *Backlog* do Produto. Tal artefato é produzido no início do projeto e é constantemente alterado de acordo com o amadurecimento dos requisitos do produtos no decorrer do desenvolvimento. Uma equipe inexperiente na metodologia ágil pode não planejar adequadamente o produto e comprometer a qualidade e os custos envolvidos para a criação do *software*.

4.3.4. Pagamento pelas mesmas funcionalidades do software mais de uma vez, em virtude de funcionalidades impossíveis de serem implementadas em um único ciclo, ou em virtude da alteração de funcionalidades ao longo do desenvolvimento do software

Funcionalidades do produto selecionadas para implementação podem não ser finalizadas durante a *Sprint*. Isso causa um grande problema, pois a funcionalidade inacabada deve ser alocada para outra *Sprint* e pode afetar outras funcionalidades que seriam implementadas após a conclusão desta. Ou pior, necessite criar outras funcionalidades e também alterar as já existentes. Mais uma vez, prazos e custos serão afetados.

4.3.5. Não disponibilização do software em ambiente de produção para a utilização e avaliação dos reais usuários

O *Scrum* entrega ao final de uma *Sprint* parte do produto em funcionamento no ambiente de produção para que os reais usuários possam validar as funcionalidades solicitadas e verificar a sua aderência às necessidades do negócio. Caso essa entrega seja postergada, o objetivo da metodologia não será atendido. Como consequência dos erros de concepção verificados tardiamente, os esforços e recursos utilizados serão desperdiçados.

4.3.6. Forma de pagamento não baseada em resultados

O ponto de função é a métrica comumente adotado nas instituições públicas em contratações para desenvolvimento de *software*, por determinações em lei. Ponto de função mede o tamanho funcional de *software*, não o esforço aplicado em sua construção. Isso gera críticas por parte das empresas contratadas, já que o ponto de função pode não ser compatível com o esforço e recursos utilizados para o desenvolvimento do produto.

Diante dos riscos apresentados, verifica-se que o órgão deve se planejar bem antes de realizar uma licitação para contratar uma empresa para atuar no desenvolvimento de *software* utilizando metodologia ágil. Deve-se verificar previamente se ambos têm maturidade suficiente para a implementação dos novos métodos de gestão de projetos propostos pelo *Scrum*.

5. Conclusão

Conforme exposto na introdução e na apresentação do *Scrum*, esta é a metodologia ágil utilizada para gerenciamento e planejamento de *software* com seus papéis, eventos, artefatos e regras. Ele tem como objetivo principal realizar entregas rápidas de partes do *software* através do desenvolvimento em ciclos. Essa metodologia conquista cada vez mais espaço na Administração Pública Brasileira, alguns órgãos já a utilizam, cada um com suas adaptações. A transição de uma metodologia tradicional para uma ágil deve ser de forma gradativa, os valores adotados são diferentes. Não é fácil implementar mudanças, todos devem se empenhar e estar motivados a aprender e a adotar novas práticas e regras. Mas se o *Scrum* for adaptado com a política local e implementado corretamente poderá trazer benefícios aos processos e aos resultados entregues.

Já nas vantagens para a Administração Pública foi apresentado o crescimento constante do número de demandas e de fracassos no cenário atual da TI das entidades públicas, impulsionou a necessidade de se conhecer e experimentar as metodologias ágeis. O projeto é entregue mais rápido e em pequenas partes funcionais com alto nível de qualidade; os papéis de cada membro da equipe são claros e objetivos; os desenvolvedores tem a liberdade de determinar o que conseguem realizar em uma *Sprint* e como será feito; e todos têm um processo transparente, com informações sobre o andamento do projeto reportadas diariamente.

Na apresentação de desafios e riscos na implementação do *Scrum* em entidades públicas abordou-se que os métodos ágeis utilizam a divisão dos ganhos com a Equipe de Desenvolvimento de acordo com o desempenho de cada membro. A Administração Pública terá o desafio de implementar uma política de remuneração por resultados para motivar a equipe e para que o *Scrum* possa se manter em pleno funcionamento. Também foi relatado sobre os riscos que podem ser enfrentados pela Administração na implementação do *Scrum* como uma nova metodologia. De acordo com um estudo do TCU, estes riscos foram encontrados em órgãos que contrataram empresas para o desenvolvimento de *softwares* utilizando o *Scrum* como metodologia. Os riscos foram classificados em processos, pessoas e produtos.

Esse trabalho teve como contribuição a demonstração simples e objetiva dos diversos benefícios que a implantação do *Scrum* pode oferecer aos órgãos públicos. Entretanto essas instituições devem estar preparadas para a mudança de cultura e para as adaptações da nova forma de trabalho com a política local. Além disso, devem planejar motivações para a equipe em formas de prêmios por desempenho e também se precaverem contra os possíveis riscos que podem surgir com a implementação da nova metodologia.

Referências

BEK, K. et al. **Manifesto for Agile Software Development**, 2001. Disponível em: <<http://www.agilemanifesto.org>>. Acesso em: 18 de maio de 2015.

BRASIL. TCU, Plenário. Acórdão 2314/2013. **Levantamento de Auditoria**. Disponível em: <<https://contas.tcu.gov.br/juris/Web/Juris/ConsultarTextual2/Index.faces?textoPesquisa=scrum>>. Acesso em: 18 de maio de 2015.

COMUNIDADE DE DESENVOLVIMENTO ÁGIL DO BRASIL **Guia Open Source sobre Desenvolvimento Ágil**, janeiro de 2013. Disponível em: <<http://www.desenvolvimentoagil.com.br/scrum/>>. Acesso em: 08 de maio de 2015.

COMUNIDADE ECLIPSE **Scrum Overview**, 2008. Disponível em: <<http://epf.eclipse.org/wikis/scrumpt/>>. Acesso em: 24 de maio de 2015.

CUNHA, D. **Metodologia de gerenciamento baseada em Scrum**, dezembro de 2011 Disponível em <<http://www.olharcritico.com/engenhariadesoftware/>>. Acesso em: 21 de maio de 2015.

GABARDO, M. A. et al. **Discussão sobre Motivação de Equipes na Implementação de Métodos Ágeis no Desenvolvimento de Sistemas na Administração Pública Federal**. 2009, 17 f. Universa – Universidade Católica de Brasília (UCB) Brasília – DF – Brasil. Disponível em: <<http://portal2.tcu.gov.br/portal/pls/portal/docs/1/2053980.pdf>>. Acesso em: 24 de maio de 2015.

HENRIQUES, R. **Uma Proposta de aplicação do Scrum no Departamento de Gestão da Informação do MDS**. 2014. 21 f. Trabalho de Conclusão de Curso (Especialização em Gestão de TI na Administração Pública) Faculdade Fortium Brasília – DF – Brasil.

MEDEIROS, H. **Introdução ao Scrum: Eventos e regras**. 21 de novembro de 2011. Disponível em: <<http://www.devmedia.com.br/introducao-ao-scrum-eventos-e-regras/29567/>>. Acesso em: 17 de maio de 2015.

MINDMASTER BRASIL **Fundamentos do Scrum**, 14 de outubro de 2013. Disponível em: <<http://pt.slideshare.net/MindMasterBrasil/scrum-master-fundamentos-certificacao>>. Acesso em: 24 de maio de 2015.

PRATES, I. **Metodologia Ágil (Scrum) no setor público**, 2014. Disponível em: <<http://mundogeo.com/blog/2014/08/10/metodologia-agil-scrum-no-setor-publico/>>. Acesso em: 17 de maio de 2015.

SCHWABER, K. **Guia do Scrum**. Scrum Alliance: transforming the world of work. Maio de 2009. 22 p.

UEDA, M., **Kanban – Planing Board: A Arte de Bater o Olho**, 27 de outubro de 2011. Disponível em <<http://www.tiespecialistas.com.br/2011/10/kanban-planning-board-a-arte-de-bater-o-olho/>>. Acesso em: 23 de maio de 2015.

VIEIRA, D. **Scrum: A Metodologia Ágil Explicada de uma forma Definitiva**, 26 de maio de 2014. Disponível em: <<http://www.mindmaster.com.br/scrum/>>. Acesso em: 19 de maio de 2015.