unipampa
Universidade Federal do Pampa

# Trabalho 01: **Desenvolvimento Ágil**

**Dionatã Dorneles Mafaldo** matrícula nº 151151477
**Guilherme Legramante Martins** matrícula nº 1551150228
**Jean Trindade Piagetti** matrícula nº 151150230
**Naihara Sousa Amorim** matrícula nº 151150946
**Raul Vitor Costa Lessa** matrícula nº 151150948

Resumo: *O presente trabalho tem por finalidade expor alguns aspectos dos métodos de Desenvolvimento Ágil de Software, expondo uma perspectiva de trabalho com os referidos métodos. Serão apresentados alguns conceitos principais e fatores que implicaram no surgimento destas metodologias, com foco nos métodos Scrum e Extreme Programming(XP) que exemplificarão este desenvolvimento ágil. As ferramentas utilizadas no processo, bem como suas principais definições, abordagens e implicações estarão presentes no decorrer deste artigo.*

## 1. INTRODUÇÃO

Após pesquisa bibliográfica, foi elaborada uma pesquisa a respeito dos Métodos de Desenvolvimento Ágil de Software. Será apresentado primeiramente o contexto do surgimento da metodologia ágil e de seu manifesto. Os métodos Scrum e Extreme Programming serão definidos e explanados com relação a sua implementação e conceitos principais. Seguido, das situações que melhor se encaixam para este tipo de abordagem, bem como as ferramentas que podem ser utilizadas nestes processos ágeis.

Na seção Definição será tratado sobre os conceitos principais e sobre os princípios definidos no Manifesto Ágil. Também serão definidos os conceitos de Extreme Programming e Scrum. Após, na seção Aplicações, contextualiza-se as melhores ocasiões para utilizar uma metodologia ágil e os motivos que levam a isso, em detrimento dos métodos tradicionais. Algumas ferramentas que auxiliam no processo de desenvolvimento ágil como VIMEE, *RemotePP*, CVW serão expostas e também demonstradas as formas que poderão ser utilizadas. Finalizando, a seção Desafios abordará os pontos que ainda precisam ser melhorados para que as metodologias em questão sejam aplicadas da melhor forma possível.

## 2. DEFINIÇÃO

### 2.1 Metodologias Clássicas

Os métodos tradicionais ou clássicos foram unanimidade na forma de desenvolvimento de software até a década de 90. Porém, estes processos que são orientados

a documentação implicam algumas dificuldades principalmente para organizações que não possuem recursos para investir nos processos de produção de software. Como consequência disso, estas empresas na sua maioria pequenas, optam por não utilizar nenhum processo. O que poderá influenciar na qualidade do produto entregue ao cliente.

## 2.2 Metodologias Ágeis

Os métodos tradicionais de desenvolvimento de software não se mostram eficazes em algumas situações. Para Pressman [PRE10] em essência, as Metodologias Ágeis foram desenvolvidas com o objetivo de vencer as fraquezas percebidas e reais da Engenharia de Software.

O termo "Metodologias Ágeis" tornou-se conhecido em 2001, quando dezessete especialistas em processos de desenvolvimento de software, apresentaram os métodos ágeis para desenvolvimento de software.

Destacaram-se dentre os demais métodos o método Scrum, e o método XP (*Extreme Programming*). Nesta ocasião, estabeleceram-se princípios comuns compartilhados por todos os métodos ágeis de desenvolvimento de software. Foi então criada a aliança ágil e o Manifesto Ágil.

## 2.3 Manifesto Ágil

O Manifesto Ágil sugere que seja dada ênfase a alguns valores em detrimento de outros durante o processo de desenvolvimento de software. Indivíduos e interações mais que processos-e-ferramentas. Software em funcionamento mais que documentação-abrangente. Colaboração com o cliente mais que negociação de contratos. Responder a mudanças mais que seguir um plano. Ou seja, os valores tradicionais de desenvolvimento não são excluídos como um todo, apenas são colocados em um segundo plano visando a agilidade no processo.

Comparadas a outras metodologias, o Scrum e o XP produzem pouca documentação, pois documentam apenas o essencial visando otimizar o processo. Geralmente são mais utilizadas em projetos que: exijam muitas mudanças; os requisitos sejam passíveis de alterações; a recodificação do programa não acarreta um custo elevado; a equipe é pequena; o desenvolvimento rápido é fundamental.

## 2.4 Extreme Programming (XP)

Segundo o site devmedia [DEV15], a XP é mais indicada para equipes pequenas e médias que baseam-se na rápida mudança nos seus requisitos. Se diferencia das metodologias tradicionais principalmente pelo *feedback* constante, a abordagem incremental e o encorajamento da comunicação entre as pessoas. Ou seja, necessita de um estreitamento entre as partes envolvidas no projeto, desenvolvedores e clientes devem estar em constante interação. E quanto a abordagem incremental, significa que o projeto será desenvolvido em partes que funcionarão independentes. Um protótipo inicial é criado e a partir dele as demais funcionalidades vão sendo adicionadas, assim se contrapondo ao modelo tradicional que estrutura todo o projeto e entrega um produto final, o que às vezes esgota os prazos devido a complexidade do projeto.

O XP objetiva principalmente a agilidade no desenvolvimento e a consequente satisfação do-cliente.

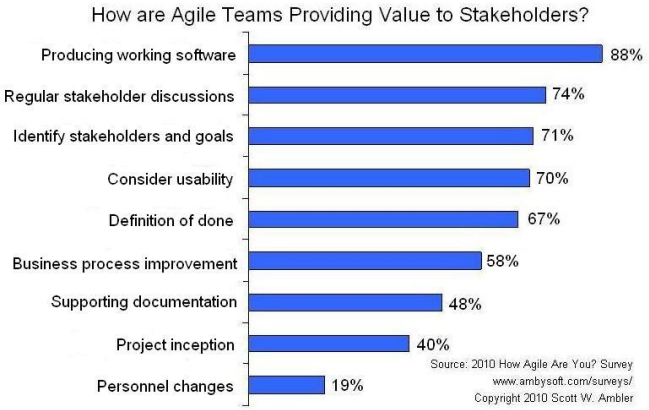


Figura 01

Na figura 01 é possível observar um gráfico que mostra os principais benefícios obtidos com a metodologia em questão.

## 2.5 Scrum

Segundo o site Desenvolvimento Ágil [DES15], Scrum é uma metodologia ágil para gestão e planejamento de projetos de softwares. Basicamente os projetos são divididos em ciclos mensais denominados *Sprints*. Estes são nada mais que iterações, divisões do trabalho para facilitar o processo, uma vez que há um enfoque parcial do projeto, definindo objetivos curtos, estes por sua vez, mais fáceis de serem atingidos do que todo o projeto em uma etapa apenas. As funcionalidades a serem implementadas estarão contidas em uma lista chamada *Product Backlog*. No início de cada ciclo é feita uma reunião rápida conhecida como *Sprint Planninng Meeting* onde o chefe da equipe, o *Product Owner*, define os itens que deverão ser implementados e a equipe define o que poderá entregar no *Sprint* em questão. A figura 02 explana como é conduzido este processo.

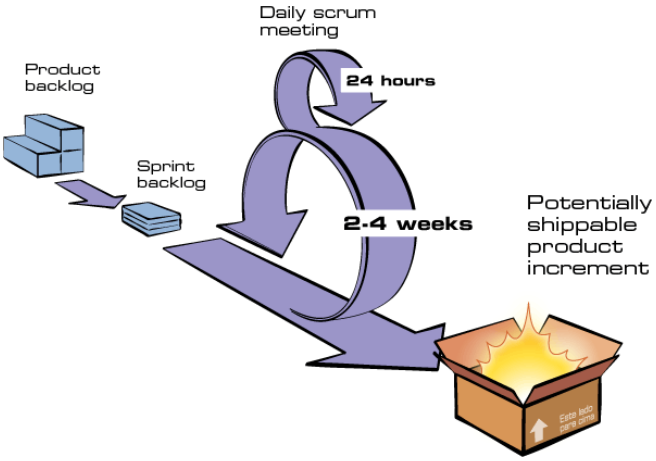


Figura 02

## 3. APLICAÇÃO

As aplicações dos métodos ágeis de desenvolvimento de software, Scrum e XP, embora apresentem diferenças, são de extrema importância em pequenas e médias empresas cujos requisitos variam constantemente. Segundo Cohen [COH04], tudo depende de três pontos chaves: Cultura, Pessoas e Comunicação. Onde Cultura quer dizer cultura da organização, que deve apoiar a negociação. Pessoas devem ser confiantes e competentes. A comunicação deve ser em um ambiente que facilite a rápida interação entre os membros da empresa.

Um dos fatores mais importantes para definir a aplicabilidade desse método é o tamanho do projeto, pois quanto maior o projeto, mais complicada a comunicação "olho-no-olho". Portanto, as aplicações desses métodos ficam restritas a pequenas equipes.
Para determinar a aplicabilidade dos métodos ágeis é necessária uma análise mais refinada do projeto, segundo Cohen [COH08], é preciso

prover o "filtro de aplicabilidade" para esse propósito.

A comparação entre os métodos ágeis revela que eles podem suportar diferentes fases do ciclo de vida de um produto em diferentes níveis de produção. Essas são características individuais do desenvolvimento ágil que são usadas como critério de seleção de aplicabilidade. Em certos cenários o desenvolvimento ágil é uma questão a ser discutida ainda. Em equipes com menos de 20 desenvolvedores, por exemplo, esforços de desenvolvimento distribuído. Com isso, Boehm [BOE04] sugere a análise de risco para escolher entre métodos ágeis e preditivos (dirigidos pelo planejamento). Sugerem que cada método possui seu ambiente ideal. Enquanto o desenvolvimento ágil requer um ambiente de baixa criticidade, desenvolvedores com no mínimo 4 a 6 anos de experiência no mercado, mudanças frequentes no escopo do projeto, equipe que saiba lidar com o "caos". Já o ambiente de desenvolvimento preditivo ou dirigido pelo planejamento requer um ambiente de alta criticidade, desenvolvedores com no mínimo dois anos de experiência no mercado, poucas mudanças no escopo do projeto, grandes equipes cuja cultura procura a ordem, ou seja, equipes que se mantêm constantes.

## 4. FERRAMENTAS

Diferentemente dos métodos de produção de software convencionais, o desenvolvimento ágil não é focado em ferramentas e processos, mas sim nos indivíduos e interações. Baseado nisso, ferramentas como VIMEE, *RemotePP*, CVW, e programas similares auxiliam no processo de desenvolvimento ágil em paralelo ao IDE. Essas ferramentas visam uma integração maior entre a equipe para criar um ambiente colaborativo, em que os programadores podem comunicar-se e trabalhar de forma intercalada. As funcionalidades que esse tipo de ferramenta geralmente possui são: IM (mensagens instantâneas), comunicação via áudio, gerenciamento de tarefas e agenda, visualização dos artefatos produzidos e compartilhamento de documentos. O que diferencia é como tais ferramentas são utilizadas no contexto do desenvolvimento ágil, pois estas ferramentas de desenvolvimento distribuído de software dão suporte aos processos dinâmicos aplicados ao desenvolvimento ágil. Tais ferramentas auxiliam nas atividades rotineiras do método ágil adotado, pois contribuem na interação entre os indivíduos e consequentemente, tornam o processo mais produtivo.

A colaboração e integração da equipe em um projeto é parte essencial do processo de software baseando em desenvolvimento ágil. Quando o desenvolvimento desses softwares é distribuído, a produtividade dos mesmos aumenta em nível. Por essa demanda foram elaborados softwares e ferramentas com o intuito de deixar a produção de software mais colaborativa e iterada, deixando as equipes ou pares cada vez mais entrosados.

Para Bohen [BOE04], as ferramentas colaborativas geralmente são utilizadas paralelamente ao IDE, entretanto, as recompensas são de grande valor. Enfatiza-se: redução de desgaste, melhoria no senso de contexto, além de contribuir para a rastreabilidade entre artefatos de colaboração.

O autor ainda destaca que, o meio de colaboração deve ser adequado ao contexto do trabalho em que o projeto se encontra, para que atendam as necessidades da equipe e também levem em consideração o contexto do trabalho em que o projeto se encontra para adequá-las às quais são mais aplicáveis à situação e considerar também o esforço individual com o esforço em equipe.

## 5. BENEFÍCIOS

Os métodos Scrum e XP visam elevar a qualidade do processo de software culminando em um produto elevada qualidade.

No XP, as necessidades são adaptativas. Dessa forma, se adaptam a novos fatores durante o desenvolvimento do projeto, ao invés de tentar analisar previamente todas as situações de contingência durante o desenvolvimento. Esta análise prévia é sempre difícil e representa alto custo, além de tornar-se um problema quando forem necessárias alterações no planejamento.

A mudança em si não caracteriza um problema, mesmo porque ela ocorrerá de qualquer forma. O problema é como receber, avaliar e responder a elas. Em uma metodologia tradicional, um software poderá ser construído por inteiro e depois se descoberto que ele não atende o propósito para que foi desenvolvido As metodologias ágeis trabalham com constante *feedback*, o que permite adaptar-se rapidamente a eventuais mudanças nos requisitos.

Estas alterações são, muitas vezes, críticas nas metodologias tradicionais, que não apresentam meios de reagir às mudanças. Outro aspecto positivo das metodologias ágeis são as entregas constantes de partes operacionais do software. Desta forma, o cliente não precisará aguardar um longo período para obter o software funcionando e, notar que o mesmo não atende aos requisitos.

A integração e o teste contínuo também possibilitam a melhora na qualidade do software. Não sendo mais necessário existir uma fase para integração de módulos, uma vez que eles são continuamente integrados e eventuais problemas são resolvidos constantemente.

O Scrum, que trabalha com a complexidade de desenvolvimento de softwares através do controle de verificação, adaptação e visibilidade de requisitos de um processo baseado na experiência. Fazendo uso de uma série de regras e práticas, onde controle não significa autonomia para criar o que foi previsto, mas sim dirigir o processo para orientar o trabalho em direção a um produto com o maior valor agregado possível.

Para atingir esses objetivos, o Scrum emprega uma estrutura iterativa e incremental da seguinte maneira: no início de cada iteração, a equipe analisa o que deve ser feito e então seleciona aquilo que acreditam que pode se tornar um incremento de valor ao produto ao final da iteração. A equipe então realiza o desenvolvimento daquela iteração e ao final apresenta o incremento da funcionalidade, para que os stakeholders possam verificar e requisitar alterações no momento apropriado.

O coração do Scrum é a iteração. A cada ciclo a equipe analisa os requisitos e a tecnologia e suas habilidades. Então, dividem-se para construir e entregar o produto moldando-se diariamente às complexidades, dificuldades e surpresas que poderão surgir.

Este método é mais frequentemente usado para gerenciar softwares complexos e desenvolvimento de produtos, utilizando práticas: iterativo e incremental.

O Scrum aumenta significativamente a produtividade e reduz o tempo de benefícios quando comparado aos processos clássicos. Também possui processos que permitem que as organizações ajustem-se sem problemas às circunstâncias, em rápida evolução das necessidades, e produzam um produto que atenda às metas de negócios em evolução. De acordo com a Figura 01, consegue-se observar claramente como funciona esse sistema, ou seja, indivíduos e interações entre processos e ferramentas de software que trabalham sobre abrangente documentação, a colaboração do cliente sobre a negociação dos contratos respondendo as mudanças no decorrer do processo de produção do sistema.

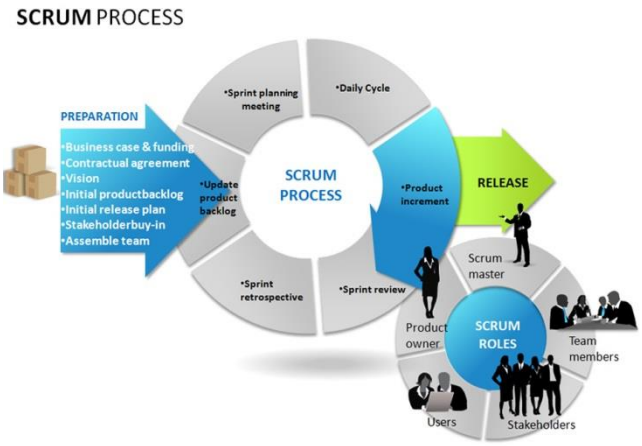


Figura 01 – Método Scrum


Figura 02 – Ciclos Metodologia Scrum

A figura 02 evidencia a divisão do *Product Backlog* em pacotes menores, seguindo dos *sprints* e posteriormente de uma avaliação dos resultados obtidos.

Um processo ágil Scrum beneficia a organização. Ajudando a aumentar a qualidade das entregas, reagir melhor às mudanças, proporcionando estimativas otimizadas, obtendo um controle do cronograma do projeto, ou seja, respeitando os prazos. Como resultado, os projetos que utilizam este método alcançam taxas mais elevadas de satisfação do cliente.

## 6. DESAFIOS

São inúmeras as vantagens atribuídas às metodologias ágeis. Porém, deve-se levar em consideração os objetivos que pretendem ser atingidos. O projeto em questão também deverá ser analisado, bem como sua equipe, pois os métodos de desenvolvimento ágil de software possuem algumas particularidades que poderão inviabilizar sua implementação.

Necessita da aplicação de boas práticas de projeto e programação, também e uma relação estreita entre cliente e desenvolvedor.

E, funcionam melhor em determinadas situações.

Embora existam conferências dedicadas ao assunto há alguns anos, o número de publicações sobre Scrum e XP ainda é reduzido quando comparado ao de outras áreas da Informática. Na indústria, os resultados de sua adoção são animadores, mas a comunidade científica tem demonstrado um posicionamento cético visto que diversas práticas propostas contrariam conceitos amplamente difundidos e utilizados tanto nas universidades, quanto na indústria.

## 7. CONCLUSÕES

As metodologias ágeis são uma realidade recente e surgem como uma opção as limitações no desenvolvimento tradicional de software.

Neste trabalho abordamos uma perspectiva dos métodos ágeis Scrum e Extreme Programming (XP). Foi possível entender sobre as formas que estes métodos são executados, seus principais benefícios e quais as melhores situações que poderão ser utilizados. Apesar do conhecimento ainda superficial do tema e da forma prática da implementação destes métodos, o trabalho nos proporcionou uma familiarização com este tipo de metodologia que poderemos utilizar em futuros projetos de Engenharia de Software.

## REFERÊNCIAS

[PRE10] PRESSMAN, R. S. Engenharia de Software. 6. ed. São Paulo, SP: McGraw-Hill, 2010.

[SOM07] SOMMERVILLE, I. Engenharia de Software. 8.ed. São Paulo, SP: Pearson Prentice Hall, 2007.

[COH04] COHEN, D., LINDVALL, M., & COSTA, P. (2004). An introduction to agile methods. In*Advances in Computers* (pp. 1-66). New York: Elsevier Science.

[ABR03] ABRAHAMSSON, P., WARSTA, J., SIPONEN, M.T., & RONKAINEN, J. (2003). New Directions on Agile Methods: A Comparative Analysis. *Proceedings of ICSE'03*, 244-254

K. Beck. *Extreme Programming Explained: Embrace Change*. Boston, MA: [s.n.], 1999. 157 p.

[BOE04] B. BOEHM. *Balancing Agility and Discipline: A Guide for the Perplexed*. Boston, MA: Addison-Wesley, 2004. 55-57 p.

[DEV15] Devmedia, “Conceitos básicos sobre Metodologias Ágeis para Desenvolvimento de Software (Metodologias Clássicas x Extreme Programming). Último acesso: Junho, 2015. Disponível-em: *http://www.devmedia.com.br/conceitos-basicos-sobre-metodologias-ageis-para-desenvolvimento-de-software-metodologias-classicas-x-extreme-programming/10596*

[DES15] Desenvolvimento Ágil, “Scrum”. Último acesso: Junho, 2015. Disponível em: *http://www.desenvolvimentoagil.com.br/scrum*