Desenvolvimento Ágil

Introdução

A indústria de desenvolvimento de software tem se tornado uma das mais importantes indústrias do nosso tempo. Empregando milhares de trabalhadores em todos os países do mundo, essa indústria cria alguns dos produtos mais essenciais que nós utilizamos para manter o nosso estilo de vida

Na intensa competitividade no ambiente de desenvolvimento de software, o que separará os melhores dos piores, os vencedores dos perdedores? Esta resposta é simples: a sua habilidade de criar e entregar mais rapidamente os produtos de software com mais qualidade e que melhor reflitam as reais necessidades dos clientes.

METODOLOGIA ÁGIL DE DESENVOLVIMENTO

O desenvolvimento de software precisa ser reconhecido como um processo imprevisível e complicado. Reconhecer que um software nunca foi construído da mesma forma, com a mesma equipe, sob as mesmas circunstâncias antes é a grande mudança do pensamento tradicional de desenvolvimento de software. Mas, o mais importante é reconhecê-lo como um processo empírico: que aceita a imprevisibilidade e tem mecanismos de ação corretiva.

Uma característica das metodologias ágeis é que elas são adaptativas ao invés de serem preditivas. Dessa forma, elas se adaptam a novos fatores durante o desenvolvimento do projeto, ao invés de tentar analisar previamente tudo o que pode ou não acontecer no decorrer do desenvolvimento.

História

O termo “Metodologias Ágeis” tornou-se popular em 2001 quando um grupo de dezessete especialistas em processos de desenvolvimento de software decidiu se reunir nos EUA, para discutir maneiras de melhorar o desempenho de seus projetos.

Foi então criada a Aliança Ágil e o estabelecimento do Manifesto Ágil contento os conceitos e princípios comuns compartilhados por todos esses métodos. Desde então o termo Desenvolvimento Ágil passou a descrever abordagens de desenvolvimento que seguissem estes conceitos que implicam em

Valorizar

- Indivíduos e interação entre eles são 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.

O Manifesto Ágil não rejeita os processos e ferramentas, a documentação, a negociação de contratos ou o planejamento, mas simplesmente mostra que eles têm importância secundária quando comparado com os indivíduos e interações, com o software funcionando, com a colaboração com o cliente e as respostas rápidas a mudanças e alterações.

O Manifesto Ágil

É muito importante entender que os conceitos do manifesto ágil definem preferências e não alternativas no desenvolvimento de software, encorajando a focar a atenção em certos conceitos sem eliminar outros. Assim para seguir os conceitos ágeis deve-se valorizar mais a certas coisas do que a outras.

1. Indivíduos e interação entre eles mais que processos e ferramentas:

Os softwares não são construídos por uma única pessoa, eles são construídos por uma equipe, então elas precisam trabalhar juntas (incluindo programadores, testers, projetistas e também o cliente). Processos e ferramentas são importantes, mas não são tão importantes quanto trabalhar juntos.

2. Software em funcionamento mais que Documentação abrangente:

a documentação deve existir para ajudar pessoas a entender como o sistema foi construído, mas é muito mais fácil entender como o funcionamento vendo o sistema funcionar do que através de diagramas que descrevem o funcionamento ou abstraem o uso.

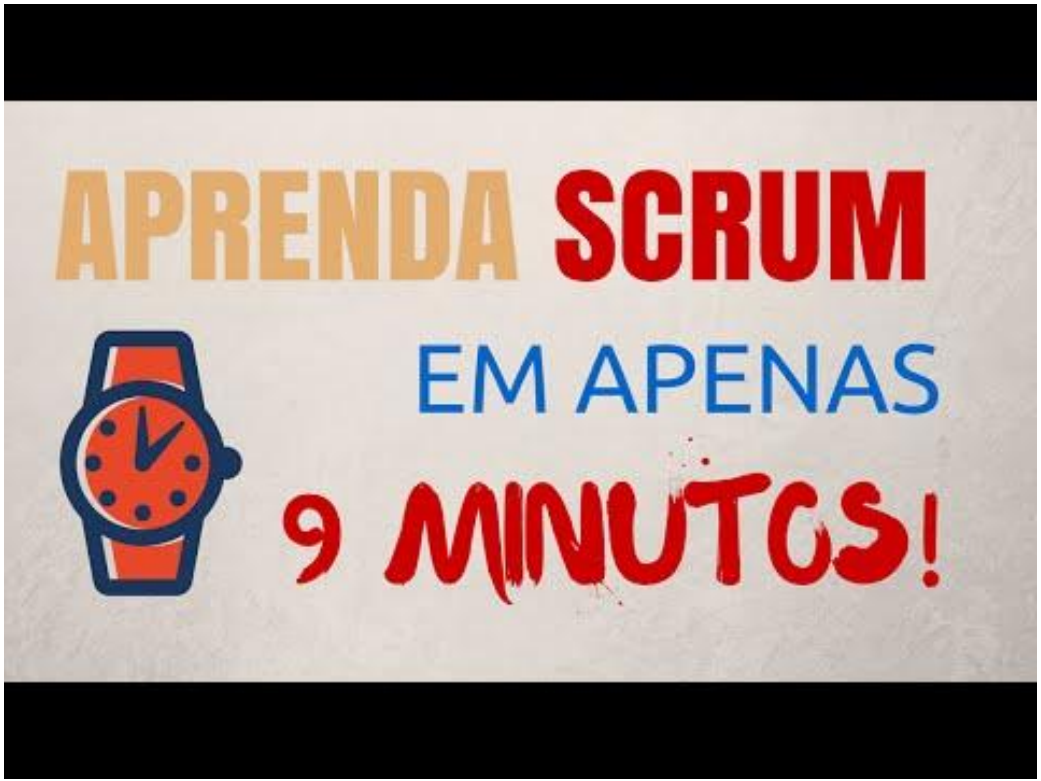
3. Colaboração com o cliente mais que negociação de contratos:

Somente o cliente pode dizer o que ele espera do software e normalmente eles não sabem explicar exatamente o que eles esperam e ainda, eles mudam de idéia ao longo do tempo e conforme eles vêm o software funcionando. Ter um contrato é importante para definir as responsabilidades e direitos mas não deve substituir a comunicação. Trabalhos desenvolvidos com sucesso têm constante comunicação com o cliente para entender suas necessidades e ajuda-los a descobrir a melhor forma de expressa-las

4. Responder a mudanças mais que seguir um plano:

mudanças é uma realidade no ambiente de negócios e elas acontecem por inúmeras razões: as regras e leis sofrem alterações, as pessoas mudam de idéia e a tecnologia evolui. O software precisa refletir essas mudanças. Um projeto de software certamente deve ter um plano mas ele deve ser flexível o suficiente para comportar as mudanças quando elas aparecerem, senão ele se torna irrelevante.

SCRUM



https://www.youtube.com/watch?v=XfvQWnRgxG0&t=281s

ESTÓRIAS DE USUÁRIO

As Estórias de usuários são fundamentais para a compreensão das necessidades do negócio e auxiliam a equipe na buscar de um objetivo comum, por simularem situações reais de usuários, como:

O vendedor quer cadastrar um cliente

As estórias podem ser registradas em cartões e terem prioridades associadas, como dos requisitos de negócio convencionais.

O vendedor quer cadastrar um cliente

Prioridade: **3 Pontos**

As estórias podem ser detalhadas e posteriormente terem sua prioridade alterada, conforme exemplo abaixo:

O vendedor deve poder cadastrar um cliente. O sistema não poderá gravar um cliente já cadastrado. O CEP e CPF devem ser validados.

Complexidade: **5 PONTOS**

Como pode ser observado, quanto mais detalhada a estória, mais fácil ficam as fases posteriores, de desenvolvimento e testes, por exemplo.

TDD - DESENVOLVIMENTO DIRIGIDO POR TESTES

O TDD é também uma prática ágil de desenvolvimento, que resulta em uma suíte automatizada de testes unitários e permite praticar um desenvolvimento orientado a testes, onde antes de desenvolver efetivamente, se define expectativas do software, buscando atender a necessidade do usuário, como apresentado no exemplo de estórias, antes de criar o requisito, foi identificada a necessidade de validar CEP e CPF, por exemplo, rotinas as quais podem ser automatizadas.

ATDD - DESENVOLVIMENTO DIRIGIDO Á TESTES DE ACEITAÇÃO

ATDD é uma variação do TDD, mas que dirige seu processo de construção baseado em testes de aceitação.

Para entender na prática, vamos definir antes o que são critérios de aceitação.

Trata-se de uma lista de regras para atender requisitos definidos, no caso, podem ser estórias definidas.

Baseado na estória do vendedor cadastrar um cliente, podemos exemplificar os critérios como sendo:

Não gravar um cliente já cadastrado;

CEP válido;

CPF válido.

Sendo esses os critérios básicos a serem atendidos. Destacando que esses são apenas exemplos, que podem ser detalhados e quebrados em estórias menores, conforme a necessidade identificada pela equipe ou pelo negócio.

BDD - DESENVOLVIMENTO ORIENTADO A COMPORTAMENTO

Como todos os modelos ágeis, o BDD é baseado em usuários, buscando agregar valor de negócio, através de um vocabulário comum, possibilitando comum entendimento entre TI (Tecnologia de Informação) e Negócio.

Cria-se comportamentos baseados no usuário, facilitando a compreensão de todos os envolvidos.

Esse modelo defende que o negócio e tecnologia devem “falar” a mesma língua, também que o valor de “negócio” deve estar claro e que nada deve ser construído precocemente, sem definir o comportamento do usuário.

Os pilares do BDD são “Dado” (Given), “Quando” (When) e “Então” (Then).

Para melhor entender, vamos exemplificar utilizando o exemplo anterior, baseado em um critério de aceitação (CA):

CA1 - Não gravar um cliente já cadastrado.

DADO: um cliente;

QUANDO o cliente já estiver com CPF cadastrado em outro cliente;

ENTÃO o sistema emite uma mensagem que já existe cliente cadastrado com esse CPF;

E não salva o cadastro;

E retorna a tela de edição.

Como pode ser observado, quando detalhado o CA, baseado no BDD, é possível visualizar o comportamento do usuário e mesmo não havendo sido descrito uma linha de código, a lógica fica nítida, o que facilita as fases posteriores.

Exercício

1. Escolha uma das metodologias expostas e crie uma proposta do por que adotá-la

2. Elaborar um resumo de mais alguma metodologia ágil e com um exemplo de como aplicá-la