DIO - Digital One Innovation

Bootcamp .NET Everis #2

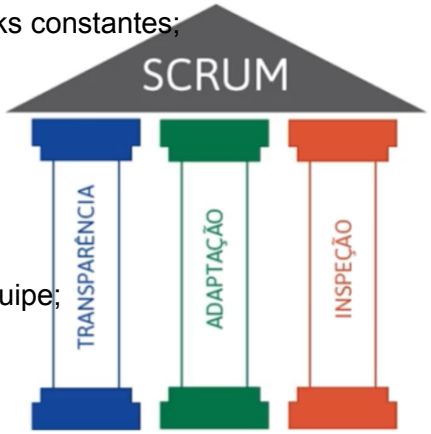
Introdução ao SCRUM

Parte 1 - Conceitos Básicos
Desafios do Desenvolvimento de Software: fazer organização alcançar objetivo.
Objetivos de Negócio -> Processo de Desenvolvimento (Concepção, análise e design, desenvolvimento, testes e implantação) -> Produto de Software.
Gestão de Projetos: Tradicional x ágil
Tradicional: processo em cascata. Cada parte do processo é feito de uma vez e o seguinte só começa quando o anterior termina (demorado, consequentemente, no fim entrega o desejado). Escopo definido no início (preditivo).
Ágil: software dividido em partes de no máximo um mês de prazo, ex: requisito 1 (1 mês), requisito 2 (mais 1 mês), etc… (Requisito 1 > análise > construção > testes > liberação > Requisito 2 > etc...). Escopo definido ao longo do projeto (adaptativo).

Tradicional	Ágil
Escopo definido na fase Inicial do Projeto **(Preditivo).**	Escopo definido ao longo do Projeto **(Adaptativo).**
Projeto é controlado por fases e marcos.	Projeto é controlado por funcionalidades entregues.
Cliente só vê o software funcionando na fase final do Projeto.	Cliente pode ver parte do software funcionando na parte inicial do Projeto.
Resistência a Mudanças.	Mudanças constantes de acordo com feedbacks contínuos.

Ágil = **rapidez nas mudanças** e desembaraço; coisas complexas de forma simples (por partes); equipe comprometida com objetivos; **maior valor para o cliente**. O foco é resolver o problema do cliente, e não o sistema em si.
SCRUM é um dos frameworks de gerenciamento de projetos ágeis (o mais usado). Equipes pequenas.

- Conversar mais e escrever menos;
- demonstrar o software aos usuários e obter feedbacks constantes;
- requisitos mudam ao longo do tempo;
- aprender progressivamente.



Razões para adotar o SCRUM:

- Entregue em partes menores (2 a 4 semanas);
- Melhor gerencimento de riscos;
- Comprometimento, motivação e transparencia da equipe;
- Maior valor para o negócio.

- Maior valor para o negócio;
- Usuários envolvidos durante todo o ciclo e não só no início e fim;
- Aplicações das lições aprendidas para uma melhoria contínua;

Características do time SCRUM:
- Equipes capazes de se auto-organizarem;
- As tarefas do time e todos são responsáveis;
- Forte comprometimento com os resultados.

Papéis e Responsabilidades
Product Owner (PO): Representante de uma área de negócio. Não é um comitê, é uma pessoa. Prioriza as funcionalidades de acordo com o valor de negócio. Garante que o time de desenvolvimento entenda os itens do backlog (funcionalidades do sistema) no nível necessário.
Scrum Master (SM): Garantir o uso correto do scrum. Não é gerente de projetos. Age como facilitador para o PO. Auxilia a equipe a remover impedimentos. Treina o time em autogerenciamento e interdisciplinaridade.
Time de Desenvolvimento (Dev): Possui habilidades suficientes para desenvolver, testar, criar e desenhar, ou seja, tudo que for necessário para entregar o software funcionando.

Cerimônias do SCRUM
Time box: tempo máximo para fazer uma cerimônia ou sprint.
Sprint é o principal evento do SCRUM. É o principal evento do SCRUM. Duração: no máximo 30 dias corridos. O que o ocorre dentro de uma sprint: planejamento da sprint, reuniões diárias 15 min (daily meeting), revisão da sprint (review), retrospectiva da sprint.
Planejamento da sprint (time box de 8h para uma sprint de 30 dias). Os 3 participam. Nas primeiras 4h é definido o que fazer, de
Daily meeting (ou standup metting) 15 min debate-se o que foi feito e etc.
Revisão da sprint: time dev apresenta para o PO o trabalho que foi feito.
Retrospectiva da sprint: 3h. Reunião da equipe para lições aprendidas ao final de cada sprint. Transparência. Garante que todos estejam na mesma página.

Fundamentos de um projeto ágil
Papéis e responsabilidades do PO: Tem a visão do que será desenvolvido, as necessidades a serem atendidas, público e objetivos a serem alcançados. Nem ele, nem o SM são gerentes de projetos. PO é um representante da área de negócios. Ele entende a demanda e extrai o maior valor possível. Ajuda o time a entender a demanda e tira as dúvidas ao longo da sprint.
Refining e Planning.
Release planning: cada vez que um produto de software é criado ou modificado, o fabricante e seus desenvolvedores decidem sobre como distribuir ou o novo produto ou a modificação às pessoas que o utilizam. Existem 2 tipos: de múltiplas squads (vários times) e de projeto (?)

Analisando escopo e definindo prioridades
MVP: minimum valuable product
Product backlog: Épicos (incremento sem muito detalhamento que ajuda a direcionar os caminhos que se deve seguir) e Estória (detalhamento dos épicos. Um épico normalmente

se divide em várias estórias onde ficam descritos o que deve acontecer e suas regras de negócio).
Escrevendo uma estória: Nome, descrição, regras de negócio, tela, kpi (objetivos/valor que a estória precisa atingir), tagueamento (como a estória será tagueada para poder mensurar os KPI), critérios de aceite (qual o passo a passo de todos os caminhos felizes possíveis a estória deve cumprir para que ela seja considerada aceita).

Riscos
Positivos: muito ignorados, mas um dos fatores de maiores ganhos no desenvolvimento de sistemas. Pode-se priorizar outros itens no backlog.
Negativos: itens que podem afetar o prazo, custo ou escopo de um projeto de maneira que pode acabar inviabilizando-o.

Papel do PO na transformação digital
Transformação digital: processo no qual as empresas fazem uso da tecnologia para melhorar o desempenho, aumentar o alcance e garantir resultados melhores. É uma mudança estrutural nas organizações dando um papel essencial para a tecnologia.

Conceitos e atividades essenciais para o sucesso de um projeto ágil
Aprenda sobre os conceitos e planejamento de tarefas.
Épico > Estória > Tarefa (estória é uma tarefa descrita em nível de negócio; tarefa é um conjunto de atividades que o time de dev deve desempenhar para entregar uma estória.)
Exemplo de uma estória:
Nome da Estória; Descrição: Eu, como, quero, quando; Regras de negócio: separar regras de front-end de regras de back-end; Tela: link ou imagem das telas a serem desenvolvidas; KPI: quais os objetivos/valor a estória precisa atingir; Tagueamento: como a estória será tagueada para poder mensurar os KPI; Critérios de aceite: qual o passo a passo de todos os caminhos felizes possíveis a estória deve cumprir para que ela seja considerada aceita.

O que é um critério de aceite?
É objetivo que a estória tem que concluir. É uma lista de critérios que precisa ser alcançados para que a estória atenda os requisitos do cliente e seja aceita pelo PO. Os critérios de aceitação tem o objetivo de: definir limites para as user stories. Ajudar o PO a detalhar em alto nível o que é necessário para entregar valor ao cliente.

Estimativa e planejamento
Algumas atividades para mensurar esforço e complexidade: planning poker; planning tshirt size

Rotinas de um time ágil
Daily e Retrospectiva (reuniões para o time ficar ciente dos acontecimentos da sprint).
Na Daily, entre PO, SM e Devs, os únicos que devem OBRIGATORIAMENTE participar são dos Devs. Na retrospectiva, os DEVS e SM (PO não necessariamente).
Refinamento (refinement): acontece antes da planning (no final da sprint anterior) participação de TODOS.
Review: todos os interessados participam inclusive stakeholders. Analisam se foi tudo atendido. Verifica se a demanda agrega valor ao negócio.

Maturidade do time: Transparência, comprometimento, coragem; Inspeção, abertura, foco, respeito; Adaptação. Quando o próprio time “puxa” o PO, sem nem necessitar mais de SM.

Estrutura de Dados, Arrays e Registro

Introdução

O que é Estrutura de Dados

É uma estrutura organizada de dados na memória de um computador ou em qualquer dispositivo de armazenamento, de forma que os dados possam ser utilizados de forma correta. Essas estruturas encontram muitas aplicações no desenvolvimento de sistemas, sendo que algumas são altamente especializadas e tarefas específicas (ex: estrutura de busca recursiva, outras). Usando as estruturas adequadas através de algoritmos, podemos trabalhar com uma grande quantidade de dados, como aplicações em bancos de dados ou serviços de busca, data mining, etc…
Situação para que uma estrutura seja efetiva, cria-se através do Algoritmo, que é um conjunto de instruções estruturadas e ordenadas, seu objetivo é realizar uma tarefa ou operação específica. Os algoritmos são utilizados para manipular dados nas estruturas de várias formas, como por exemplo: inserir, excluir, procurar e ordenar dados.
Imaginando que um guarda-roupa é um banco de dados, se estiver organizado será mais fácil encontrar as coisas, isto é, ele estará bem estruturado.
Em uma estrutura de dados devemos saber como realizar um determinado conjunto de operações básicas como por exemplo: Inserir dados, excluir, localizar elemento, percorrer todos os itens constituintes da estrutura para visualização, classificar, que se resume em colocar os itens de dados em uma determinada ordem (numérica, alfabética, etc).
Principais estruturas de dados: Vetores e matrizes (arrays), registro, lista, pilha, fila, árvore, tabela hash e grafos.

Vetores e Matrizes

Também chamados de Arrays. São estruturas de dados simples que pode, auxiliar quando há muitas variáveis do mesmo tipo em um algoritmo. Vetor ou array-unidimensional é uma variável que armazena várias variáveis do mesmo tipo. O vetor é uma estrutura de dados indexada, que pode armazenar uma determinada quantidade de valores do mesmo tipo.
Matriz ou array multi-dimencional é um vetor de vetores, ou seja, que possui duas ou mais dimensões. Uma matriz 4x3 (4 colunas e 3 linhas, 12 valores).

Registro

Enquanto Arrays nos permitem armazenar vários dados de um único tipo, o recurso de Registro nos permite armazenar mais de um tipo de dado. Um exemplo de uso seria um registro de cadastro de cliente, com campos CPF, Nome, Endereço, Telefone, etc.
Toda estrutura de registro tem um nome (ex: cliente), e seus campos podem ser acessados por meio do uso do operador ponto (.). Por exemplo, para acessar o preço de um livro, poderíamos utilizar a seguinte declaração: “livro.preco”

Entenda o que são Listas, Pilhas e Filas

Introdução

Listas

Estrutura de dados do tipo lista armazenam dados de um determinado tipo em uma ordem específica. A diferença entre listas e arrays é que listas possuem tamanho ajustável, enquanto arrays possuem tamanho fixo.
Listas ligadas: existem os nós, e cada um dos nós conhece o valor que está sendo armazenado em seu interior além de conhecer o elemento posterior a ele: por isso ela é chamada de lista ligada, pois os nós são amarrados com essa indicação de qual é o próximo nó. Lista pode ir crescendo conforme novos itens se inserem.
Lista duplamente ligada: são bidirecionais. Permitem uma navegação reversa, já que os elementos sabem quem é o elemento anterior.

Pilhas (Stack)

É uma estrutura de dados que serve como uma coleção de elementos e permite o acesso a somente um item de dados armazenado, tipo uma pilha de peças que se encaixam. Tipos de pilha:
LIFO (Last in first out) ou UEPS: a estrutura do tipo pilha LIFO o primeiro elemento a ser retirado é o último que tiver sido inserido.
FIFO (first) ou PEPS: o primeiro a ser retirado será o primeiro que foi inserido (como se retirasse a peça da base da pilha/stack).

Filas

A estrutura do tipo fila admite remoção de elementos e inserção de novos sujeita a seguinte regra de operação: o elemento removido é o que está na estrutura há mais tempo, ou seja, o primeiro objeto inserido na fila é também o primeiro a ser removido seguindo o conceito FIFO.

Estruturas de dados do tipo Árvore Tabela Hash e Grafos

Introdução

Árvores

É uma estrutura de dados que organiza seus elementos de forma hierárquica, onde existe um elemento que fica no topo da árvore, chamado de raiz e existem os elementos subordinados a ele, que são chamados de nós ou folhas.

Tabelas hash

Também conhecida por tabela de dispersão ou espalhamento. É uma estrutura de dados especial, que associa chaves de pesquisa a valores. Por que espalhar? A tabela hash permite a associação de valores a chaves. Valores: é a posição ou índice onde o elemento