Juliana Baptista dos Santos França (D) (1) AV

Tópico 1:

Processos ágeis surgem com o desafio de apoiar o desenvolvimento de software, introduzindo agilidade e atendendo de forma mais direta as mudanças incorporadas ao software em virtude do mundo real complexo que é instanciado.

Desenvolver software é um processo criativo e que envolve ações de Engenharia. Mesmo com sua complexidade de desenvolvimento, o software (sua construção) é apoiada por processos a fim de orientar as ações e trazer ordem ao caos.

Para a construção de produtos de software de alta qualidade, sabe-se que ações de controle precisam ser aplicadas nas fases do processo de software. Ainda falando em qualidade, sabe-se que quanto mais próximo estivermos do cliente/solicitante do projeto de software, maiores são as chances de levantarmos Requisitos de qualidade.

Os processos ágeis propõem uma interação mais direta e de confiança com o cliente, entendendo também que a ansiedade e expectativas do cliente precisam ser controladas com entregáveis pequenos e constantes.

O manifesto ágil trás princípios gerais que fundamentam o parágrafo anterior. Foram 4 princípios gerais que integraram da experiência e prática dos envolvidos. Sendo eles: Software entregue mais do que documentação abrangente, Resposta a mudança mais do que seguir um plano, Colaboração com o cliente mais do que negociação de contrato, produzir (comunidade) software é mais do que processos e ferramentas.

O Manifesto Ágil estendo seus princípios em mais 12 Aw (2)
como: equipes auto organizáveis, software (partes) funcionais entregues em 2 semanas a 2 meses, clientes/envolvidos participantes ativos neste processo, equipes e projetos sustentáveis, equipes engajadas ao projeto, forte comunicação entre os envolvidos.

A literatura e a prática apontam diferentes abordagens que assumem em suas técnicas e ações os princípios definidos no manifesto Ágil. Exemplos dessas abordagens temos o SCRUM, XP, Kanban, Lean, Scrumkanban.

O Scrum é hoje a abordagem mais popular e difundida. Ela tem por fundamento entregas funcionais em pequenos períodos; ações de melhorias para as suas práticas e produto; e relacionamento integrado entre time/cliente/usuários, ou seja, stakeholders. No Scrum existem 3 papéis muito bem definidos: o Scrum Master o product owner (PO) e Development team. Nele as equipes/times são pequenos, contendo de 3-9 colaboradores. Mínimo de 3 para haver colaboração.

O Scrum Master é responsável por atuar como um facilitador no processo de desenvolvimento. Além de apoiar questões técnicas que surgem, é seu papel buscar um bom ambiente de interação entre os envolvidos. O PO é o "dono do produto" é ele quem tem informações do negócio. Já a equipe de desenvolvimento é a responsável pela operacionalização da construção do software. Nas equipes de desenvolvimento não há papel definido como design, testador, arquiteto.... Todos os colaboradores podem atuar nas diferentes frentes de trabalho.

O Scrum é organizado em Ações. Algumas delas ocorrem em reuniões periódicas como daily, planning, Review e Retrospectiva.

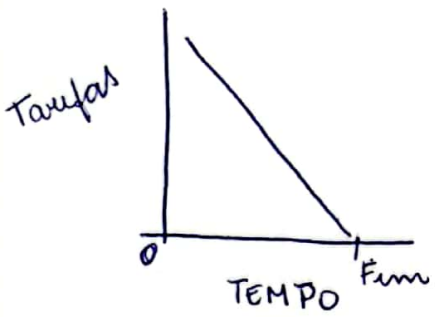
O SCRUM é organizado em Sprints com duração média de 2 a 4 semanas (podendo chegar a mais tempo). Os artefatos gerais que estão inseridos no SCRUM são: o backlog do produto, o backlog da sprint, os gráficos de acompanhamento e controle e o entregável.

No backlog do produto são inseridas as necessidades/requisitos levantados com o cliente. No caso de processos ágeis estes são caracterizados/definidos como histórias do usuário. Essas histórias detalhadas/especificadas são incluídas no backlog da sprint ao serem priorizadas para a sprint corrente.

No backlog do produto é informado para cada História a sua complexidade, descrição e priorização. No início de cada sprint ocorre a reunião de planejamento com a participação do time de desenvolvimento, scrum master e PO. Nesta reunião são apresentadas as prioridades e o time discute sobre as possibilidades quanto a parte técnica. Cada colaborador do time se auto aloca em uma ou mais tarefas se comprometendo a cumpri-las durante a sprint. Essas tarefas são deslocadas para o backlog da sprint e a sprint é iniciada efetivamente. A cada dia ocorre a reunião diária com a equipe. Essa reunião tem duração máxima de 15 minutos e todos os envolvidos permanecem de pé e relatam o que foi feito no dia anterior, o que fará no dia corrente e se estão enfrentando algum problema.

Ao final da sprint ocorre a meeting review onde o entregável é apresentado. A reunião de retrospectiva normalmente ocorre em marcos do projeto ou na entrega final. Nela são discutidas as lições aprendidas, o que deu certo, o que deu errado. É o que chamam de "Lavagem de Roupa Suja". Na reunião de retrospectiva, apenas a equipe interna participa.

Para o contrôle do processo de desenvolvimento, o Scrum prevê o gráfico burnout. Nele é apresentada a relação entre o tempo do projeto e as tarefas desenvolvidas. A estrutura base do gráfico é apresentada abaixo:



Estruturas diferentes do gráfico ao lado podem representar problema como atraso no projeto.

A abordagem XP segue o mesmo princípio da agilidade. Ela tem como fundamentos a melhoria do processo, equipes engajadas entregáveis funcionais em curto espaço de tempo. Uma prática muito comum ao Extreme programming (XP) é o pair programming. Esta prática prevê o desenvolvimento em pares de uma dada tarefa de desenvolvimento. Dessa forma os profissionais podem discutir sobre uma questão complexa e chegarem juntos a uma solução de maior qualidade.

O pair programming é uma técnica aplicada também em outros processos de software, ou com o objetivo de potencializar a qualidade, ou para impulsionar novos colaboradores no time.

O Kanban foi popularmente difundido em função de seu quadro de acompanhamento e contrôle das tarefas de desenvolvimento. Nele são dispostas as histórias de usuário do produto, as histórias selecionadas p/ desenvolvimento atual, aquelas que estão efetivamente em desenv., as em teste e as finalizadas. Exemplo do quadro segue.

Backlog	Historia selecionada	Análise		Construção		Teste	Done
		Em execução	feito	Em execução	Feito		
⊛ HU. 1	HU 1						
HU. 2	HU 3						
HU. 3							
⋮							
HU. N							

⊛ H.U. → Historia do usuário

O Scrumkanban alinha o mesmo discurso e práticas defendidas pelo kanban, incorporando as reuniões previstas no Scrum como as reuniões diarias, de planejamento, retrospectiva e revisão. Já o Lean surgiu na indústria automobilistica com o objetivo de introduzir agilidade ao processo, evitando desperdícios na construção dos produtos.

Os processos ágeis de software são também conhecidos como processos leves. No entanto, sua leveza não é sinônimo de caos. Embora seus princípios tenham uma relação franca e direta com os envolvidos, reduzindo o peso de produção de documentos, modulos, contratos e especificação, isso não significa que não há documentos do projeto. Os processos ágeis prezam por ações efetivas que retornem valor ao projeto e aos envolvidos. Dessa maneira, entende-se que decisões do projeto precisam estar registradas para manter o histórico para ações e projetos futuros, a especificação das histórias são conduzidas, os contratos existem.

Outra questão que gera discussão quanto ao produto gerado através de processos ágeis, é quanto a seus testes. Os testes de aceitação junto ao usuário são comuns, mas para estabelecer agilidade ao processo, muitas equipes direcionam seus testes unitá-

vios e de integração através de testes automatizados.

. Os processos ágeis de software são atualmente populares em função de seus princípios e práticas de agilidade, com entregáveis funcionais a cada incremento | sprint. No entanto, ele não é a solução para todos os tipos de projeto de desenvolvimento de software. Projetos de alta complexidade versam pelos métos formais; os projetos mais estáticos e simples, um simples processo em cascata (comunicação — planejamento — modelagem — construção — entrega) seria suficiente e com custo mais controlado.

Estratégias como a prevista no XP onde um colaborador só pode iniciar uma nova historia de usuário quando tiver finalizado a tarefa que está em sua responsabilidade, ou mesmo o poker-planning aplicado no SCRUM para estimar a complexidade das histórias de usuário, são iniciativas que apoiam não apenas a agilidade, mas incorporam controle e qualidade ao processo de desenvolvimento ágil.

Dessa maneira, é possível concluir que os processos ágeis de software, além de introduzirem agilidade e serem leves e objetivos, eles também previnem ações de controle e qualidade de suas atividades do processo e produto gerados.

Tópico 3:

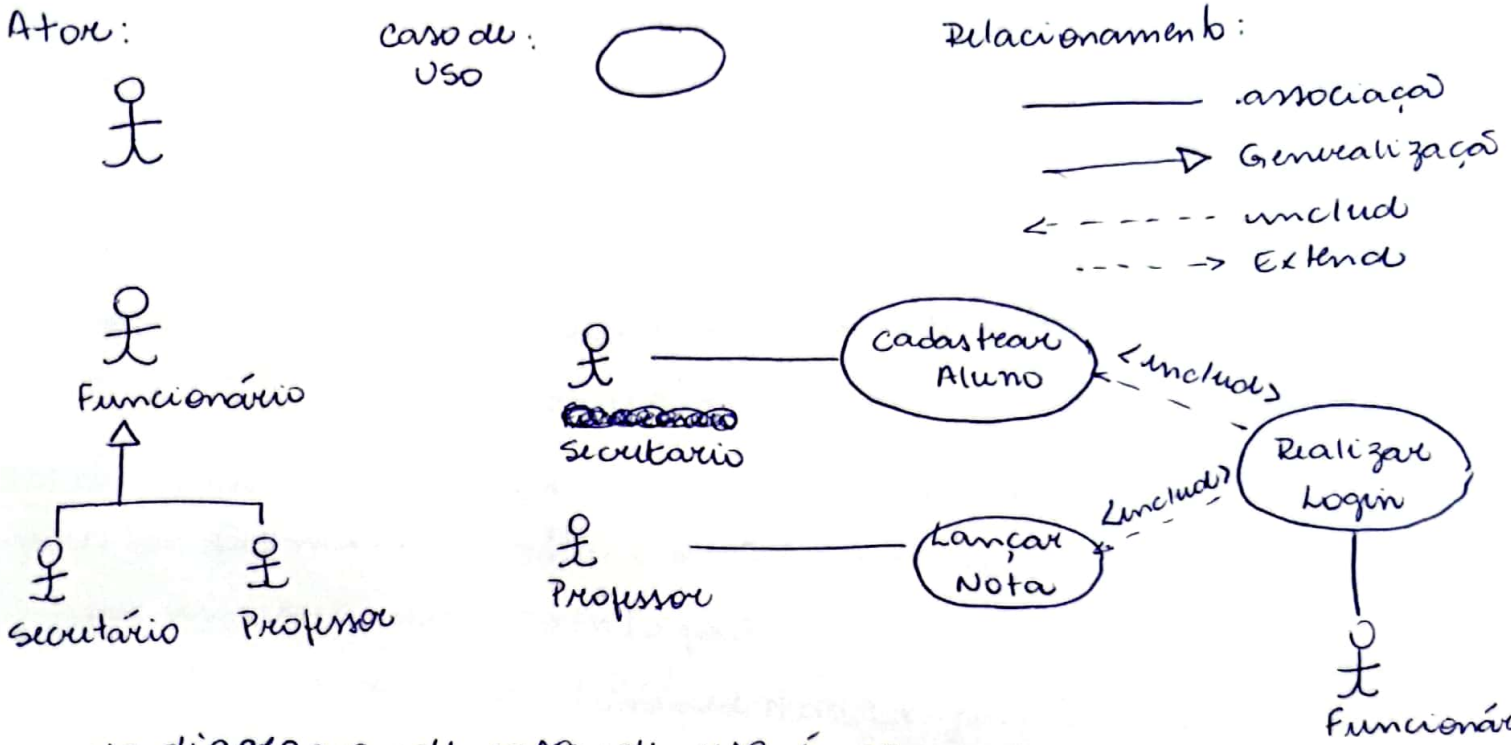
Os casos de uso representam as interações previstas entre os envolvidos com o software e as funcionalidades (ações). Considerando as fases básicas do processo de software definidas por Pressman: Comunicação – planejamento — modelagem — construção e entrega, os casos de uso habitam a fase de modelagem considerando os Requisitos previamente levantados.

Os casos de uso descrevem o cenário de operação que será realizado pelos desenvolvedores no projeto e implementação do software. Os casos de uso são representados em um diagrama de casos de uso e na sua descrição. Eles podem ser casos de uso essencial ou Real.

Os casos de uso essencial não apresentam em sua descrição termos tecnológicos, estando portanto mais próximos do cliente/negócio. Já os casos de uso Real, vemos em sua descrição termos e ações associadas a tecnologia como tipo de banco de dados, linguagem de programação, ou mesmo recursos de interface envolvidos na interação.

O diagrama de casos de uso é composto por elementos que formam sua notação, sendo eles: Atores, casos de uso, e relacionamentos. Os atores são aqueles que interagem/se comunicam com os casos de uso(ações) eles podem ser pessoas(usuários), Agentes, Organizações, ou outros sistemas. O caso de uso representa uma ação do sistema que é derivado dos requisitos levantados. Os relacionamentos podem ser: comunicação, herança, extensão, inclusão. O de comunicação representa a interação direta entre o ator e o caso de uso, a herança/generalização pode ocorre entre casos de uso e entre atores, onde um ator filho, herda as características de um ator pai, por exemplo.

A extensão (extend) e inclusão (includ) permitiu uma melhor organização e reuso na fase de projeto. O extend apresenta ações a serem chamadas por outro caso de uso, sem obrigatoriedade. O Includ, inclui um caso de uso em outro, o chamado obrigatoriamente. Exemplos da notação do diagrama de casos de uso seguem.



No diagrama de casos de uso é apresentado em um Retangulo a fronteira do sistema.

Uma vez definido o diagrama de UC, é possível partir p/ a sua descrição. Ela pode ser narrativa ou sequencial. Considerando a sequencial, são previstos alguns passos para serem definidos como: nome do caso de uso, identificador, pré condição, atores envolvidos, fluxo básico, fluxos alternativos, pontos de extensão. Na descrição do caso de uso, em especial no fluxo básico e alternativos, deve haver uma orquestração entre ator e sistema. Ex.:

Fluxo Básico Cadastrar Aluno

1. O ator solicita cadastrar aluno
2. O sistema exibe as informações a serem completadas
3. O ator preenche as informações

n - Caso de uso termina

Requisito é uma necessidade ou desejo do cliente, segundo Pressman e Maxim. Eles podem ser classificados como Requisitos funcionais (Representam ações do sistema), Requisitos de negócio (necessidades do negócio), Requisitos não funcionais (qualidade ou restrições), e Requisitos inversos (aqueles requisitos que não serão implementados no projeto/sistema).

A Eng de Requisitos prevê 7 fases: concepção, levantamento, Elaboração, Negociação, Especificação, Validação, e Gestão. Nas fases de Elaboração, Negociação e Especificação, há a produção do documento de requisitos que é iniciado com os requisitos levantados, além de informações do software como um todo (viabilidade, estado atual, solução desejada), e esses requisitos são detalhados em uma especificação que pode ser em linguagem natural, gráficos e modelos. Sistemas de alta complexidade envolvem esses diferentes tipos em sua documentação.

Um dos modelos UML (Unified Modal Language) aplicados nessa especificação são os modelos de casos de uso do tipo essencial. Os casos de uso são derivados dos requisitos funcionais. Essa modelagem mostra os cenários de interação entre os atores e as funcionalidades (Traduzidas nos casos de uso). Como a descrição do diagrama de casos de uso, nesta fase do projeto de software, faz pouca ou nenhuma menção à tecnologia, facilita a interação com o cliente. Dessa maneira, a modelagem de caso de uso pode apoiar a comunicação com o cliente no processo de especificação detalhada dos requisitos. Com este instrumento é possível detalhar mais as informações de cadastro, por exemplo, o processo de interação entre sistema e ator, a descoberta de novas regras de negócio. É possível também validar os

requisitos junto ao cliente e também com os demais especialistas na revisão técnica. Na revisão técnica, representantes dos stakeholders validam a documentação de requisitos gerada na intenção de buscar ambiguidades, dificuldade de entendimento, ausência de informações, conflitos. A modelagem de casos de uso trás o cenário especificado e com riqueza de detalhes que apoiam a busca por inconsistências e ausencia de informação.

Conclui-se que a modelagem de casos de uso trata-se de um forte instrumento de apoio ao processo de Engenharia de Requisitos de software.

Tópico 9:

A análise de ponto de função é uma estratégia para estimar o tamanho, custo, complexidade do software, antes dele ser desenvolvido. Existe uma diferenciação nem sempre tão clara quanto aos termos Medição, medida e métrica. Medida é quando há a coleta de apenas um elemento; medição um mais de um, e métrica é o entendimento das medidas e medições gerando indicadores. A contagem de pontos de função nada mais são do que uma métrica de software.

A contagem de pontos de função independe da tecnologia ou linguagem de programação. A análise de pontos de função pode ser iniciada assim que os requisitos são levantados e especificados. Para esta Análise de pontos de função tecnicamente é preciso eliminar desta métrica funções internas do software. São de interesse para esta métrica apenas funções que provejam interações

entre atores (indivíduos, BD, Arquivos, Agentes ...).

Uma vez eliminadas as funções internas do software, inicia-se a contagem de Pontos de Função. Esta contagem de pontos de função ocorre em duas frentes, sendo elas:

- Funções de dados — AIE / ALI → Associados à Estrutura das funções
- Funções Transacionais → Associado às interações/comunicações

As funções de dados provem os arquivos lógicos internos e externos, enquanto que as funções transacionais provem as interações em interfaces e banco de dados, por exemplo.