**ARTIGO ATENÇÃO MUDAR O TÍTULO**

**UM COMPARATIVO DAS METODOLOGIAS ÁGEIS XP E SCRUM PARA O DESENVOLVIMENTO RAD**

Lucas Fagundes Teixeira, Wagner Rodrigo da Silva,

Curso de Bacharelado em Engenharia de Software – Centro de Ensino Superior de Juiz de Fora (CESJF) – Campus Academia  
360100-000 – Juiz de Fora – MG– Brasil

lucasft\_01@outlook.com, [wagnerrodrigo.pan@gmail.com](mailto:wagnerrodrigo.pan@gmail.com)

***Resumo.***O

**1 INTRODUÇÃO**

As metodologias ágeis surgiram para suprir certos entraves no desenvolvimento que as metodologias tradicionais não conseguiam tratar. Feedback rápido com o cliente e rápida adaptação com os inevitáveis problemas ao longo do projeto são apenas algumas de outras complicações não tratadas pelo sistema arcaico. Os resultados dessas dificuldades podem ser notados em alguns dados recolhidos em 1995[1] onde, usando como base 8380 projetos, mostram que apenas 16,2% dos projetos foram entregues respeitando os prazos e os custos e com todas as funcionalidades especificadas. Aproximadamente 31% dos projetos foram cancelados antes de estarem completos e 52,7% foram entregues, porém com prazos maiores, custos maiores ou com menos funcionalidades do que especificado no início do projeto.

A idéia das metodologias ágeis é o enfoque nas pessoas e não em processos ou algoritmos. Além disso, existe a preocupação de gastar menos tempo com documentação e mais com a implementação. Uma característica das metodologias ágeis é que elas são adaptativas ao invés de serem preditivas. Com isso, elas se adaptam a novos fatores decorrentes do desenvolvimento do projeto, ao invés de procurar analisar previamente tudo o que pode acontecer no decorrer do desenvolvimento.

Dentre as várias metodologias ágeis existentes, as mais conhecidas são a *Extreme Programming* e a *Scrum*. E são nestas que o artigo tem enfoque.

# **2. Referencial Teórico**

Segundo o Manifesto Ágil, definido por Beck et al. (2001)

Um conjunto de regras pensadas e elaboradas com um propósito de melhoria essa é a proposta de desenvolvimento de software, está metodologia de desenvolvimento foi sugerida por (SOMMERVILLE, 2007)

Estas metodologias foram criada para resolver dificuldades encontradas na área de software

**2.1. Padrões e procedimentos utilizados para o desenvolvimento de software**

oplicação a um novo colaborador. Como forma de boas prática de desenvolvimento de DB s

# **3. METODOLOGIAS ÁGEIS XP E SCRUM PARA O DESENVOLVIMENTO RAD**

## 

## **3.1 XP**

A Extreme Programming (XP) é uma metodologia ágil para equipes pequenas e médias que desenvolvem software baseado em requisitos vagos e que se modificam rapidamente. [2]

Conforme Tabela 1, a metodologia XP, busca trazer valores e motivações para toda equipe envolvida no projeto de desenvolvimento. Levar confiança ao cliente de que suas necessidades serão respeitadas diante de um informativo de prioridades é de vital importância para que o projeto não seja interrompido.

**Tabela 1 – Valores de Extreme Programming**

|  |  |
| --- | --- |
| FeedBack | Comunicação |
| Simplicidade | Coragem |

Mostradas como uma série de atividades a serem seguidas, as práticas de XP norteiam a equipe desenvolvedora durante o projeto. Os valores citados anteriormente, somados as práticas ( Tabela 2), resultam em um grupo de boas atitudes.

**Tabela 2 – Práticas de Extreme Programming**

|  |  |  |
| --- | --- | --- |
| Cliente Presente | Jogo do Planejamento | Integração Contínua |
| Pequenas Versões | Metáforas | Desenvolvimento Orientado a Testes |
| Projeto Simples | Time Coeso | Refatoração |
| Testes de Aceitação | Ritmo Sustentável | Padrões de Codificação |
| Reuniões em Pé | Posso Coletiva | Programação em Pares |

**3.1.1 A Equipe e Seus Membros**

Dentro de uma equipe XP, não existem pessoas com ações mais importantes que as outras, não existindo hierarquia. Cada membro faz sua contribuição ao projeto da melhor maneira que for possível.

**Tabela 3 – Membros de Equipe de Extreme Programming**

|  |  |
| --- | --- |
| Gerente de Projetos | Treinador |
| Analista de Testes | Redator Técnico |
| Desenvolvedor | Rastreador |

**a)** Gerente de Projetos:

Objetivo: Assegurar uma comunicação eficiente entre o cliente e todos os envolvidos no projeto. Também devem monitorar o progresso da equipe e ajudá-la a perceber tudo que foi conquistado.

**b)** Analista de testes:

Objetivo: Ajudar clientes e desenvolvedores a escreverem testes para as histórias. Além disso, auxiliam os desenvolvedores na automatização do testes ao longo da iteração. Quando a equipe não consegue automatizar alguns testes, os analistas de teste os executam manualmente.[4]

**c)** Desenvolvedor:

Objetivo: Ajudar a estimar as histórias, implementar as histórias e automatizar tarefas repetitivas. Também são responsáveis por criarem testes automatizados para tudo o que produzem usando o TDD- que em portuguuês significa desenvolvimento orientado a testes. Além de criarem novas funcionalidades, é necessário refatorar o sistema constantemente para aprimorar a arquitetura, eliminar duplicações e tornar o código mais limpo.[5]

**d)** Treinador:

Objetivo: Deve identificar no que cada um é bom, chamar atenção para oportunidades de melhorias e lembrar a equipe das regras. Pode, eventualmente, fazer programação pareada.[6]

**e)** Redator técnico:

Objetivo: Ajudar a equipe a criar e manter a documentação do projeto. Por tanto, Os redatores técnicos devem compreender quem é o público alvo das publicações, assegurar que a linguagem certa seja usada e fazer com que a informação seja atualizada continuamente.[6]

f) Rastreador:

Objetivo: Coletar as métricas que vão sendo desenvolvidas e compara-las com as métricas que foram estimadas, realizando uma análise das eventuais divergências. Também é responsável por avaliar a viabilidade do objetivo conforme as limitações de recursos e tempo.[6]

**3.1.2 Ciclo de Vida**

Um projeto XP passa pelas seguintes fases: exploração, planejamento inicial, iterações do release, produção, manutenção e morte.[3]

**a)**Exploração:

*Objetivo:* Investigar possíveis soluções e verificar a viabilidade de tais soluções.

*Funcionamento:* Os programadores elaboram possíveis arquiteturas e tentam visualizar como o sistema funcionará considerando o ambiente tecnológico (hardware, rede, software, performance, tráfego) onde o sistema irá rodar.

b)Planejamento inicial:

Objetivo**:** Entrar em acordo com o cliente em uma data de entrega para o primeiro release.

*Funcionamento:* Os programadores, juntamente com o cliente, definem as estórias (use case simplificados) a serem implementadas e as descrevem em cartões. Os programadores assinalam uma certa dificuldade para cada estória e, baseados na sua velocidade de implementação, dizem quantas estórias podem implementar em uma iteração. Depois, os clientes escolhem as estórias de maior valor para serem implementadas na iteração – isso é chamado planejamento de iteração. O processo então se repete até terminar as iterações do release. O tempo para cada iteração deve ser de uma a três semanas e para cada release de dois a quatro meses.

c) Iterações do release:

*Objetivo:* Ter uma idéia melhor das tecnologias e do domínio do problema de modo que as iterações poderão ser mais curtas nos releases subseqüentes e fazer estimativas mais confiáveis com o que se aprendeu das iterações passadas.

*Funcionamento: S*ão escritos os casos de teste funcionais e de unidade. Os programadores vão seguindo mais ou menos o seguinte fluxo de atividades na seguinte ordem (Em cada iteração): escrita dos casos de testes; projeto e refatoramento; codificação; realização dos testes; e integração. À medida que esse fluxo vai sendo seguido, o sistema vai sendo construído segundo os princípios, valores e práticas apresentados nas seções anteriores.

d) Manutenção:

*Objetivo:* Reajustar o processo para conseguir adequar às imprevisibilidades que vão surgindo no decorrer do processo. Exemplo: implementação de novas funcionalidade, chegada de novos membros e a reescrita do código para o seu melhoramento.

*Funcionamento:* Estando em XP sempre simultaneamente produzindo novas funcionalidades, mantendo o sistema existente rodando, incorporando novas pessoas na equipe e melhorando o código. Mecanismos como: refatoramento, introdução de novas tecnologias, e introdução de novas idéias de arquitetura podem ser utilizados em um projeto XP.

e) Morte:

*Objetivo:* Dar fim à um projeto, seja pelo motivo do cliente já estar satisfeito, seja pela inviabilidade em dar continuidade ao desenvolvimento.

*Funcionamento: Quando não mais existir novas histórias, é o momento de finalizar o projeto. É o momento de escrever algumas páginas (de 5 a 10 páginas) sobre a funcionalidade do sistema, um documento que auxilie no futuro a saber como realizar alguma alteração no sistema. Toda a equipe que trabalhou no sistema deve ser reunida para reavaliação. Aproveite a oportunidade de analisar o que pode ter causado queda no sistema e o que fez o projeto avançar.*

Figura 1: Ciclo de Vida do XP

A close up of a map

Description generated with high confidence

## 

## **3.2 Scrum**

Esta metodologia ágil foi criada em 1993 por Jeff Sutherland e, de acordo com o próprio criador, teve como base várias ideias de um estudo conduzido em 1986 por dois professores de administração japoneses, Hirotaka Takeuchi e Ikujiro Nonaka, publicado na Harvard Business Review.

Neste estudo, os dois professores conseguiram identificar que nas empresas haviam pequenas equipes com grande autonomia e liberdade trabalhando movidas por um grande propósito. Os superiores não ditavam como os trabalhos deviam ser feitos, eles apenas removiam os problemas que impediam a equipe de ser bem sucedida. Isto sendo feito, o trabalho era conduzido em um fluxo contínuo de geração de valor, evitando desperdícios de tempo e dinheiro.

Em seu livro, Scrum - a arte de fazer o dobro do trabalho na metade do tempo, Jeff Sutherland fala que para o Scrum realmente funcionar, alguém da equipe sênior de gerenciamento deve ter uma alta compreensão do quão prejudiciais ao fluxo do desenvolvimento os obstáculos são. Além disto, todos devem ter ciência da drasticidade do efeito da eliminação destes obstáculos. No entanto, as pessoas não costumam adotar estas recomendações por exigir muita honestidade consigo mesmas e com os outros.

Apenas com uma breve introdução ao Scrum, já é possível notar a importância dada as pessoas por esta metodologia. Pois, em meio a infinidade de contratempos em um processo e a natural diferença entre os membros do grupo, a melhor alternativa é deixar com que ocorra uma auto-organização e adaptação quando necessário. Mas, para que isso seja possível de ser feito, um conjunto de valores, princípios e práticas devem ser adotados e adequados a realidade da organização. Por este motivo, o Scrum é considerado como um framework altamente adaptável as mais diversas situações e no fim o resultado será uma versão que é exclusivamente sua.

Ao decorrer deste tópico 3.2 será citado as práticas, os valores, membros da equipe e todo o ciclo do Scrum.

**3.2.1 Práticas do Scrum**



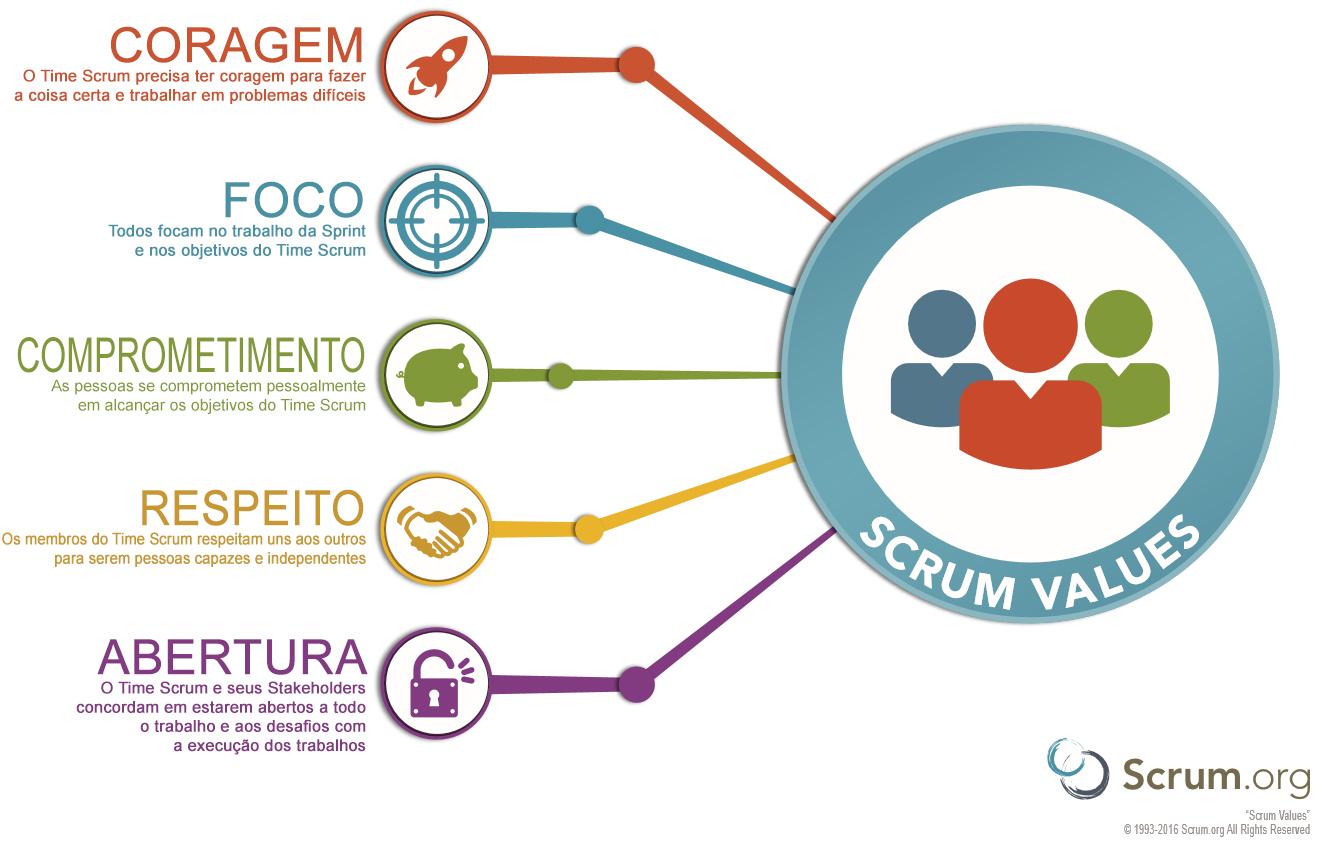
Figura 2: Práticas do Scrum

Figura 2 retirada do endereço <http://www.mindmaster.com.br/scrum/>

**3.2.2 Valores do Scrum**

Quando os valores de comprometimento, coragem, foco, abertura e respeito são assumidos e vividos pelo Time Scrum, os pilares do Scrum de transparência, inspeção e adaptação tornam-se vivos e constroem a confiança para todos. Os membros do Time Scrum aprendem e exploram estes valores à medida que trabalham com os eventos, papéis e artefatos do Scrum (2016, Guia do Scrum 2016, p.5)

Figura 3



Traduzido livremente por Fábio Cruz e localizado no caminho http://www.fabiocruz.com.br/valores-scrum/ .

Original no site da scrum.org no caminho https://www.scrum.org/Resources/What-is-Scrum

**4. Considerações Finais**

a modelagem de dados é algo desafiador para equipes de Tecnologia da Informação (TI), especialmente quando os modelos são desenvolvidos por profissionais diferentes.

# **Referências**

HIROTAKA TAKEUCHI, [IKUJIRO NONAKA](https://hbr.org/search?term=ikujiro+nonaka) - **The New New Product Development Game.** Disponível em https://hbr.org/1986/01/the-new-new-product-development-game. Acesso em 28 de Dezembro de 2017.

DENISSON VIEIRA - **Scrum: A Metodologia Ágil Explicada de forma Definitiva.** Disponível em http://www.mindmaster.com.br/scrum. Acesso em 28 de Dezembro de 2017.

ALARIC DAILEY **– Database Design and Naming Conventions** Disponível emhttps://www.codeproject.com/Articles/232457/Database-Design-and-Naming-Conventions. Acesso em 01 novembro de 2017.

DEVMEDIA **– Introdução aos Stored Procedures no SQL Server** Disponível em *[http://www.devmedia.com.br/introducao-aos-stored-procedures-no-sql-server/7904].* Acesso em 22 novembro de 2017.

DATASUS **– MAD Metodologia de Administração de Dados**. Disponível em http://datasus.saude.gov.br/estrutura-mad/norma-mad-menu. Acesso em 31 de outubro de 2017.

DATE, C. J.. **Introdução a Sistemas de Banco de Dados. Elsevier Editora, 2004**.

JEFFREY EDISON **– Naming Coventions in Database Modeling.** Disponível em http://www.vertabelo.com/blog/technical-articles/naming-conventions-in-database-modeling. Acesso em 4 de novembro de 2017.

MSDN **– Especificar Convenções de Nomenclatura (Assistente de Geração de Esquema) (Analysis Services - Dados Multidimensionais)** Disponível em <https://msdn.microsoft.com/pt-br/library/ms177199(v=sql.120).aspx>. Acesso em 01 de outubro de 2017.

SEHROPE SARKUNI **– How I Write SQL, Part 1: Naming Conventions** Disponível em https://launchbylunch.com/posts/2014/Feb/16/sql-naming-conventions/ Acesso em 04 de novembro de 2017.

SILVEIRA, P.; Silveira, G.; LOPES, S.; Moreira, G.; STEPAAT, N. & Kung, F. **–** **Introdução à Arquitetura de Design de Software: Uma Introdução à Plataforma Java** **Elsevier Brasil, 2011**