Resumo Prova 1 - APOO - Análise e Projeto Orientada a Objeto

Profª: Marcia Regina M Cassitas Hino

GRR	NOME
2017208552	Bruno Leandro Diniz

Introducao e Fundamentos OO

- O que é um sistema?
    - Sistema é um sistema é um conjunto de partes coordenadas para atingir um objetivo.
        - Sistema respiratório (respiração)
- Paradigmas
    - Um paradigma é um padrão, a forma como você soluciona um problemas.
    - Paradigma de programação é um meio de se classificar as linguagens de programação baseado em suas funcionalidades.
    - Entender os paradigmas auxilia o desenvolvedor a entender melhor e estruturar o sistema.
    - Exemplos:
        - Programação estruturada
        - Programação orientada a objeto
- Orientação a Objetos
    - Significa organizar o mundo real como uma coleção de objetos que incorporam estrutura de dados (atributos) e um conjunto de ações (métodos) que manipulam estes dados.
    - Conceitos:
        - Objetos: É uma entidade lógica que contém atributos e métodos
            - Atributos: são as características próprias dos objetos
                - Exemplo: nome da pessoa, cor do carro, valor da transação
            - Métodos: são blocos de instruções que manipulam os atributos
                - Exemplo: falar (pessoa), acelerar (carro), debitar (transação)
        - Classes: É uma estrutura de dados para declarar variáveis. Essa estrutura serve para definir objetos
            - Uma instância de uma classe é chamada de Objeto.
            - Definir uma classe não cria um objeto, assim como um tipo de variável não é uma variável.
    - Características:
        - Encapsulamento, Abstração, Herança e Polimorfismo

> Classes não são Objetos, mas a estrutura de um Objeto

Classe Cachorro { Atributos { raça: cor: altura: } Métodos { fazerCocô() crescer() latir() } return Objeto (conjunto de Atributo e Método) }

- EXERCÍCIO 1: Identifique as classes, atributos e métodos do cenário abaixo: João controla todo o gasto mensal da sua conta de luz em uma planilha eletrônica. Para cada conta de luz ele registra: data em que a leitura do relógio de luz foi realizada, número da leitura, quantidade de Kw gasto no mês, valor a pagar pela conta e data do pagamento. Mensalmente se pesquisam o mês de menor consumo e o de maior consumo

---

- Encapsulamento
    - Um dado está encapsulado quando envolvido por código de forma que, só é visível na rotina onde foi criado.
    - Não interessa saber como é o funcionamento interno da classe e sim sua função. É como uma “caixa preta”.
    - Se refere ao agrupamento de dados com os métodos que operam nesses dados ou à restrição do acesso direto a alguns dos componentes de um objeto.
    - Exemplos:
        - Quando você aperta o botão “ligar” do aparelho remoto da televisão, você não sabe o que acontece, como o sinal chega na televisão, que circuitos são ativados, mas a televisão “liga”.
        - Quando você aperta o botão “play” do rádio, você sabe o que acontece? Como o som é emitido?
    - As opções de visibilidade que podem ser definidas em uma classe para atributos e serviços são:
        - (+) public: os elementos são acessíveis por todas as classes
        - (#) protected:os elementos são acessíveis por subclasses, ou pela própria classe
        - (-) private: os elementos são acessíveis somente pela própria classe
- Abstração
    - A abstração é o processo de filtragem de detalhes sem importância do objeto, para que apenas as características apropriadas que o descrevem permaneçam
    - Deve ser executada com algum objetivo para saber o que é importante e o que não é
    - Extrair tudo o que for essencial e nada mais
    - FLORESTA
        - Lenhador: tamanhoArea, quantidadeArvores, valorMercado / estimarValor(), venderLenha(), calcularLucro()

        - Passarinho: alimento, agua, ninho / comer(), beber(), dormir()

- Herança
    - Significa ser capaz incorporar os atributos e métodos de uma classe previamente definida.
    - Tem por objetivo criar uma hierarquia de objetos do mundo real, estabelecendo um relacionamento de pai e filho
    - Por meio da herança pode-se reutilizar ou alterar os métodos de classes existentes, bem como adicionar novos atributos a fim de adaptá-los a novas situações. Isso se chama especialização das classes
        - Herança simples: herda de um objeto
        - Herança múltipla: herda de vários objetos

- EXERCÍCIO 2: Organize hierarquicamente em um diagrama as seguintes classes:
    - 1.VeiculoTerrestre, 2.VeiculoAquático, 3.Carro, 4.Moto, 5.Barco, 6.NavioCargueiro, 7.Caminhão, 8.VeiculoAnfibio, 9.Submarino, 10.VeiculoAereo, 11.Aviao, 12.Veiculo, 13.HidroAviao

- EXERCÍCIO 3: Organize hierarquicamente em um diagrama as seguintes classes:
    1. MembroDaEscola, Professor, Coordenador, Funcionário, Ex aluno, CorpoDiscente (*), AlunoDaGraduacao, AlunoDaPosGraduacao, Atendente, CorpoDocente, Segurança
    - Discente é todo aluno regularmente matriculado, em regime de dependência ou trancado.

- Polimorfismo
    - É o princípio pelo qual classes derivadas de uma mesma superclasse podem invocar o “mesmo” método, porém com comportamentos distintos para cada uma das classes derivadas.
    - Denota uma situação na qual um objeto pode se comportar de maneiras diferentes ao receber uma mensagem.
    - Consiste em utilizar o mesmo nome do método dentro de uma classe, ou em uma hierarquia de classes, com comportamentos diferentes.
    - É alcançado com auxílio do uso de herança nas classes e a reescrita de métodos das superclasses nas suas subclasses.

Fundamentos de Modelagem OO

- Descrição diagramática de algo a ser implementado em linguagem de programação
    - Modelagem prescritiva: antes do código
    - Modelagem descritiva: após o código

- Ato de criar uma representação de algo do mundo real por meio de modelos, simplificação da realidade:
    - Planta de uma casa representa mas não é a casa
    - Modelagem de sistemas: representações do sistema que estamos desenvolvendo

- OBJETIVOS:
    - Especificar estrutura e/ ou o comportamento do sistema.
        - Softwares complexos demandam planejamento. Os de baixa complexidade podem ser construídos direto.
        - Proporcionar guia para construção.
        - Documentar tomadas de decisões.

- VANTAGENS:
    - Descrição mais facilmente compreensível
        - Mais próxima da forma como as pessoas pensam
        - Não é natural “pensar” em linguagem de programação
    - Proporciona diferentes pontos de vista
        - Descrição dos elementos que compõem um programa (estrutura)
        - Descrição do programa em execução (dinâmica)
        - Possibilidade de visão global
        - Possibilidade de atenção a detalhes.

O QUE TEMOS QUE APRENDER PARA MODELAR EM OO?

- Conhecer os conceitos referentes a modelagem
    - Saber porque modelar
    - Conhecer os paradigmas de OO
    - Conhecer os requisitos de uma modelagem completa
    - Conhecer uma linguagem de modelagem
    - Ex: UML e todos seus diagramas

- Saber que passos seguir
    - Saber usar a linguagem de modelagem adotada

- Avaliar o que for produzido
    - Avaliar consistência do todo
    - Parâmetros de qualidade

Levantamento Requisitos

- Primeiros passos para iniciar um sistema?

- Levantamento de requisitos
- É importante:
  - Identificar o problema a ser resolvido.
  - Entender como o problema é tratado atualmente.
  - Investigar soluções existentes no mercado (concorrentes).
  - Perguntar aos usuários sobre a experiência atual.
  - Identificar características de usabilidade.PÓ

    > Identifique necessidades dos usuários e estabeleça requisitos.

- Requisitos
  - É um mapeamento do que se espera que o sistema apresente.
  - É uma condição ou capacidade para a qual um sistema deve atender.
  - É como se fosse uma espécie de declaração de o que o sistema deve ter ou de como deveria operar.
  - Não é fácil:
    - Entender as funcionalidades;
    - Listar necessidades futuras;
    - Entender problemas que você não está familiarizado;
    - Entender detalhes de processos;
    - Ter a visão do todo.
  - Exemplos de requisitos:
    - Funcionalidades ao nível do usuário
      - Emitir certificado de participação do aluno no evento da semana pedagógica.
    - Propriedades gerais do sistema
      - O sistema deverá manter um registro de toda consulta e alteração no cadastro.
    - Uma regra de negócio
      - A nota do aluno será a média aritmética simples entre as notas parciais.
    - Restrições do sistema ou do seu desenvolvimento
      - O sistema deve ser desenvolvido com ferramentas Microsoft®.
      - O sistema deve atender as normas de acessibilidade existentes. Atenção!
    - O levantamento pode indicar visões diferentes e necessidades contraditórias;
    - Usuários geralmente conhecem muito do negócio e pouco de sistema, enquanto que os analistas conhecem muito de sistemas e pouco do negócio;
    - Usuários podem omitir informações, acreditando que é algo óbvio.
  - Requisitos são importantes para projetos de TI, pois são a base para:
    - Definição de contratos;
    - Elaboração/análise de propostas;
    - Planejamento de atividades;
    - Estimativas de custo;
    - Modelagem do sistema;
    - Desenvolvimento das funcionalidades.
  - Sendo assim....
    - O mapeamento e entendimento completo dos requisitos é fundamental para se obter um sistema e um processo de desenvolvimento de qualidade.
    - O sistema deve ser desenvolvido de maneira a evoluir para atender necessidades futuras eminentes dos usuários, mesmo que não seja um requisito claro por parte deles.
    - O objetivo do requisito é que seja completo o suficiente para que o desenvolvimento possa ser realizado e que se possa validar com o usuário se o que está sendo entregue é o que foi solicitado, e se atende sua necessidade.

1. Requisitos Funcionais (RF)
   - Funcionalidade (funções que o sistema deve realizar)
   - Serviços (que se espera que o sistema faça)
2. Requisitos Não Funcionais(RNF)
   - Qualidade
   - Tipos:
     - Requisitos do Produto:
       - Especificam o comportamento do software (ex.: eficiência, desempenho, segurança, portabilidade, usabilidade, confiabilidade).
       - “A base de dados deve ter acesso apenas de usuários autorizados”.
     - Requisitos Organizacionais:
       - Consequência de políticas e procedimentos das empresas (ex.: padrões do cliente)
       - O processo de desenvolvimento deve estar de acordo com normas ISO 9126”.
       - O prazo de entrega final do sistema é de 3 meses”.
     - Requisitos Externos:
       - Derivados do ambiente ou fatores externos ao sistema (ex.: legislação)
       - O sistema deverá se comunicar com o SQL Server” (interoperabilidade).
       - O sistema deverá atender às normas legais, tais como padrões, leis, etc” (legal).
       - O sistema não apresentará aos usuários quaisquer dados de cunho privativo” (ético).

- Requisitos Técnicos
  - Recursos necessários para que a aplicação possa operar, tais como processamento, memória, armazenamento, link de comunicação.
  - COMPATIBILIDADE: aplicação deve rodar independente da arquitetura e do dispositivo que a esteja hospedando.
  - DISPONIBILIDADE: Que tipo de informação é necessário acesso em tempo real?
    - Que tipo de informação pode haver um atraso? Qual o tamanho deste atraso?
  - ARMAZENAMENTO: •Por quanto tempo registros antigos devem ser mantidos?
- Requisitos Ambientais
  - AMBIENTE SOCIAL: Fisicamente próximas? Existe necessidade de colaboração de dados?
    - Síncrono ou assíncrono? Como será a comunicação?

        - AMBIENTE ORGANIZACIONAL: Hierarquia, Estrutura técnica (funcionários e equipamentos)
            - Implantação de projetos, Investimentos em tecnologia?
        - AMBIENTE TÉCNICO: Quais são as tecnologias utilizadas? O que precisa manter a compatibilidade?
            - Limitações tecnológicas?
    - Atividade: Identifique os requisitos funcionais e não funcionais:
        - “Um aplicativo de celular de emissão de passagens vende passagens de trem e avião. Quando o usuário entra no aplicativo, um menu com os possíveis destinos é mostrado ao usuário, junto com uma mensagem para que o usuário selecione um destino. Os usuários podem escolher seu destino a partir dessa lista. Os destinos sugeridos devem ser organizados de maneira a facilitar a escolha pelo usuário. Após a escolha do destino, o sistema deve responder prontamente se há poltrona disponível. Somente após essa verificação é solicitado ao usuário que informe o número de seu cartão de crédito e o código de segurança. O cartão é validado. Quando a transação de crédito é validada, a passagem é emitida. O formato do bilhete de passagem deve seguir o padrão definido pelo Sistema Nacional de Tráfego Ferroviário ou pelo Sistema Nacional de Tráfego Aéreo, a depender do transporte escolhido”.

- Técnicas de levantamento de requisitos
    - Objetivo:
        - Ajudar os analistas de sistemas e negócio na condução do levantamento de requisitos do sistema.
    - Atividades envolvidas:
        - Identificar os objetivos
        - Selecionar o público alvo adequado
    - Qual técnica usar? Analisar:
        - Recursos disponíveis($$$)
        - Tempo disponível(para execução e compilação de resultados)
        - Disponibilidade de usuários(presencial, remota etc)
    - Técnicas de levantamentov:
        - Entrevistas
            - Obter informações que não são registradas e estão concentradas em algumas pessoas.
            - Obter a opinião das pessoas sobre o sistema.
            - Recomendações
                - Não atrase.
                - Explique o objetivo da pesquisa para conquistar a confiança do entrevistado.
                - Escute mais que pergunte.
                - Vá preparado. Evite ler as perguntas do roteiro.
                - Tomar notas durante a entrevista.
                - Fazer o relatório da visita e validar as informações.
        - Questionário
            - É um formulário que pode ser distribuído impresso ou on-line com perguntas que os usuários e demais participantes.
            - Otimizar o entendimento do processo se um grande número de pessoas deve ser entrevistado e não há recursos para entrevistá-las individualmente.
            - Quando há diversos grupos de usuários que podem estar em localizações geográficas diferentes.

                > Inclua sempre uma alternativa “sem opinião”. Não sei responder.

        - Observação
            - O observador é inserido no ambiente de trabalho onde a solução será usada observando o trabalho cotidiano e tomando notas das tarefas em execução nas quais as partes interessadas estão envolvidas.
            - Quando usar?
                - Para avaliar o processo de trabalho existente.
                - Para melhorar ou modificar o sistema atua.
                - Descobrir problemas de performance.
                - Confirmar dados recolhidos na entrevista.
            - Recomendações:
                - Deve ocorrer preferencialmente em um período em que o serviço seja caracterizado como normal, para não haver distorções;
                - O observador deve colocar-se no ambiente sem interferir,
                - O observador não deve fiscalizar o trabalho;
        - Reunião
            - Comunicação direta entre as partes interessadas
            - Integração do grupo em torno do mesmo assunto, formando uma equipe de trabalho
            - Coleta de sugestões e críticas da equipe
            - Quando usar?
                - obter uma resposta rápida de várias pessoas sobre um determinado assunto;
                - para resolução de problemas ou questões a serem esclarecidas, compartilhadas com os demais participantes;
                - para envolver o grupo da tomada de decisão;
                - para resolver situações de conflito (busca de consenso);
            - Recomendação:
                - Não defender suas ideias ao extremo; não mostrar contrariedade; não mudar de opinião para evitar conflitos
        - Brainstorming
            - “tempestade de ideias”
            - Gerar ideias e selecionar
            - Explicar as regras:
                - Não critique!
                - Não avalie!
                - Seja aberto a todas as ideias, mesmo aquelas que parecerem malucas.
                - Escrever as ideias dos participantes no quadro.
        - Cenários
        - Protótipos

- QUAIS OS PROBLEMAS NO LEVANTAMENTO DE REQUISITOS?
    - NÃO VER O TODO: REQUISITOS AUSENTES OU SUBENTENDIDOS.
    - FALTA DE COMPLETUDE OU INCONSISTÊNCIA.
    - FALTA DE CLAREZA, PERMITINDO INTERPRETAÇÕES DÚBIAS.
    - FALTA DE DEFINIÇÃO CLARA DE RESPONSABILIDADES.
    - FALTA DE ENVOLVIMENTO DO USUÁRIO.
    - DIFICULDADE PARA ENTENDER O PROBLEMA.
    - MUDANÇA DE OBJETIVOS.
    - FUNCIONALIDADES ADICIONAIS AOS REQUISITOS. (XÍCARA DE OURO)
- Em relação às fontes de informação
    - Stakeholders
        - Disponibilidade / Restrições de contato
    - Sistema atual
        - Disponibilidade para consulta / Documentação / Manual.
    - Perspectivas de futuro
    - Leis / Regulamentações
    - Normas
        - O que deve ser seguido
    - Regras da organização

Projeto OO e UML (LINGUAGEM DE MODELAGEM UNIFICADA!!)

- Linguagem visual utilizada para modelar sistemas baseados no paradigma de orientação a objetos.
- Incorpora noções do desenvolvimento de software totalmente visual e se baseia em diagramas que são modelados e classificados em visões de abstração.
- Características:
    - Os projetistas pensam em termos de coisas, em vez de funções.
    - A funcionalidade do sistema é expressa em termos de serviços oferecidos pelos objetos.
        - Objetos são abstrações do mundo real ou entidades do sistema que se auto gerenciam.
        - Objetos se comunicam por passagem de mensagem.
- Na fase de análise de um projeto OO, busca-se identificar objetos e descrevê-los.

  Classe Livro { Atributos { titulo: ano: categoria: } Métodos { emprestar() } return Objeto (conjunto de Atributo e Método) }
- Vantagens de um projeto OO:
    - Facilidade de manutenção, onde os objetos podem ser entendidos como entidades independentes;
    - Os objetos são componentes potencialmente reutilizáveis;
    - Para vários sistemas existe um mapeamento nítido das entidades do mundo real para objetos no sistema
- Visões UML:
    - Mostram diferentes aspectos do sistema que está sendo modelado.
        - VISÃO DE CASO DE USO
        - VISÃO LÓGICA
        - VISÃO DE IMPLEMENTAÇÃO
        - VISÃO DE PROCESSOS
        - VISÃO DE IMPLANTAÇÃO
- Diagramas:
    - Casos de uso: Expressam a funcionalidade de um sistema
    - Atividades: Representam o fluxo de atividades dos processos de negócio
- O desenvolvimento de um sistema em UML divide-se em cinco fases:
    - Análise de requisitos
    - Análise
    - Design
    - Implementação (programação)
    - Testes
- Rational Unified Process (RUP)
    - É uma metodologia para desenvolvimento de processos que utiliza abordagem de orientação a objetos em sua concepção.
    - Usa como base a notação UML para ilustrar os processos.
    - FASES RUP
        - Iniciação: Quando se especifica da visão do sistema e se define o escopo do sistema.
        - Elaboração: Quando se faz o planejamento das atividades necessárias e dos recursos requeridos, bem como a especificação do sistema e o design da sua arquitetura. Foco na arquitetura.
        - Construção: Desenvolvimento do produto como uma série de interações incrementais. Desenvolver o sistema.
        - Transição: Fornecimento do produto para o usuário (fabricação, distribuição e treinamento). Foco na implantação.

Diagrama de Caso de Uso

- CASO DE USO
    - Um caso de uso especifica o comportamento de um sistema ou parte de um sistema e é uma a descrição de uma sequencia de ações.
    - Representação das funcionalidades do sistema visíveis externamente e dos elementos externos.
    - Representa quem faz o que (interação) com o sistema sem considerar comportamento interno do sistema, ou seja, o que o sistema faz, e não como ele faz.

- OBJETIVOS
    - Identificar entidades relevantes, como se relacionam e como se comportam
    - Ser passível de compreensão tanto por desenvolvedores como por usuários
    - Descrever o sistema sob uma perspectiva externa (o que ele faz, não como faz)
    - Ser completo, consistente e não ambíguo
- DIAGRAMA
- O diagrama de caso de uso contém vários casos de uso. A quantidade exata de casos de uso obviamente depende da complexidade do sistema em desenvolvimento.
- Quanto mais complexo o sistema, maior a quantidade de casos de uso.
- DIMENSÕES
    - Formato
        - Diz respeito à estrutura utilizada para organizar a sua narrativa textual. Os formatos comumente utilizados são o contínuo, o numerado e o tabular
            - "O Cliente chega ao caixa eletrônico e insere seu cartão, em seguida o sistema requisita a senha do Cliente. Após o Cliente fornecer sua senha"
            - "Cliente insere seu cartão no caixa eletrônico. 2.Sistema apresenta solicitação de senha. 3.Cliente digita senha."
    - Grau de detalhamento
        - Varia de sucinto até detalhado (expandido).
    - Grau de abstração
        - Diz respeito a existência ou não de menção a aspectos relativos à tecnologia
        - Ex: Para o caso de uso “realizar pedido”, pode ser por telefone, por e-mail, etc.
- ATORES
    - Um caso de uso envolve a interação dos atores com o sistema.
    - Um ator representa um conjunto de papéis que os usuários dos casos de uso desempenham na interação com esse caso.
    - Corresponde a um papel representado no sistema (Ex: funcionário, cliente, fornecedor).
    - Pode ser pessoas, organizações, outros sistemas ou equipamentos
- INDAGAÇÕES
    - Quem utiliza o sistema?
    - Quem instala e mantém o sistema?
    - Que outros sistemas/dispositivos utilizam o sistema ou são utilizados por ele?
    - Quem obtém informação do sistema?
    - Quem provê informação ao sistema?
    - O que o sistema faz automaticamente?

> Ator (boneco)--------{reservar livro}

- TIPOS DE CASOS DE USO
    - Caso de uso primário
        - São os que representam os objetivos dos atores
    - Caso de uso oposto
        - É o que “desfaz” outro (Ex: cancelar pedido)
    - Caso de uso temporal
        - Caso de uso temporal são os disparados pelo sistema e não por um ator (Ex: o sistema realiza de forma automatica).
    - Caso de uso secundário
        - Os secundários são os que não trazem benefício direto aos atores, mas são necessários para que o sistema funcione corretamente (Ex: manutenção de cadastro; de usuários).
- TÉCNICAS BÁSICAS DE MODELAGEM
    - Identificar os atores que interagem com o sistema.
    - Organize os atores, identificando papéis gerais e mais específicos.
    - Para cada ator considere as formas primárias que o ator interage com o sistema.
    - Considere também as formas excepcionais em que cada ator interage com o sistema.
    - Organize esses comportamentos como um caso de uso.
    - Descreva o fluxo de eventos de maneira que permita compreendê-lo com facilidade.
- EXERCÍCIO:
    - Vários projetos são realizados em uma empresa. Os cem empregados da empresa trabalham em pelos menos um projeto. Há um sistema implantado na empresa que permite aos participantes que estão alocados em um determinado projeto marcarem suas horas de trabalho. Esse sistema também permite que os gerentes de cada projeto, ao fim do mês, gerem relatórios com os totais de horas trabalhadas de cada participante
    - Quantos são os atores?
    - Quais seus papéis?
    - Quais os casos de uso?

```
- RELACIONAMENTOS
    - O ator comunica-se com o sistema através do envio e recebimento de mensagens.
    - Os relacionamentos podem ser de 4 tipos:
        * Associação
            - Indica que há uma interação (comunicação) ou relacionamento entre um caso de uso e um ator.
        * Generalização/especialização
            - Entre atores: Atores podem herdar, ou seja, um ator pode desempenhar os mesmos papéis que o outro, e também papéis adi
            - Entre casos de uso: Quando existem dois ou mais casos de uso com características semelhantes, mas com pequenas diferen
        * Inclusão <<include>>
            - Utilizado quando existe uma situação ou rotina comum a mais de um caso de uso. (evitar repetição de caso de uso).
            - Este tipo de relacionamento indica uma obrigatoriedade: Quando um determinado caso de uso tem um relacionamento de inc
        * Extensão <<extensão>>
            - São utilizadas para descrever cenários opcionais de um caso de uso.
            - Ocorrem em situação específica onde determinada condição deve ser satisfeita. Assim, diferente da relação de inclusão,

- Extensão x Inclusão
    * Extensão <<extends>> indica que, em certas situações, ou em determinado momento, um caso de uso poderá ser “expandido” pela ex
    * Inclusão <<include>> indica que um caso de uso incorpora a funcionalidade de outro caso de uso.

- CONCRETO ou ABSTRATO
    * Concreto: iniciado diretamente por um Ator
    * Abstrato: não iniciado diretamente por um ator, geralmente relacionado a outro caso de uso.
```

- EXERCÍCIO 1: SISTEMA DE SAQUE DE DINHEIRO Identificar atores, casos de uso e tipos de relacionamentos entre eles. Para efetuar o saque do dinheiro, solicitado por um cliente, em uma ATM deverá ser necessário a validação do cartão pessoal do cliente, a validação da senha pessoal do cliente e se o saldo disponível na conta do cliente é suficiente para o saque solicitado.
- EXERCÍCIO 2: SISTEMA EDITOR DE TEXTO Identificar atores, casos de uso e tipos de relacionamentos entre eles. Em um aplicativo de editor de texto ao se inserir um texto é possível verificar a ortografia do texto que está sendo inserido, alterar o template em questão e encontrar sinônimos para as palavras contidas no texto.

- Um caso de uso deve descrever as funcionalidades de um sistema.
- Em geral, cada passo do fluxo de um caso de uso descreve uma das seguintes situações: i.uma interação entre um ator e o sistema ii.uma ação que o sistema realiza para atingir o objetivo do ator iii.uma ação que o sistema realiza para proteger os interesses de um interessado.
- Um bom modelo contempla:
    - Nome
        - Nome do caso de uso
    - Descrição
        - Descrição sucinta, em um único parágrafo, descrevendo o objetivo do caso de uso
    - Atores
        - Nome dos atores primários (aqueles que iniciam a execução) e dos atores secundários (aqueles que participam ou executam em uma determinada circunstância)
    - Pré-condições
        - Condições a serem atendidas antes da execução. Caso não sejam, o caso de uso não pode ser executado
    - Pós-condições
        - O que deve ser verdadeiro após a execução, considerando que o fluxo é realizado com sucesso
    - Fluxo básico
        - Descreve a sequência de passos a serem realizados em situação normal (quando tudo dá certo)
    - Fluxo alternativo
        - Descreve a sequência de passos alternativa (geralmente devido a uma escolha do usuário)
    - Fluxo de exceção
        - Se refere ao tratamento de erros durante a execução
        - Observa-se que a maioria das exceções ocorre nos passos em que alguma informação é incluída no sistema pelos atores. Isso porque, quando isso ocorre, muitas vezes, realizam-se validações. Quando uma dessas validações falha, ocorre uma exceção
    - Regras de negócio
        - Lista de requisitos atendidos (funcionais ou não funcionais)
    - Classes / entidades
        - Relação de classes necessárias no caso de uso

Títulos	Descrição dos títulos
Nome	Efetuar Saque
Descrição	Este caso de uso permite que um cliente do banco efetue um saque, retirando dinheiro de sua conta bancária.
Atores	Cliente
Pré-	O caixa automático deve estar conectado ao sistema bancário.

Títulos	Descrição dos títulos
condição	
Pós-condição	O saque é efetuado gerando débito do valor na conta do cliente e entregando o mesmo valor, em espécie, ao cliente.
Regras de negócio	RF01, RN01, RNF01, RNF02
Classes	Cliente, Conta, Cartão, Transação, Saque
Fluxo básico	O cliente insere seu cartão no caixa automático, que analisa o cartão e verifica se ele é aceitável.
	Se o cartão é aceitável, o caixa automático solicita que o cliente informe a senha.
	O cliente informa a senha.
	O caixa automático envia os dados do cartão e da senha para o sistema bancário para validação.
	Se a senha estiver correta, o caixa solicita que o cliente informe o tipo de transação a ser efetuada.
	O cliente seleciona a opção saque e o caixa solicita que seja informada a quantia.
	O cliente informa a quantia a ser sacada.
	O caixa envia uma requisição para o sistema bancário para que seja efetuado um saque na quantia especificada pelo cliente.
	Se o saque é autorizado, as notas são separadas e liberadas.
Fluxo alternativo	Cancelamento: O cliente pode cancelar a transação a qualquer momento, desde que o saque ainda não tenha sido autorizado pelo sistema bancário.
Fluxo de exceção	O cartão não é aceitável: Se o cartão não é aceitável (porque a tarja magnética não pode ser lida ou porque é de um tipo de cartão desconhecido), uma mensagem de erro de leitura é mostrada e o cartão é liberado.
	Senha incorreta: Se a senha informada está incorreta, uma mensagem deve ser mostrada para o cliente que poderá entrar com a senha novamente. Caso o cliente informe três vezes senha incorreta, o cartão deverá ser bloqueado.
	Saque não autorizado: Se o saque não for aceito pelo sistema bancário, uma mensagem deve ser exibida e a operação deve ser cancelada.
	Não há dinheiro suficiente disponível no caixa eletrônico: Nesse caso, uma mensagem de erro é exibida e a operação deve ser cancelada.

- RF01 : O sistema de caixa automático deve permitir que clientes efetuem saques em dinheiro.
- RN01 : Não devem ser permitidas transações que deixem a conta do cliente com saldo negativo.
- RNF01 : O sistema de caixa automático deve estar integrado ao sistema bancário
- RNF02 : As operações realizadas no caixa automático devem dar respostas em até 10s a partir da entrada de dados.
- EXERCÍCIO 1: DESCRIÇÃO DE CASO DE USO
    - Com base no diagrama de caso de uso do jogo de forca, descrever o caso de uso “Manter Tema”, utilizando a estrutura recomendada na disciplina.
    - adm ----- {manter tema}
    - adm ----- {manter banco de palavras}
    - jogador ----- {iniciar jogo}
    - jogador ----- {realizar jogada}
    - jogador ----- {incluir nome no painel de pontuação}

Diagrama de Atividades

- Representam o fluxo de atividades dos processos de negócio
- Mostra o fluxo de uma atividade para outra
- Maneira alternativa de representar interações, podendo expressar como as ações são executadas, o que elas fazem, quando elas são executadas, e onde elas acontecem
    - Objetivos:
        - Capturar as ações que serão executadas quando uma operação é disparada;

        - Mostrar como um negócio funciona em termos de atores, fluxos de trabalho, organização e objetos.

- Componentes:

    - Estado Inicial (bolinha preenchida)
    - Estado Final (bolinha dentro da outra)
    - Ação ou atividade (retângulo)
    - Decisão (prisma)
    - Gatilho ou arco (---->)
    - Bolinha com X no meio (atividade termina naquela situação e vai seguir de outro ponto)

- Raias:

    - Permite que se documente o que acontece e quem faz acontecer;
    - Permite que, em modelagem de domínio, o diagrama de atividade represente pessoas ou departamentos responsáveis por cada atividade.
    - Para usar raias, você deve organizar seus diagramas de atividades em zonas verticais separadas por linhas.