**UNIVERSIDADE FEDERAL DO RIO DE JANEIRO**

**ESCOLA POLITÉCNICA**

**DCC/NPPG**

**Estudo do impacto causado pelo modelo de contratação *outsourcing* na gestão de projetos *SCRUM* para software científico**

**André Luiz Antunes de Moraes**

**2015**

**Estudo do impacto causado pelo novo modelo de contratação outsourcing na gestão de projetos scrum para software científico**

**André Luiz Antunes de Moraes**

Monografia apresentada no curso de Pós-Graduação em Gestão de Portfólio, Programas e Projetos, da Escola Politécnica da Universidade Federal do Rio de Janeiro.

**Orientador**

Nome do Orientador

Rio de Janeiro

Setembro / 2016

**Estudo do impacto causado pelo modelo de contratação outsourcing na gestão de projetos SCRUM para software científico**

**André Luiz Antunes de Moraes**

**Orientador**

Nome do Orientador

Monografia submetida ao Curso de Pós-Graduação em Gestão e Gerenciamento de Projetos Gestão de Portfólio, Programas e Projetos, da Escola Politécnica, da Universidade Federal do Rio de Janeiro – UFRJ, como parte dos requisitos necessários à obtenção de título de Especialista em Gestão de Portfólio, Programas e Projetos

Aprovado por:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Nome do Prof. 1 , Titulação

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

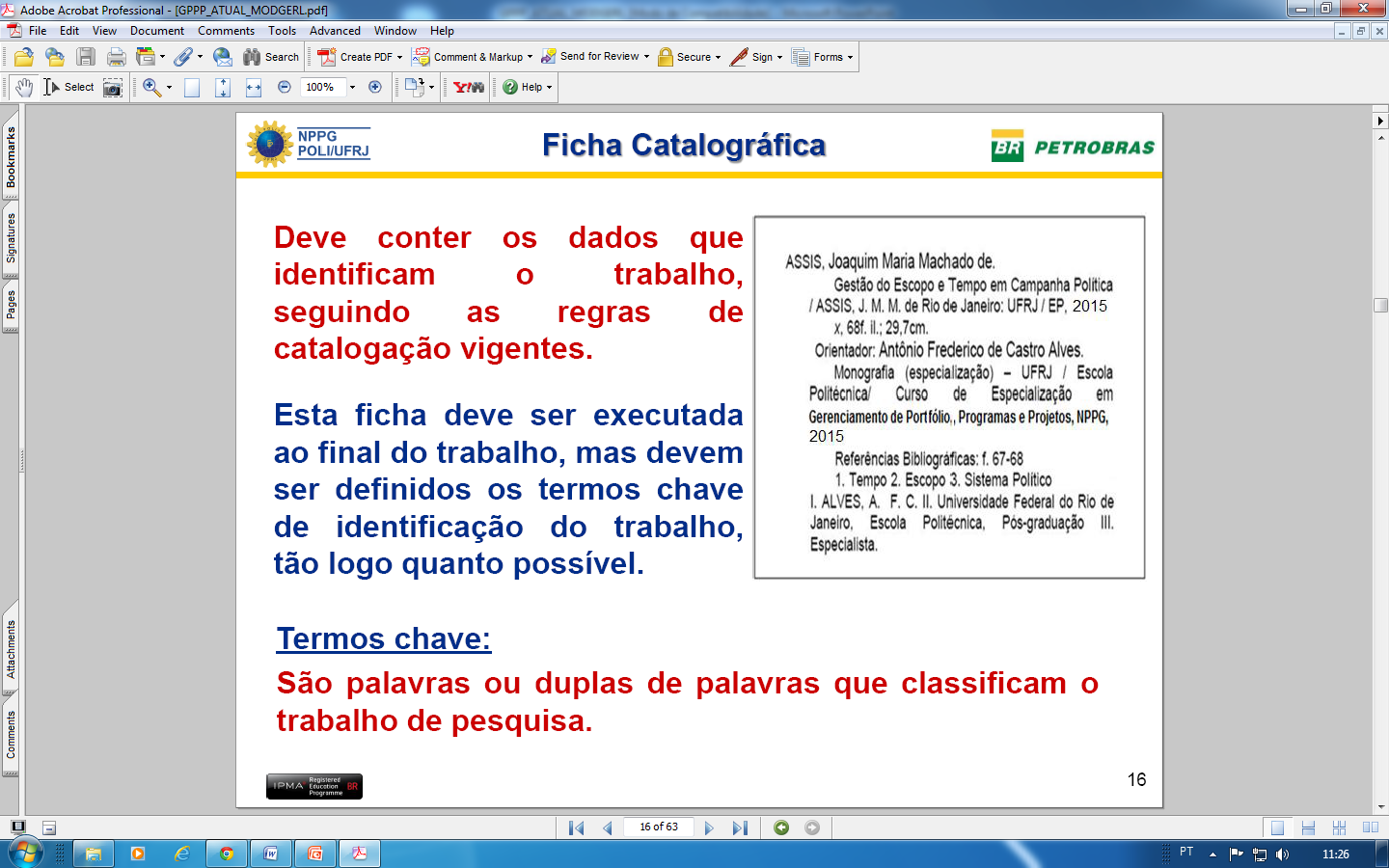
Nome do Prof. 2 , Titulação

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Nome do Prof. 3 , Titulação

Rio de Janeiro

Setembro / 2016



**RESUMO**

**Estudo do impacto causado pelo modelo de contratação outsourcing na gestão de projetos SCRUM para software científico**

**André Luiz Antunes de Moraes**

Resumo da Monografia submetida ao corpo docente do curso de Pós-Graduação em Gestão de Portfólio, Programas e Projetos – Universidade Federal do Rio de Janeiro – UFRJ, como parte dos requisitos necessários à obtenção de título de Especialista em Gestão de Portfólio, Programas e Projetos.

Resumo da Monografia: XXXXXXXXXX XXXXXXXXXXXXXXXX XXXXXXXXXX XXXXXXXXXXXXX XXXXX XXXXXXXXXX XXXXXX XXXXXXXXXXXXX XXXXXX XXXXXXXX XXXXXXXXXXXXXXX XXXXXX XXXXXX XXXXXXXXXX XXXXX XXXXXXX XXXX XXXXXXXXX XXXXX. (até 250 palavras)

Termos Chaves: SCRUM

Contrato Outsourcing

Software Científico

Rio de Janeiro

Setembro / 2016

**DEDICATÓRIA**

Dedico essa monografia a minha família por sua capacidade de acreditar em mim e investir no meu sucesso.

**AGRADECIMENTOS**

Agradeço а todos оs professores pоr mе proporcionar о conhecimento nãо apenas racional, mаs а manifestação dо caráter е afetividade dа educação nо processo dе formação profissional.

**SUMÁRIO**

1. INTRODUÇÃO 1

1.1. Contextualização 1

1.2. Objetivo do trabalho 1

1.3. Justificativa ao trabalho 1

1.4. Metodologia Empregada 3

1.5. Conteúdo dos capítulos 3

2. REVISÃO DA LITERATURA 5

Desenvolvimento de Software Cientifico 5

2.1.1. Definição 5

2.1.2. Particularidades 6

2.1.3. Construção de software científico 7

2.1.4. Riscos de um software científico 7

Metodologias de Desenvolvimento de Software: do tradicional ao ágil (SCRUM) 7

2.1.5. Definição 7

2.1.6. Metodologia Tradicional 7

2.1.7. Metodologia SCRUM 7

2.1.8. Comparando as metodologias 11

Metodologia PMI aplicado a SCRUM 11

2.1.9. Gerenciando projetos com PMI 11

2.1.10. PMI aplicado a metodologia de desenvolvimento SCRUM 11

2.1.11. Problemas e incompatibilidades 11

Modelos de contratação de serviços de desenvolvimento de software 11

2.1.12. Métodos de contratação segundo PMI 11

2.1.13. Contratação de serviços com fixo e escopo fechado 11

2.1.14. Métodos de contratação segundo a legislação brasileira 11

Padrões Petrobras 12

2.1.15. Métodos Ágeis 12

2.1.16. Métodos Ágeis aplicados a desenvolvimento científico 12

2.1.17. Modelo de Contratação de Serviços 12

3. SOFTWARE CIENTÍFICO X MÉTODOS ÁGEIS X CONTRATAÇÃO DE SERVIÇOS X GESTÃO DE PROJETOS 13

3.1. METODOS ÁGEIS APLICADO NAS GRANDES CORPORAÇÕES 13

3.1.1. Utilização 13

3.1.2. Problemas 13

3.1.3. Petrobras 13

3.2. NEGOCIANDO CONTRATOS COM MÉTODOS ÁGEIS 13

3.2.1. Utilização 13

3.2.2. Problemas 13

3.2.3. Petrobras 13

3.3. MÉTODOS ÁGEIS APLICADO EM SOFTWARE CIENTÍFICO 13

3.3.1. Utilização 13

3.3.2. Problemas 13

3.3.3. Petrobras 13

3.4. GESTÃO DE PROJETOS COM MÉTODOS ÁGEIS 14

3.4.1. Utilização 14

3.4.2. Benefícios 14

3.4.3. Problemas 14

3.4.4. Petrobras 14

4. ESTUDO DE CASO 15

4.1. Projeto RCDUT 15

4.2. Escopo Inicial 15

4.3. Escopo Final 15

4.4. Cronograma 15

4.5. Orçamento 15

4.6. Análises e Considerações no estudo realizado 15

5. CONSIDERAÇÕES FINAIS 16

5.1. Críticas e Comentários 16

5.2. Recomendações para Futuros Trabalhos. 16

REFERÊNCIAS 17

Referências Bibliográficas 17

Referências Normativas 18

Indicações Eletrônicas 19

**APENDICES**

**ANEXOS**

**LISTA DE FIGURAS**

Figura 1. Tendência da Sociedade do Conhecimento..............................................................1

Figura 2. Processos Essenciais da Gestão do Conhecimento...............................................10

Figura 3. XXXXX.....................................................................................................................15

**LISTA DE QUADROS**

Quadro 1. Indicadores de Projeto..........................................................................................5

Quadro 2. XXXXX................................................................................................................13

Quadro 3. XXXXX................................................................................................................19

**LISTA DE TABELAS**

Tabela 1. Desenho esquemático da Tripla Restrição...........................................................8

Tabela 2. Grupos de Processos Durante o Projeto............................................................20

Tabela 3. XXXXX................................................................................................................25

**LISTA DE SIGLAS E ABREVIATURAS**

**PMBoK** – Corpo de Conhecimento em Gerenciamento de Projetos (Project Management Body of Knowledge)

**EAP** – Estrutura Analítica de Projetos (Work Breakdown Structure)

**TAP** – Termo de Abertura de Projetos

**TCU** – Tribunal de Contas da União

**CENPES** – Centro de Pesquisas da Petrobras

**GLOSSÁRIO**

***EAP*** *– É uma estrutura de árvore exaustiva, hierárquica, orientada as entregas que precisam ser feitas para completar um projeto.*

***GOLD PLATING*** *– Refere-se a adicionar funcionários a um sistema que não foram solicitadas pelo usuário porque o desenvolvedor acha que o sistema fica melhor com as novas funcionalidades.*

***CENPES*** *– Centro de pesquisas da Petrobras*

# INTRODUÇÃO

## 1.1. Contextualização

A Petrobras ao longo dos últimos anos vem sofrendo pressão do TCU para uma mudança no formato de contratação de prestação de serviços com objetivo de diminuir a terceirização dentro da empresa e, nos casos em for realmente necessária, utilizar um novo formato de relação entre a empresa, colaborador e o objeto de prestação de serviços.

O CENPES, por sua vez, vive uma realidade de construção de projetos de software científicos que atendem à área de pesquisa nos seguimentos da área fim da companhia. Esses projetos, em sua grande maioria, são cercados por incertezas, complexidade e necessidade de conhecimento profundo dos assuntos tratados.

Ao longo dos últimos 8 anos ocorreu um amadurecimento no processo de construção dessa modalidade de *software*, saindo das tradicionais metodologias de desenvolvimento em cascata para desenvolvimento ágil em SCRUM. Os benefícios trazidos nas áreas de gestão de escopo, prazo, custo e qualidade desses projetos são fortemente reconhecidos pelo cliente.

## 1.2. Objetivo do trabalho

Apontar possíveis impactos causados pela mudança no formato de contratação para construção de softwares científicos nas áreas de gestão de escopo, custo, prazo e qualidade de um projeto. Para isso serão abordadas as características diferenciadas de um software científico, o uso da metodologia ágil SCRUM dentro de grandes corporações, o sucesso do uso dessa metodologia na construção desse tipo de software e finalmente uma abordagem que contrapõe o tipo de contratação proposto e a utilização dos métodos ágeis.

Apresentar um estudo de caso de um projeto de construção de um software cientifico desenvolvido na metodologia ágil SCRUM onde serão evidenciadas as constantes mudanças de escopo ao longo do projeto e a partir disso buscar na literatura problemas que podem ser gerados no modelo de contratação proposto.

## 1.3. Justificativa ao trabalho

Os valores de missão e visão de uma empresa são traduzidos ao longo de sua existência nos diversos planejamentos estratégicos que ela define. Estratégia essa que se desdobra em portfólios, programas e projetos cujo sucesso está associado à sua correta condução e entrega dentro da expectativa de custo, tempo, qualidade e principalmente na aderência do escopo definido pelos maiores interessados no sucesso do projeto.

KERZNER(2009) define o sucesso de um projeto de maneira mais ampla a utilizada no passado, quando se considerava apenas uma atividade concluída dentro de restrições de tempo, custo e desempenho; passando a definir o sucesso de um projeto como:

* Dentro do período de tempo alocado
* Dentro do custo orçado
* No desempenho ou especificação nível adequado
* Com a aceitação pelo cliente / usuário
* Com mínimo ou mutuamente acordados mudanças de escopo
* Sem perturbar o fluxo principal trabalho da organização
* Sem mudar a cultura corporativa

Dentro desse contexto, as instituições lidam com projetos diferenciados cujo escopo não é de tácito conhecimento pelos seus próprios idealizadores, algo que vem a ocorrer depois de alguns meses depois do seu início e muitas das vezes até momentos que antecedem o seu fim. Dentro dessa categoria encontram-se os projetos cujo o produto são softwares científico.

KELLY (2015) define software científico como uma aplicação de software que inclui um grande componente de conhecimento do domínio de aplicação científico e é usado para aumentar o conhecimento da ciência com o propósito de solucionar problemas do mundo real, utilizando a palavra “cientifica” para se referir a aplicações de engenharia. Ressalta também que o termo “software científico” tem sido usado por uma variedade de tipos de software que não compartilham os mesmos requerimentos de qualidade ou mesmas prioridades de gerenciamento dos softwares de commodities que são gerenciados para atender prazos de entrega e restrições orçamentárias.

Através de um processo quase que empírico, foram necessários anos para verificar que o trabalho realizado na construção de softwares científicos dentro do CENPES, utilizando o tradicional modelo *waterfall* de desenvolvimento de software e os padrões de gerenciamento de projeto definidos pela companhia não se aplicavam adequadamente aos tipos de projetos desenvolvidos no centro de pesquisa. Com o surgimento, amadurecimento e adoção das metodologias de desenvolvimento ágil nos projetos de software científicos, ficou evidente para o cliente o ganho em termos de velocidade e qualidade do produto entregue.

É importante ressaltar que esse ganho de produtividade no desenvolvimento do software cientifico vem não somente da adoção da metodologia ágil, mas também da oportuna configuração do modelo de trabalho dentro da unidade, que conta com profissionais experientes que ao longo dos anos se especializaram e acumularam conhecimento tanto no negócio da companhia quanto nos inúmeros padrões corporativos que regulam critérios de qualidade, confidencialidade, governança de gestão de projetos e outros.

Nesse contexto de trabalho, o novo modelo de contratação *outsourcing* que prevê a venda desses tipos de software como *commodities* através de pacote de serviços fechado põe em risco as conquistas construídas ao longo dos anos, a atual eficiência do tipo de trabalho desenvolvido pela gerência e consequentemente o sucesso dos projetos de *software* cientifico desenvolvidos no CENPES.

Esse trabalho busca na literatura como as corporações conseguem lidar com o desenvolvimento de software cientifico, o uso de metodologias ágeis para construção desse tipo de software, os formatos de contratação existentes e sua aplicabilidade a esse tipo de produto e a gestão desses tipos de projetos de acordo com o tipo de contrato de prestação de serviços. No final, pretende-se utilizar esse trabalho para auxiliar no mapeamento dos impactos gerados por esse modelo de contratação e buscar soluções para minimizá-los no que se refere a projetos envolvendo o desenvolvimento de softwares científicos.

## 1.4. Metodologia Empregada

A metodologia empregada para esse trabalho implicou em ampla pesquisa bibliográfica. Foram realizadas consultas em livros, artigos científicos, periódicos, sites da internet e padrões corporativos com o propósito de obter-se amplo entendimento sobre os assuntos abordados pelo tema e obter-se o estado da arte para lidar com os problemas expostos. Através do estudo de caso, utilizar um projeto realizado com sucesso e evidenciar os possíveis efeitos nocivos caso se optasse por o modelo de contratação proposto.

## 1.5. Conteúdo dos capítulos

Este trabalho está organizado em cinco capítulos, além desta introdução, descritos resumidamente a seguir:

No Capítulo 2 é realiza uma revisão da literatura apresentando os principais conceitos que serão abordados durante a discussão do tema: Desenvolvimento de Software Cientifico, Metodologias de Desenvolvimento de Software: do tradicional ao ágil (SCRUM), Metodologia PMI aplicado a SCRUM, Modelos de contratação de serviços de desenvolvimento de software e Padrões Petrobras.

No Capítulo 3, é realizada uma pesquisa bibliográfica sobre o estado da arte que envolve a relação do desenvolvimento de softwares científicos, a utilização dos métodos ágeis no desenvolvimento desses tipos de projeto, como é a relação dos métodos ágeis com diferentes tipos de contratação e principalmente com o tipo de contrato que define um pacote de serviços fechado e finalmente os desafios de gestão de projetos que envolvem a construção de softwares científicos.

No Capítulo 4, é apresentado um estudo de caso de um projeto desenvolvido utilizando o modelo atual de trabalho, suas características e seu sucesso diante o cliente; e posteriormente é feita uma projeção de sua construção, focado basicamente nas características peculiares de um software científico e os desafios que teriam quer considerados para obtenção dos mesmos resultados em um modelo de trabalho proposto pelo modelo atual de contratação.

No Capítulo 5, Considerações Finais, o trabalho é concluído com um resumo das conclusões dos capítulos anteriores.

# REVISÃO DA LITERATURA

## Desenvolvimento de Software Cientifico

### Definição

O software científico tem particularidades que diferencia seu processo de construção do software não-científico. Na grande maioria das vezes são desenvolvidos pelos próprios pesquisadores ou equipes muito especializadas para auxiliar no entendimento de problemas do mundo real cuja solução é desconhecida e envolve complexidade e incerteza. Bem diferente do software tradicional que lida com problemas e soluções conhecidas e podem ser construídos através de modelos tradicionais de engenharia de software e pode ser adotada práticas amplamente dominadas de gerenciamento de projeto.

PRESSMAN (2006) define o conceito de software como um conjunto de instruções (programas de computadores) que quando executadas fornecem uma função e um desempenho desejados, possui estruturas de dados que permitem aos programas manipularem adequadamente a informação e são dotados de documentação que descrevem a operação e o seu uso.

PURRI (2006) define software científico como um software desenvolvido por pesquisadores em seus projetos de pesquisa científica. A grande maioria destes projetos é de natureza acadêmica, ou seja, projetos de iniciação científica, mestrado, doutorado, pós-doutorado, dentre outros, que necessitam da construção de software para auxiliar estas pesquisas.

Felizmente nas últimas décadas várias indústrias (petróleo, aviação, comunicação, engenharia, farmacêutica, médica, etc.) viram seu negócio prosperar a partir do momento que começaram a investir na produção de conhecimento cientifico com foco em inovação, seja através de parcerias com as universidades, seja incentivando sua força de trabalho a produzir conhecimento. Dentro desse contexto de produção de conhecimento, os softwares científicos de destacam e passam a serem tratados como um diferencial estratégico e gozam do privilégio de ter altos orçamentos e importância máxima dentro da companhia.

HOWISON and HERBSLEB (2011) mencionam que o software tem desempenhado um papel cada vez mais importante na ciência, incluindo análise de informações, simulações e gerenciamento de *workflows*. Diferentemente de outras tecnologias que suportam a ciência, o software pode ser copiado e distribuído essencialmente sem custo, e potencialmente abrindo a porta para níveis sem precedentes de compartilhamento e colaboração de inovação.

COLOCAR MAIS UMA DEFINICAO DE SOFTWARE CIENTIFICO

### Particularidades

Conforme mencionado anteriormente o software cientifico tem um conjunto de particularidades que o diferenciam dos softwares tradicionais. Essas particularidades basicamente estão na engenharia de construção desse tipo software e que acabam por se propagar nas ferramentas computacionais empregadas, nas características da equipe que o constrói, no tipo de gestão e condução do projeto e até do tipo do modelo de contratação caso se opte pela terceirização.

KELLY(2015) apresenta um modelo de desenvolvimento de software baseado em aquisição de conhecimento, fundamentado em 10 anos de estudo de softwares científicos e dos cientistas que desenvolveram software como parte de sua pesquisa. Nesse trabalho ele apresenta algumas características do software cientifico:

1. Um especialista do domínio científico é necessariamente envolvido no processo de desenvolvimento do software;
2. O usuário deste software possui um conhecimento mínimo do domínio científico associado, para permitir a interpretação correta dos dados de saída;
3. O usuário é o destinatário de todas as saídas a partir do software, significando que o propósito do software não é controlar equipamentos;
4. O principal objetivo do software é fornecer dados para a compreensão de problemas específicos do mundo real, o que significa que os cientistas que estudados não desenvolvem ferramentas generalizadas e bibliotecas para suportar computação computacional;
5. A qualidade do software é substituída pela sua correção - ou mais precisamente, confiabilidade e fidedignidade para falhas, então todas as outras qualidades de software são irrelevantes.

Para realizar esses estudos KELLY estudou diferentes industrias que trabalham fortemente com o desenvolvimento de softwares científicos e em cada uma delas ele fez observações sobre o que de fato é importante para o pesquisador, contrariando muitas vezes o que a engenharia de software recomenda:

* O software é considerado inseparável da sua pesquisa e o teste é parte chave de fazer ciência;
* O cientista deve ser considerado como parte do software e o seu teste não deve ser baseado como um produto e sim baseado em construção e conhecimento
* Os cientistas criam pequenos números de testes “bem escolhidos” ao invés de grandes baterias de testes sistemáticos, e para isso usam seu conhecimento da ciência e dos algoritmos do software;
* Os cientistas gastam muito tempo testando e preocupados com seus resultados. O software não é conceitualmente uma entidade separada de sua teoria aplicada ou do seu uso no domínio de aplicação;
* Não há menção de nenhum padrão de documentação de engenharia de software tais como design ou requisitos. A fonte da informação vem da prática humana, e as documentações vem de uma vasta variedade de formatos e conteúdo, algumas vezes especificamente desenvolvidas para as necessidades do usuário/desenvolvedor;
* Todos os cientistas trabalham com foco no ganho de conhecimento ao invés de responder questões científicas;
* A abordagem dos cientistas para o desenvolvimento de software está em desacordo com o método baseado em abordagens utilizadas por engenheiros de software (SEGAL *apud KELLY, 2005)*
* Os cientistas no geral fazem exaustivos testes e muitos estão familiarizados com o código, seus compiladores e seus sistemas operacionais; estando plenamente conscientes do impacto de erros na sua ciência e da falta de conceitos de engenharia de software (especificação de requisitos, teste unitários, teste de cobertura, documentação do design, reuso do design)

### Construção de software científico

### Riscos de um software científico

Sanders, Rebecca, and Diane Kelly**. "Dealing with risk in scientific software development."** IEEE software 4 (2008): 21-28.

## Metodologias de Desenvolvimento de Software: do tradicional ao ágil (SCRUM)

### Definição

### Metodologia Tradicional

### Metodologia SCRUM

A origem literal da palavra SCRUM é o nome dados a uma jogada do rúgbi onde os jogadores dos dois times, formam um único bloco de pessoas encaixadas pelos ombros e pelos braços. A bola é jogada no meio desse bloco e neste momento um time tenta empurrar o outro até que a bola saia entre as pernas dos jogadores e o jogo continue. O ponto crucial dessa jogada é o trabalho em equipe, onde se um falhar na formação, o outro time ganha bola e domina a situação.

A criação do SCRUM como definição metodológica e metaforizando com sua origem vem de 1986, Hirotaka Takeuchi e Ikujiro Nonaka publicaram um estudo intitulado de "The New Product Development Game" (Harvard Business Review, Janeiro-Fevereiro 1986). Nesse estudo, eles compararam equipes pequenas, de alto desempenho e multidisciplinares à jogada "Scrum" do rúgbi. Takeuchi e Nonaka descobriram que utilizando esse tipo de equipe, obteriam melhores resultados.

RUBIN (2012) define SCRUM como uma prática ágil para desenvolver produtos e serviços inovadores. Ainda segundo RUBIN, é um framework para organização e gerenciamento do trabalho, baseado em valores, princípios e práticas que proveem as fundações sobre as quais as organizações adicionarão suas implementações de práticas relevantes de engenharia e suas abordagens específicas para o uso do SCRUM.

O ciclo de vida do SCRUM definido sem o formalismo de suas ferramentas pode ser transcrito como uma prática que reúne um conjunto de pessoas com atribuições e objetivos em comum debruçados sobre a construção de requisitos iniciais previamente priorizados pelo cliente; onde são realizadas entregas em períodos contínuos de intervalos pré-acordados de tempo, permitindo ajustes pelo cliente nos casos em que o produto parcialmente entregue não esteja conforme sua expectativa.

O ciclo de vida do SCRUM definido dentro do formalismo de suas ferramentas se inicia a partir do Product Backlog (requisitos do produto). A partir dele, é montado o Sprint Backlog, que são as tarefas executadas em uma Sprint. Durante a Sprint, acontecem reuniões diárias onde cada membro da equipe reporta o que fez no dia anterior e o que fará no dia corrente. Cada Sprint pode durar de 2 a 4 semanas, e ao final, acontece a Sprint Review com o cliente, para que sejam entregues os itens da Sprint, validados, aprovados para que a Sprint seja encerrada. No final da Sprint é realizada a retrospectiva, para que sejam avaliados os pontos positivos e negativos. Isso é realizado repetidas vezes até que o Product Backlog não tenha mais nenhum requisito de produto e o produto final seja entregue.

FIGURA CICLO DE VIDA

Abaixo são detalhados os principais componentes do SCRUM que são utilizados nos projetos da Petrobras com foco em projetos de software cientifico. Para melhor organização, o assunto está dividido em papéis e responsabilidades definidos pela metodologia, tipos de reuniões, artefatos produzidos e os benefícios trazidos na gestão do projeto. A partir dessas definições será possível realizar comparações com metodologias de desenvolvimento tradicionais, e verificar sua eficiência na condução da construção e da gestão de projetos de software científicos.

#### **Papéis**

##### Product Owner (PO)

O PO é o principal responsável pela definição do produto que será entregue ao final do projeto. Entre suas principais responsabilidades estão:

* Priorizar os requisitos;
* Participar do planejamento;
* Alimentar o product backlog;
* Definir critérios de aceitação e verificar se eles são atendidos;
* Colaborar com o time de desenvolvimento
* Colaborar com os *stakeholders;*

##### Scrum Master (SM)

O SM é o principal responsável por ajudar a todos os envolvidos a entenderem e abraçarem os princípios, valores e práticas do SCRUM. Entre suas principais responsabilidades estão:

* Ser um *coach* do time;
* Ser um líder servo, se disponibilizando sempre a ajudar o time;
* Ser uma autoridade de implementação do processo do SCRUM;
* Proteger o time de interferências externas;
* Resolver impedimentos que atrapalhem o andamento do trabalho do time;
* Ser agente de mudança.

##### Time

O Time é composto de um conjunto de profissionais que são responsáveis por desenhar, construir e testar um determinado produto. No desenvolvimento de software é composto por arquitetos, programadores, testadores, administradores de banco de dados, designers e outros.

#### **Ferramentas**

##### Sprint

Para RUBIN (2012) o SCRUM organiza o trabalho em iterações ou ciclos baseados em um calendário mensal chamado sprints, que se caracterizam por ter um limite de tempo, geralmente de curta duração, um objetivo que não deve ser alterado uma vez iniciado, e que deve alcançar o estado final especificado pelo time.

##### Product Backlog

O Product Backlog representa a visão total do produto pelo cliente, reunindo trabalho que só poderia ser realizado em semanas ou meses e que não podem ser construídos em uma única Sprint. Uma de suas principais características é o dinamismo, onde a cada Sprint é alterado com a inclusão, exclusão ou alteração dos seus items.

##### Histórias

##### Tarefas

#### **Reuniões**

##### Sprint Planning

Para determinar o conjunto de requisitos mais importantes para a composição de uma Sprint, o time se reúne para fazer o planejamento da Sprint. Nessa reunião, o time chega a um acordo sobre o objetivo da Sprint e seleciona os itens do prodcut backlog que estão alinhados com o objetivo e podem ser entregues no final da Sprint.

##### Sprint Execution

É todo o trabalho realizado para atingir o objetivo da Sprint, funcionando como um miniprojeto que ao final precisa entregar um incremento do produto pronto para ser utilizado caso seja de interesse do cliente.

##### Daily Meeting

É a reunião diária, geralmente realizada no mesmo horário e com curto tempo de duração onde o time tenta responder o que fez no dia anterior, o que planeja fazer no dia e se há algum impedimento ou obstáculo que impeça o andamento do seu trabalho.

##### Sprint Review

É a reunião que tem como participantes o time, os stakeholders, os patrocinadores, clientes e interessados; e tem como objetivo inspecionar e adaptar o produto que está sendo construído. Isso permite que todos tenham uma visão clara do que está sendo construído e possa ajudar a se chegar a solução mais apropriada para o negócio.

##### Retrospective

É a reunião

### Comparando as metodologias

FENG, Ji, & SEDANO, T., **Comparing Extreme Programming and Waterfall project Results**, Carnegie Mellon University, Jan 2011

Ayca Tarhan, Seda Gunes Yilmaz. **Systematic analyses and comparison of development performance and product quality of Incremental Process and Agile Process**, Information and Software Technology, Vol.26 No.5, 2014

## Metodologia PMI aplicado a SCRUM

### Gerenciando projetos com PMI

PROJECT MANAGEMENT INSTITUTE. (2013). **Um guia do conjunto de conhecimento em gerenciamento de projetos: guia PMBoK**® (5. ed. ed.). Newtown: Four Boulevard.

KERZNER, Harold – **Project Management: A Systems Approach to Planning, Scheduling, and Controlling**, 10ª ed., Editora Jonh Wiley & Sons, Inc, 2009

### PMI aplicado a metodologia de desenvolvimento SCRUM

### Problemas e incompatibilidades

MILLER, J. G. **Going Agile Project Management Practices**, 1st Edn. Maxmetrics LLC. 2013

## Modelos de contratação de serviços de desenvolvimento de software

### Métodos de contratação segundo PMI

### Contratação de serviços com fixo e escopo fechado

PROJECT MANAGEMENT INSTITUTE. (2013). **Um guia do conjunto de conhecimento em gerenciamento de projetos: guia PMBoK**® (5. ed. ed.). Newtown: Four Boulevard.

CARNEIRO, W. **Análise comparativa de modelos de contratos propostos pelo PMBOK e pelo Código Civil Brasileiro**. Revista MundoPM, n. 8, p. 6-12, 2006.

### Métodos de contratação segundo a legislação brasileira

TRIBUNAL DE CONTAS DA UNIÃO. **Guia de boas práticas em contratação de soluções de tecnologia da informação: riscos e controles para o planejamento da contratação,** 1ª ed., Tribunal de Contas da União, 2012

CARNEIRO, W. **Análise comparativa de modelos de contratos propostos pelo PMBOK e pelo Código Civil Brasileiro**. Revista MundoPM, n. 8, p. 6-12, 2006.

## Padrões Petrobras

### Métodos Ágeis

[PG-2T0-00050-J- IMPLEMENTAR SOLUÇÕES](http://rjln202/03256935004D0A69/0/DDC0F54CFF8A2C0683257A090067D0A1?OpenDocument)

PE 2T0-00505 Gestão de desenvolvimento de soluções de software através de Projeto Tradicional

### Métodos Ágeis aplicados a desenvolvimento científico

PE 2T0-00506 Gestão de desenvolvimento de soluções de software através de Projeto Ágil

### Modelo de Contratação de Serviços

PP-4B0-00037-0- REALIZAR CONTRATAÇÃO DE SERVIÇOS POR LICITAÇÃO

# SOFTWARE CIENTÍFICO X MÉTODOS ÁGEIS X CONTRATAÇÃO DE SERVIÇOS X GESTÃO DE PROJETOS

## METODOS ÁGEIS APLICADO NAS GRANDES CORPORAÇÕES

### Utilização

### Problemas

### Petrobras

G. Van Waardenburg, Hans. van Vliet, **When agile meets the enterprise**, Inform. Softw. Technol., 55 (12) (2013), pp. 2154–2171

## NEGOCIANDO CONTRATOS COM MÉTODOS ÁGEIS

### Utilização

### Problemas

Mattos A. D. “**Quando pagar multa compensa.”** Disponível em: http://www.aldomattos.com/sites/aldomattos.com/files/publicacoes/Quando\_Atrasar\_Compensa.pdf, acessado em 15/outubro/2015

### Petrobras

FRANKIN, T., **Adventures in agile contracting: evolving from time and materials to fixed price, fixed scope contracts**, in: AGILE ‘08 Conference, 2008

Zijdemans, S., & Stettina, C. (2014). **Contracting in Agile Software Projects: State of Art and How to Understand It**. Agile Processes in Software Engineering and …. Retrieved from <http://link.springer.com/chapter/10.1007/978-3-319-06862-6_6>

R. Hoda, J. Noble, and S. Marshall, **Negotiating contracts for agile projects: A practical perspective**, In: XP 2009. Volume 31 of Lecture Notes in Business Information Processing, Springer, 2009.

BATRA, D., SIN, T., and TSENG, S. (2006). **Modified agile practices for outsourced software projects**. Proceedings of the twelfth Americas Conference on Information Systems, Acapulco, Mexico August 4-6, 2006, 3872-3880.

## MÉTODOS ÁGEIS APLICADO EM SOFTWARE CIENTÍFICO

### Utilização

### Problemas

### Petrobras

M.T. Sletholt, J.E. Hannay, D. Pfahl, H.C. Benestad, H.P. Langtangen, **A literature review of agile practices and their effects in scientific software development**, in: Proceedings of SE-CSE11, 2011, pp. 1–9

Sletholt, Magnus Thorstein, et al. "**What do we know about scientific software development's agile practices?."** Computing in Science & Engineering 14.2 (2012): 24-37.

Ahalt, Stan, et al. **"Water Science Software Institute: Agile and open source scientific software development."**Computing in Science & Engineering 16.3 (2014): 18-26.

## GESTÃO DE PROJETOS COM MÉTODOS ÁGEIS

### Utilização

### Benefícios

Escopo

Prazo

Custo

Qualidade

BATRA, D., SIN, T., and TSENG, S. (2006). **Modified agile practices for outsourced software projects**. Proceedings of the twelfth Americas Conference on Information Systems, Acapulco, Mexico August 4-6, 2006, 3872-3880.

Singh, N.P. and R. Soni, 2011. **Agile software: Ensuring quality assurance and processes.** Proceeding of the International Conference on High Performance Architecture and Grid Computing, Jul. 19-20, Chandigarh, India, pp: 640-648. DOI: 10.1007/978- 3-642-22577-2\_86

Stare, Aljaž. **"Agile Project Management in Product Development Projects."***Procedia-Social and Behavioral Sciences* 119 (2014): 295-304.

### Problemas

Lagerberg, L. S. T., Emanuelsson, P., Sandahl, K., & Stahl, D., (2013). **The impact of agile principles and practices on large scale software development projects a multiple-case study of two projects at Ericsson**. Proceedings of the International Symposium on Empirical Software Engineering and Measurement.

### Petrobras

# ESTUDO DE CASO

## Projeto RCDUT

## Escopo Inicial

## Escopo Final

## Cronograma

## Orçamento

## Análises e Considerações no estudo realizado

Variação de X % no escopo original. Em um contrato de escopo fechado por prestação de serviços aplicados a um software científico teríamos grandes possibilidades de:

No término do contrato não ter o produto desejado pelo cliente;

Assinatura de aditivos contratuais para término do produto e realização das mudanças solicitadas;

Explosão do orçamento e perda na qualidade do produto final, uma vez que o custo de mudança e o impacto na qualidade de um software na fase final é muito superior a mudanças realizadas na fase de concepção e elaboração;

Custo de oportunidade de não ter o produto na data e na especificação acordada. O Projeto RCDUT por exemplo, possibilita a economia de milhões de reais evitando paradas de produção.

Perda do investimento por insolvência da empresa prestadora de serviço.

# CONSIDERAÇÕES FINAIS

## Críticas e Comentários

## Recomendações para Futuros Trabalhos.

# REFERÊNCIAS

## Referências Bibliográficas

Kerzner, Harold R. ***Project management: a systems approach to planning, scheduling, and controlling*.** John Wiley & Sons, 2013.

Kelly, Diane. **"Scientific software development viewed as knowledge acquisition: Towards understanding the development of risk-averse scientific software."** Journal of Systems and Software 109 (2015): 50-61.

Pressman, Roger S. **Software engineering: a practitioner's approach**. Palgrave Macmillan, 2005.

Moura, Marcelle Cristina, and Souza Purri. **"Estudo e Propostas Iniciais para a Definição de um Processo de Desenvolvimento para Software Científico."** (2006).

Howison, James, and James D. Herbsleb. **"Incentives and integration in scientific software production."** Proceedings of the 2013 conference on Computer supported cooperative work. ACM, 2013.

Segal, Judith. **"When software engineers met research scientists: A case study."** Empirical Software Engineering 10.4 (2005): 517-536.

Rubin, Kenneth S. **Essential Scrum: A practical guide to the most popular Agile process**. Addison-Wesley, 2012.

MILLER, J. G. “**Going Agile Project Management Practices”**, 1st Edn. Maxmetrics LLC. 2013

Ji, Feng, and Todd Sedano. "**Comparing extreme programming and Waterfall project results."**Software Engineering Education and Training (CSEE&T), 2011 24th IEEE-CS Conference on. IEEE, 2011.

Tarhan, Ayca, and Seda Gunes Yilmaz. **"Systematic analyses and comparison of development performance and product quality of Incremental Process and Agile Process."**Information and Software Technology 56.5 (2014): 477-494.

Rose, Kenneth H. **"A Guide to the Project Management Body of Knowledge (PMBOK® Guide)—Fifth Edition."** Project Management Journal 44.3 (2013): e1-e1.

Franklin, Teresa. **"Adventures in agile contracting: Evolving from time and materials to fixed price, fixed scope contracts."** Agile, 2008. AGILE'08. Conference. IEEE, 2008.

Zijdemans, Shi Hao, and Christoph Johann Stettina. **"Contracting in Agile Software Projects: State of Art and How to Understand It."**Agile Processes in Software Engineering and Extreme Programming. Springer International Publishing, 2014. 78-93.

Hoda, Rashina, James Noble, and Stuart Marshall. **"Negotiating contracts for agile projects: A practical perspective."** Agile Processes in Software Engineering and Extreme Programming. Springer Berlin Heidelberg, 2009. 186-191.

Van Waardenburg, Guus, and Hans Van Vliet. **"When agile meets the enterprise."** Information and software technology 55.12 (2013): 2154-2171.

TRIBUNAL DE CONTAS DA UNIÃO. **Guia de boas práticas em contratação de soluções de tecnologia da informação: riscos e controles para o planejamento da contratação,** 1ª ed., Tribunal de Contas da União, 2012

CARNEIRO, W. **Análise comparativa de modelos de contratos propostos pelo PMBOK e pelo Código Civil Brasileiro**. Revista MundoPM, n. 8, p. 6-12, 2006.

Batra, Dinesh. **"Modified agile practices for outsourced software projects."** Communications of the ACM 52.9 (2009): 143-148.

Sillitti, Alberto, et al. **"Managing uncertainty in requirements: a survey in documentation-driven and agile companies."** Software Metrics, 2005. 11th IEEE International Symposium. IEEE, 2005.

Sletholt, Magnus Thorstein, et al. **"A literature review of agile practices and their effects in scientific software development."** Proceedings of the 4th International Workshop on Software Engineering for Computational Science and Engineering. ACM, 2011.

Singh, Narinder Pal, and Rachna Soni. **"Agile software: Ensuring quality assurance and processes."** High Performance Architecture and Grid Computing. Springer Berlin Heidelberg, 2011. 640-648.

Lagerberg, Lina, et al. **"The impact of agile principles and practices on large-scale software development projects: A multiple-case study of two projects at ericsson."** Empirical Software Engineering and Measurement, 2013 ACM/IEEE International Symposium on. IEEE, 2013.

Torrecilla-Salinas, C. J., et al. **"Estimating, planning and managing Agile Web development projects under a value-based perspective."** Information and Software Technology 61 (2015): 124-144.

Stare, Aljaž. **"Agile Project Management in Product Development Projects."**Procedia-Social and Behavioral Sciences 119 (2014): 295-304.

Sletholt, Magnus Thorstein, et al. **"What do we know about scientific software development's agile practices?."**Computing in Science & Engineering 14.2 (2012): 24-37.

Ahalt, Stan, et al. **"Water Science Software Institute: Agile and open source scientific software development."**Computing in Science & Engineering 16.3 (2014): 18-26.

Mattos A. D. “**Quando pagar multa compensa.”** Disponível em: http://www.aldomattos.com/sites/aldomattos.com/files/publicacoes/Quando\_Atrasar\_Compensa.pdf, acessado em 15/outubro/2015

Sanders, Rebecca, and Diane Kelly**. "Dealing with risk in scientific software development."** IEEE software 4 (2008): 21-28.

## Referências Normativas

## Indicações Eletrônicas