ELIÂNAE NOGUEIRA TEIXEIRA

(F)

TÓPICO
1. PROCESSOS ÁGEIS DE SOFTWARE

O desenvolvimento de software utilizando metodologias ágeis tem ganhado cada vez mais espaço em um mundo caracterizado pela volatilidade, incerteza, complexidade e ambiguidade (V.U.C.A). A dinamicidade do mercado e as pressões de rapidez nas entregas, considerando agregação de valor ao negócio, têm gerado cada vez mais interesse pelo conceito de agilidade.

As metodologias ágeis tratam de uma mudança de mentalidade e comportamento para lidar melhor com mudanças constantes nos requisitos, nos negócios e nos ambientes. Tais metodologias são pautadas em valores e princípios descritos no Manifesto Ágil.

Os valores favorecem determinados elementos no processo de desenvolvimento que lidam com: interação e indivíduos; software funcionando; interação com o cliente e resposta a mudanças. Tais valores lidam com uma maior preocupação em um processo colaborativo, com equipes focadas e auto organizadas, com o cliente tendo papel fortalecido e participativo para constantes ciclos de feedback, de incrementos de software entregues frequentemente levando em consideração o retorno-benefício-valor ao negócio. Trata-se de um processo que busca abraçar mudanças por entender que as condições não são fixas e que adaptações precoam ser realizadas ao longo do desenvolvimento para que o resultado esteja cada vez mais próximo do desejado pelo cliente, tendo a sua satisfação como um dos critérios primordiais de qualidade.

Desta forma, ao invés de processos e ferramentas, documentação abrangente e planos rígidos entende que deve fortalecer pessoas e lidar com adequações ao plano a medida que o processo de desenvolvimento é realizado, ainda que considerem o valor dos itens mencionados.

Tratam-se de processos incrementais e iterativos com entrega de software funcionando (incrementos de valor) em ciclos rápidos e curtos, dentro de períodos de tempo estabelecidos (time-box).

Em geral, são propícios para projetos com equipes pequenas, coesas e auto-organizadas que participam ativamente de decisões, buscam a colaboração e cooperação, preferem estabelecer diálogos frequentes face à face, ainda que esse conceito esteja se adaptando devido ao cenário (pós)-pandemia e trabalho remoto com equipes distribuídas).

Tais equipes devem estabelecer e manter um ritmo sustentável, sabendo o quanto de trabalho pode ser entregue sem grandes pressões, gargalos que gerando uma carga de trabalho produtiva mais saudável e que possa estabelecer um ritmo a ser mantido e de certa forma com previsibilidade.

Tais metodologias lidam com colaboração, simplicidade, feedback, comunicação e flexibilidade (adaptação e apto a mudanças). Transparência, visibilidade também são características dessas metodologias.

Vários métodos (ou frameworks) têm sido propostos de forma a tratar tais propriedades, características e comportamentos através de práticas. Exemplos são SCRUM, XP, Kanban, FDD, Lean, dentre outros. De acordo com pesquisas recentes o SCRUM é um dos métodos mais difundidos e aplicados. Combinações são realizadas entre os métodos, por exemplo, derivando o SCRUMBAN (SCRUM + Kanban).

O SCRUM organiza o desenvolvimento em entregas em ciclos curtos.

Os requisitos são organizados em uma estrutura denominada de PRODUCT BACKLOG. Tais requisitos são descritos através de histórias de usuários e priorizados junto ao cliente. A ideia é ter descrições mais sucintas que serão detalhadas e reavaliadas ao longo do desenvolvimento. Grupos de requisitos com prioridade mais alta (pensando no valor a ser entregue p/ o negócio) são selecionados para serem desenvolvidos em ciclos curtos (SPRINTS)

A cada sprint são realizados planejamentos (SPRINT PLANNING) e reuniões de análise ao final do ciclo para avaliar andamento (SPRINT RETROSPECTIVE).

Outra prática adotada refere-se ao DAILY Meting uma reunião curta, diária, para breve discussão na equipe do que está sendo feito, do que será feito e de possíveis pontos maiores de análise. Tal reunião de certa forma discute o progresso e andamento do projeto pela equipe incentivando a comunicação, colaboração e confiança entre os membros.

O método conta ainda com alguns papéis: PRODUCT OWNER (representante do cliente para participação efetiva ao longo do desenvolvimento, sua presença deve ser constante e participativa não apenas consultiva em períodos difusos); SCRUM MASTER (atua como um líder para guiar a equipe de forma colaborativa, o papel representa o conceito de um coach-treinador); TEAM MEMBERS (membros da equipe).

O SCRUM também pode utilizar um quadro Kanban para visualizar o andamento dos itens do backlog dividindo por exemplo em estágios como TO DO, DOING, DONE, ou seja, preparado p/ iniciar desenvolvimento, em desenvolvimento e feito. O conceito de feito (PRONTO) deve ser bem estabelecido p/ entrega. Tais quadros permitem análises através de técnicas como BREAKDOWN CHARTS.

Além desses conceitos, metodologias ágeis adotam práticas de desenvolvimento dirigido a testes. A ideia do desenvolvimento de testes desde etapas iniciais adotando a descrição de casos de testes antes mesmo de iniciar a implementação. A metodologia também é adepta da automatização de testes como forma de agilizar tarefas mas garantindo qualidade. Desta forma, frequentemente ferramentas podem ser utilizados, por exemplo para derivação e apoio aos testes unitários (JUNIT).

A descrição dos requisitos frequentemente é feita através de histórias de usuários que utilizam ~~as~~ sentenças sucintas e curtas p/ descrição seguindo uma estrutura "Como um <papel> desejo <objetivo> para <razão>". Em geral, são descritas em cartões que ainda duem contar a importância e prioridade das histórias definidas junto ao cliente / parte interessada. Por isso, tal técnica conta com um elemento de conversação equipe-cliente para melhor comunicação e entendimento entre as partes. Tal interação conta ainda com a confirmação através da escrita de testes de aceitação, em geral, no verso do cartão como forma de aumentar a confiança e validação dos requisitos de acordo com expectativas e necessidades dos envolvidos. Para cada história uma estimativa deve ser realizada usando pontos de história. Em geral é usada uma sequência numérica (1, 2, 3, 5, 8, ...), similar a sequência de Fibonacci. A estimativa é realizada utilizando a técnica de planning POKER em que rodadas de estimativa são feitas pelos membros da equipe utilizando cartões até que se chegue a um consenso ou atinja um número máximo de rodadas. As estimativas são calibradas de acordo com a experiência da equipe e conhecimento do seu ritmo sustentável.

Devido as entregas constantes, uma técnica usada para auxiliar em integrações contínuas que também tem sido bastante usada é DEVOPS.

Vale ressaltar que não há bala de prata e que os processos ágeis podem não ser apropriados em qualquer contexto. Em geral, processos precisam ser adaptados para o contexto específico de utilização levando em consideração a aspectos como complexidade e características do projeto, experiência e tamanho da equipe, tipo de produto, volatilidade dos requisitos, aderência a modelos de maturidades, normas, padrões, leis, dentre outros. Ainda existe discussões sobre a escalabilidade de processos ágeis, alguns métodos têm sido propostos neste sentido (ex scaleAgile) para trabalhar projetos de grande proporção, equipe e complexidade. Métodos híbridos também têm sido discutidos para melhor combinar práticas dos métodos prescritivos / tradicionais com conceitos e práticas ágeis. De fato para processos com alto índice de mudança, dinamicidade, inovação, a agilidade pode trazer benefícios. Projetos mais previsíveis, com requisitos estáveis em contextos conhecidos podem se beneficiar de uma estrutura mais estável dirigida a planos (proc. prescritivos). Analisar cada contexto com conhecimento das boas práticas, métodos e ferramentas e foco na qualidade é uma boa opção.

TÓPICO

3. CASOS DE USO

Requisitos são descrições do que se espera do sistema, retratam as necessidades, expectativas dos clientes (partes involvidas) e restrições, propriedades, características que o sistema deve possuir. Desta forma, uma das classificações de requisito se dá em requisitos funcionais e não funcionais, respectivamente. A descrição de requisitos pode ser feita de diferentes formas e níveis de detalhes. De forma inicial, são declarações mais sucintas e usando linguagem natural (não-estruturada) de forma mais próxima ao domínio e entendimento do cliente. Métodos ágeis usam histórias de usuários para descrição dos requisitos funcionais. E outra forma de representar requisitos trata de uma descrição baseada em cenários denominada casos de uso. Casos de uso são representações de requisitos funcionais através de uma visão externa do sistema, caracteriza interações entre entidades externas e o sistema através de um conjunto de ações realizadas para atingir o objetivo do requisito descrito (função). Trata de uma perspectiva dinâmica de modelagem do comportamento do sistema. Busca representar cenários de uso com destaque para um fluxo central (principal) e mais detalhadamente alternativas e exceções. O objetivo central é que fique claro o objetivo (resultado a ser alcançado pela interação) e a sequência de passos/ações realizadas entre entidade e sistema.

A descrição pode ser feita de forma narrativa contínua, sequencial numerada, ou tabular (passos sendo distinguidos entre entidade e sistema). Além da sequência de passos/ações, podem conter dados complementares para refinamento/detalhamento de informação como: descrição e distinção de atores primários e secundários; pré e pós-condições de realização do caso de uso; restrições; regras de negócio; objetivo/propósito.

De forma central, descreve o fluxo principal de interação, também chamado de "cenário feliz", enumera a sequência de passos mais direta que leva ao objetivo/função, ou seja, sem exceções ou pontos de alternativas. Em um segundo momento, fluxos alternativo e pontos onde podem ocorrer erros/exceções são descritos de forma complementar ao fluxo principal com devidas sinalizações destes pontos de desvio no fluxo principal e quando o contrade de sequência de passos retorna ao mesmo, se for o caso.

De forma gráfica e visual, casos de uso são representados por diagramas de casos de uso que possuem como elementos: atores, casos de uso, e relações. Os atores, representados por 🯅, descrevem as entidades externas que interagem com o sistema, podendo ser papéis desempenhados por pessoas, entidades organizacionais (departamentos, instituições, etc), dispositivos, equipamentos ou outros sistemas. Casos de uso são representados por elipses com seu nome dentro da elipse, que devem ser concisos e declarar explicitamente a função. Em geral, podem ser usados verbos para designar ação. Por exemplo, em um sistema de biblioteca, poderíamos ter como atores usuário da biblioteca e bibliotecário e um caso de uso poderia ser Emprestar livro.
Como relações podemos ter: (1) comunicação - relação de ator c/ caso de uso; (2) generalização / herança - ator com ator e caso de uso com caso de uso → estabelece a relação da presença de características / propriedades comuns entre os elementos envolvidos. Exemplo: funcionário pode ser um ator pai de quem gerente herda características comuns; no entanto, gerente pode ter propried. específicas (especializações). (3) associações entre casos de uso: <<includes>> e <<extends>> Em geral, includes determina a relação de um caso de uso que apresenta um comportamento comum a diferentes situações exemplo: Identificar cliente pode ser usado em diferentes interações com um sistema de caixa eletrônico, ou sistema de venda de produtos. Já extends determina extensões / comportamentos alternativos, por exemplo, pagar fatura pode ser estendido p/ pagar com dinheiro, cartão de crédito, boleto.

Um exemplo simplificado de caso de uso:

CASO DE USO: FINALIZAR COMPRA

PRÉ-CONDIÇÃO: CLIENTE IDENTIFICADO NO SISTEMA

1. CLIENTE SOLICITA FINALIZAR COMPRA
2. SISTEMA CALCULA TOTAL E INFORMA AO CLIENTE
3. CLIENTE CONFIRMA COMPRA
4. SISTEMA CONFIRMA DADOS P/ ENTREGA DO PRODUTO
5. CLIENTE CONFIRMA CEP E ENDEREÇO
6. SISTEMA CALCULA FRETE
7. SISTEMA LISTA OPÇÕES DE PAGAMENTO
8. CLIENTE SELECIONA FORMA DE PAGAMENTO
9. SISTEMA VALIDA PAGAMENTO
10. SISTEMA FINALIZA COMPRA E ENVIA E-MAIL DE CONFIRMAÇÃO DA VENDA P/ CLIENTE CADASTRADO.

TÓPICO

9. Contagem de Pontos de Função

A análise de pontos de função é usada para realizar a estimativa de tamanho do software analisando as funções a serem providas que forneçam valor ao negócio. Estimativas de tamanho são difíceis de serem realizadas pela dificuldade inerente de realizar previsões, a falta de informações detalhadas sobre requisitos, falta de dados de projetos anteriores que sirvam de base (acrescida da dificuldade de análise da similaridade entre os projetos). Uma das técnicas mais conhecidas para cálculo de tamanho é a da contagem de linhas de código (LOC). No entanto, essa técnica lida com dificuldades sobre o que deveria de fato ser contado como linha (por exemplo, comentários deveriam ser excluídos dessa análise), diferenças entre linguagens de programação, aumento da complexidade, relação com a produtividade e experiência do desenvolvedor, para determinar de forma mais concreta depende-se do código em si (fase adiantada do desenvolvimento, dentre outros. A análise de pontos de função (APF) busca uma análise com foco nas funções do sistema. Tabelas de conversão podem ser posteriormente aplicadas para conversão do valor para linhas de código.

A contagem é primeiramente de pontos de funções não ajustados e posteriormente é aplicado um fator de ajuste a cada categoria de função analisada.

A contagem é realizada levando em consideração funções transacionais e funções de dados.

As funções transacionais se dividem em: saída externa, entrada externa e consulta externa. As funções de dados se dividem em arquivos lógicos internos e interfaces lógicas externas.

- Função transacional - saída externa: trata de informações de negócio a serem disponibilizadas aos usuários e para isso há algum tratamento do sistema para processar e disponibilizar a informação. Exemplos de itens a serem contados são relatórios, mensagens de erro, telas.

- Função transacional - entrada externa: trata de informações fornecidas pelo usuário ao sistema para processamento exemplos relacionados a dados a serem adicionados, alterados, excluídos. Em geral tais dados serão persistidos pelos arquivos lógicos internos.

- Função transacional - consulta externa: trata de interações para consultas simples, diretas e objetivas sem manipulação/processamento dos dados para retorno.

- Função de dados - arquivos lógicos internos: trata de dados persistidos pela aplicação, associados a ações de "CRUD" (create, read, undo, delete) dados adicionados, alterados e removidos.
- Função de dados - interfaces lógicas externas: trata de dados persistido de forma externa (outras aplicações) a aplicação mas usados pela aplicação.

O processo de contagem é iniciado com a definição do tipo do sistema (sistema novo, manutenção de sistema, evolução de sistema). Essa definição é usado para guiar a ordem de contagem mais adequada e que viabiliza uma maior facilidade. Para desenvolvimento de sistemas novos a contagem é iniciada por funções transacionais, seguidas das funções de dados (ordem indicada: saídas externas, consultas externas, entradas externas, arquivos lógicos internos e interfaces lógicas externas). No caso de sistemas existentes é indicado começar a contagem de forma inversa, inicia-se pelas funções de dados e depois transacionais (ordem indicada arquivo lógico interno, interfaci lógica externa, saídas, consultas e entradas).

A segunda atividade do processo de contagem consiste na definição do escopo/fronteira da aplicação. Nessa etapa, é definido o que fica dentro e fora do escopo da aplicação delimitando subsistemas que devem ser considerados da aplicação.

Posteriormente, é realizada a contagem de pontos de função transacionais e de dados derivando pontos de função não-ajustados como resultado.

Inf

Por último, fatores de ajustes são aplicados para cada grupo de função contabilizado de acordo com valores mapeados em tabelas de ajuste de acordo com complexidades envolvidas. Tais valores podem ter sido ajustado em função de critérios organizacionais e experiências em desenvolvimentos anteriores.

O resultado final trata dos pontos de função ajustados.