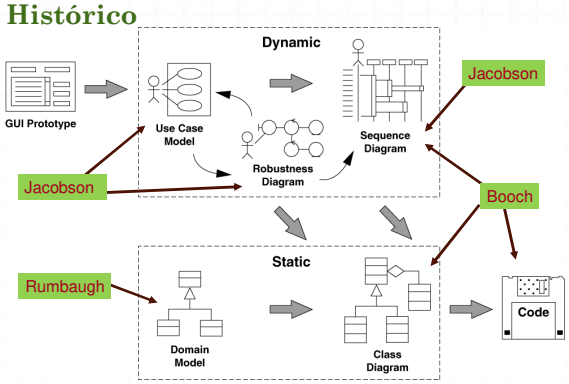
Análise e Projeto de Sistemas II

Processo Iconix

Prof.ª Carla Faria
carlafaria@ifma.edu.br
Última atualização 04/2020

Histórico



4

O que é ICONIX?

- É um processo de desenvolvimento de software orientado a objetos elaborado por Doug Rosenberg e Kendall Scott.
- Tem como característica ser dirigido por caso de uso
- É um modelo iterativo-incremental
- Tem como característica ser um processo prático, intermediária entre a complexidade do RUP(Rational Unified Process) e a simplicidade do XP(Extreme Programming), sem deixar a desejar na Análise e Projeto.
- Utiliza a linguagem UML.

2

Histórico

- Processo Unificado Rational - RUP
  - Processo muito burocrático(utiliza muita documentação)
  - Por ser complexo, paralisa a análise
- Extreme Programming - XP
  - Codificar desde o início
  - "Sem documentação formal", "O projeto é o código", "Projeto baseado em Teste"
- É possível ter um meio termo entre RUP e XP?

5

Histórico

- No início, haviam três tipos diferentes de métodos AOO: Método de Booch, Método Objectory e Técnica de Modelagem de Objetos (OMT).
- A idéia foi unir os pontos fortes de cada método:
  - OMT (Rumbaugh)
    - Modelo de Objetos do Domínio do Problema
  - Objetory (Jacobson)
    - Modelo de Domínio de Solução dirigida pelo usuário
  - Booch (Booch)
    - Modelos a nível de projeto detalhados
- Estes métodos forneciam notações, mas não processos, ou seja, eles diziam que diagramas utilizar para representar partes do nosso sistema, mas não diziam como os diagramas se relacionavam.

3

Histórico

- É possível ter Análise e Projeto OO sem precisar paralisar nosso processo, desde que eles sejam simples.
- Surge o ICONIX, uma abordagem prática para desenvolvimento de softwares, que ajuda você a ir dos casos de uso para a codificação rápida e eficientemente, utilizando um subconjunto da UML.

6

Características Principais

- Impede a paralisia na análise
- Uso moderado da UML
  - Apenas um subconjunto da UML é utilizado
- Enxuto mas eficiente
- Alto grau de rastreabilidade
  - Possui mecanismos para verificar, em todas as fases, se os requisitos estão sendo atendidos

7

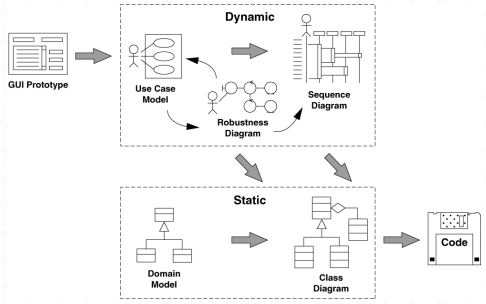
Visão Geral

- Visão Estática
  - Mostra o funcionamento do sistema sem nenhum dinamismo e interação do usuário
  - Modelo de Domínio e Diagrama de Classe
- Visão Dinâmica
  - Mostra o usuário interagindo com o sistema
  - Diagrama de Caso de Uso, Diagrama de Robustez, Diagrama de Seqüência

8

Visão Geral

- Visão Estática e Dinâmica



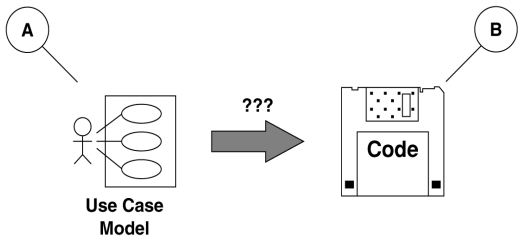
9

Visão Geral

- O processo ICONIX trabalha a partir de um protótipo de interface onde se desenvolvem os diagramas de caso de uso baseados nos requisitos do usuário. Com base nos diagramas de caso de uso se faz a análise de robustez para cada caso de uso e com os resultados obtidos, é possível desenvolver o diagrama de seqüência e, posteriormente, povoar o modelo de domínio já revisado com métodos e atributos descobertos originando, assim, o diagrama de classe.

10

Visão Geral



Como ir do Diagrama de Caso de Uso para a Codificação?

11

Visão Geral

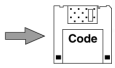
- Para entender como chegar à codificação a partir do diagrama de caso de uso, vamos analisar o funcionamento do ICONIX de forma invertida, ou seja, partindo do código em direção ao diagrama de caso de uso.

12

Visão Geral

?

Nós temos um pequeno protótipo e escrevemos alguns casos de uso.



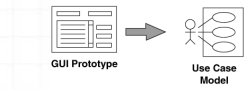
13

Visão Geral

- Mas antes de chegarmos ao código:
  - Nós precisamos de um conjunto completo de classes, com métodos e atributos
  - Nós mostramos essas informações em diagramas de classes a nível de projeto.

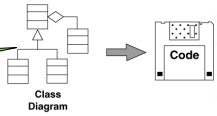
14

Visão Geral



?

Os diagramas de classe a nível de projeto servem como a estrutura do nosso código



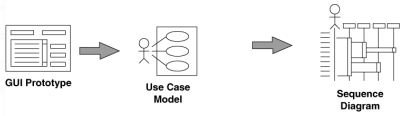
15

Visão Geral

- Mas antes de termos classes com atributos e métodos:
  - Nós precisamos alocar comportamento para as classes.
  - Para isto, utilizamos o diagrama de seqüência, que é uma excelente ferramenta para decidir em que classe cada método vai ficar.
  - Sendo assim, nós elaboramos um diagrama de seqüência para cada caso de uso.

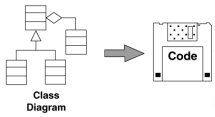
16

Visão Geral



?

Nós alocamos métodos para as classes durante a elaboração do diagrama de seqüência



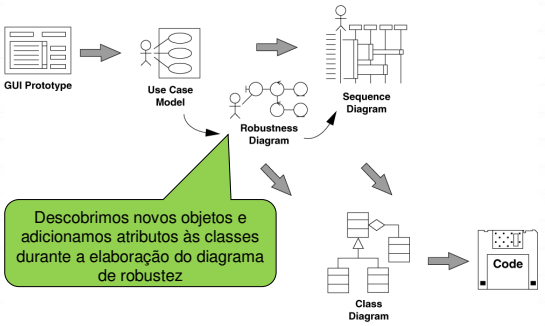
17

Visão Geral

- Mas antes de elaborarmos diagramas de seqüências:
  - Nós precisamos ter uma boa idéia de que objetos estarão executando em quais casos de uso, e que funções o sistema executará como resultado de ações do usuário.
  - Obtemos esta informação através do Diagrama de Robustez.

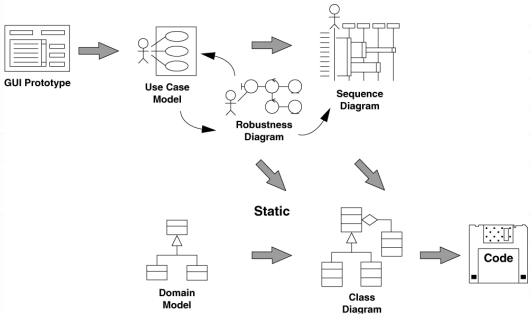
18

Visão Geral



19

Visão Geral



22

Visão Geral

- Mas para elaborar diagramas de robustez:
  - Nós precisamos descrever o uso do sistema no contexto do modelo de objetos.
  - Para isso, nós precisamos escrever casos de uso que não sejam vagos, abstratos, de modo que possamos criar o projeto a partir deles.
  - Nós precisamos escrever casos de uso que referenciem os nomes dos objetos utilizados no modelo de domínio.

20

Visão Geral

- Nós refinaremos nosso diagrama de classe a nível de análise(modelo de domínio) continuamente à medida em que nós exploramos o comportamento dinâmico do sistema em mais e mais detalhes durante a análise e projeto.
- Isto resultará no diagrama de classe a nível de projeto, a partir do qual nós podemos codificar.

23

Visão Geral

- E antes de tudo, para iniciar:
  - Nós precisamos identificar as abstrações principais que estão presentes no domínio do problema, ou seja, nós precisamos um modelo de domínio.
  - Para isto, utilizamos o diagrama de classe para representar o modelo de domínio.

21

Passos

1. Elaborar o Protótipo da Interface
2. Identificar Objetos do Domínio
   (Modelo de Domínio)
3. Definir o Comportamento dos Requisitos
   (Diagrama de Caso de Uso)
4. Executar a Análise de Robustez
   (Diagrama de Comunicação)
5. Alocar Comportamentos para as Classes
   (Diagrama de Seqüência)
6. Finalizar o Modelo Estático
   (Diagrama de Classe)
7. Codificar

24

1 2 3 4 5 6

Identificar Objetos do Domínio

- Tem o propósito de identificar abstrações no mundo real que serão os objetos principais do nosso futuro sistema, bem como os relacionamentos entre os mesmos.
- É representado pelo Modelo de Domínio, que é um diagrama de classes a nível de análise, simplificado (sem atributos ou métodos).
- Esta representação não tem pretensão de ser perfeita e completa, na verdade ao longo do processo este modelo será revisado diversas vezes até chegar a um ponto considerado ideal.
- Este diagrama será a base para o diagrama de classes a nível de projeto.

25

1 2 3 4 5 6

Construindo Modelos de Domínio

- Analisar o domínio do problema, identificando as possíveis classes.
- Este é um bom momento de identificarmos os relacionamentos de generalização e especialização que ocorrem entre as classes.

26

1 2 3 4 5 6

Construindo Modelos de Domínio

- Dicas:
  - Não perca tempo tentando descobrir multiplicidades e associações entre as classes. Neste ponto do processo, ainda não temos informação suficiente para fazer boas escolhas neste sentido. Isto pode causar paralisia de análise.
  - Não perca tempo tentando descobrir operações para as classes do modelo de domínio, somente após a análise de robustez e o diagrama de seqüência é que teremos condições de fazer isso com mais precisão.
  - Não pense em reutilização de código, mas sim em atender os requisitos do usuário.
  - Com relação às associações entre as classes, não perca tempo decidindo se é agregação ou composição.
  - Utilize nomes óbvios e auto-explicativos para as classes.

27

1 2 3 4 5 6

Estudo de Caso

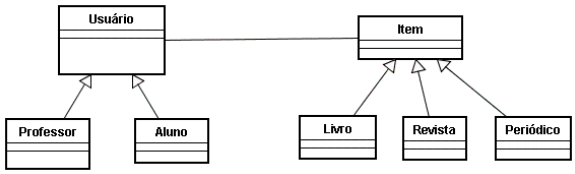
- Suponhamos que queremos desenvolver um software para informatização de uma biblioteca onde um dos requisitos seria que os usuários pudessem pegar emprestado itens do seu acervo. Os usuários podem ser alunos ou professores. Eles poderão reservar itens que não estão disponíveis. Os professores poderão pegar até 15 itens simultaneamente por até 20 dias, enquanto que os alunos, 3 itens por apenas 7 dias. Haverá uma multa diária por atraso na devolução de itens.

28

1 2 3 4 5 6

Modelo de Domínio

- Exemplo



29

1 2 3 4 5 6

Definir o Comportamento dos Requisitos

- Tem o propósito de identificar as funcionalidades do sistema sob o ponto de vista dos usuários.
- Ele deve detalhar, de forma clara e legível, todos os cenários que os usuários executarão para realizar alguma tarefa.
- Representado pelo diagrama de caso de uso.

30

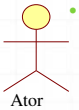
Diagrama de Caso de Uso

- O Diagrama de Caso de Uso tem por objetivo apresentar uma visão externa das funcionalidades que o sistema deverá oferecer aos usuários.
- É de grande auxílio para a compreensão dos requisitos do sistema, ajudando a especificar, visualizar e documentar as características, funções e serviços do sistema desejados pelo usuário.
- Identifica os usuários que irão interagir com o sistema, quais papéis esses usuários irão assumir e quais funções um usuário específico poderá requisitar.

31

Diagrama de Caso de Uso

- É formado por atores, casos de uso e relacionamentos.
- Ator



  - Representa um papel que o usuário pode desempenhar no sistema a ser modelado.
    - Pessoas
    - Dispositivos de Hardware
    - Outros Softwares
  - Mais precisamente um ator é uma representação de uma coleção de papéis que objetos externos podem desempenhar quando interagem com o software.
  - Tipos de atores:
    - Principal
    - Secundário

32

Diagrama de Caso de Uso

- Caso de Uso:

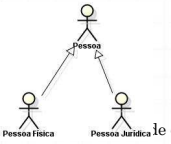
Caso de Uso

  - É uma seqüência de ações que um ator (pessoa, entidade externa ou outro sistema) executa no sistema para atingir um objetivo particular.
- Detalhamento do Caso de Uso:
  - Nome do Caso de Uso
  - Descrição Breve
    - Descrição do objetivo do caso de uso
  - Atores
    - Atores que participam do caso de uso (principal e secundário)
  - Cenário Principal
    - Descrição do acontecimento de forma correta
  - Cenário Alternativo
    - Descrição do que pode dar errado

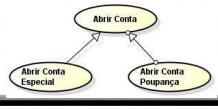
33

Diagrama de Caso de Uso

- Relacionamentos
  - Atores e Casos de Uso
    - Generalização / Especialização
      - Define tipos gerais e especializados de atores



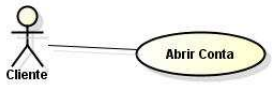
      - Define tipos gerais de casos de uso



34

Diagrama de Caso de Uso

- Relacionamentos
  - Atores e Casos de Uso
    - Associação
      - Define uma ligação entre o ator e os casos de uso
        - Demonstra que o ator utiliza a funcionalidade do sistema representada pelo caso de uso em questão



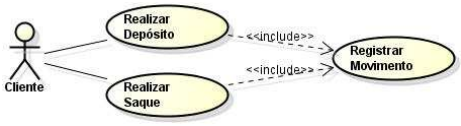
35

Diagrama de Caso de Uso

- Relacionamentos
  - Casos de Uso
    - Include (Use) - Obrigatoriedade

<<include>>

      - Significa que um caso de uso explicitamente incorpora o comportamento de outro caso de uso.
        - Usa-se esse relacionamento para evitar a descrição do mesmo fluxo de eventos várias vezes.



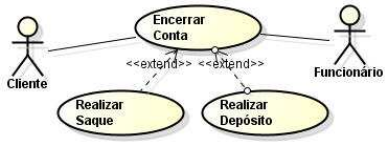
36

Diagrama de Caso de Uso

- Relacionamentos
  - Casos de Uso
    - Extend - Opcionalidade

<<extend>>

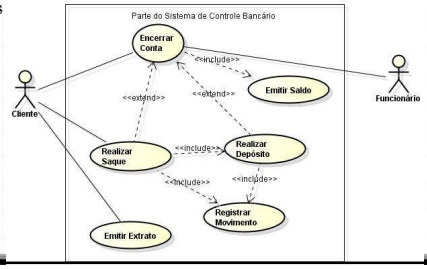
      - Significa que um caso de uso pode incorporar o comportamento de outro caso de uso de forma indireta estendendo o caso de uso.
        - Usa-se esse relacionamento para modelar uma parte desse caso de uso onde o usuário enxerga como um comportamento opcional;
        - Usa-se esse relacionamento para modelar vários fluxos de eventos que podem ser inseridos em um dado momento.



37

Diagrama de Caso de Uso

- Fronteira do Sistema
  - Permite identificar um subsistema ou mesmo um sistema completo, além de destacar o que está contido no sistema e o que não está. Normalmente os atores são externos



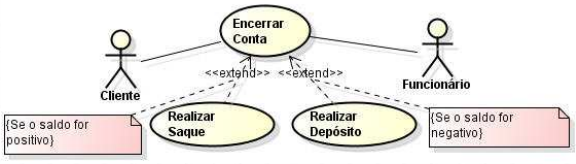
40

Diagrama de Caso de Uso

- Restrições em Associações de Extensão

{Restrição}

  - Restrição é um texto entre chaves em uma nota explicativa. São utilizadas para definir validações, consistências, condições, que devem ser aplicadas, para que um caso de uso estendido seja executado.



38

Diagrama de Caso de Uso

- Como construir:
  - Identificar tantos quantos casos de uso pudermos, descrever cada caso de uso detalhadamente com clareza e sem ambigüidades, manter um ciclo de refinamento de texto contínuo. É comum durante a descrição encontrar novos casos de uso.
  - O formato básico do texto é substantivo-verbo-substantivo.
  - Atualizar o modelo de domínio, pois é comum encontrarmos novos objetos e isso pode influenciar na compreensão do problema previamente modelado.

41

Diagrama de Caso de Uso

- Extend Point – Ponto de Extensão

Caso de Uso
extension points
ExtensionPoint4

  - Um Ponto de Extensão identifica um ponto no comportamento de um caso de uso a partir do qual esse comportamento poderá ser estendido pelo comportamento de outro caso de uso, se a condição para que isso ocorra for satisfeita.

39

Diagrama de Caso de Uso

- Como construir: (cont.)
  - Durante a descrição dos casos de uso procurar definir claramente os cursos de ação básicos (realização da ação pelo usuário) e os cursos alternativos (algo que impeça o curso de ação básico).
  - Verificar se todos os casos de uso atendem as necessidades funcionais desejadas no sistema.
  - Verificar se para cada caso de uso foi descrito o curso básico de ação e o curso(s) alternativo(s).

42

Diagrama de Caso de Uso

- Dicas:
  - É preciso descrever todos os detalhes de ações do usuário e respostas do sistema. Isso servirá para uma boa análise robusta. Também é bom lembrar que os casos de uso servirão para construir o manual de usuário. É melhor errar por excesso de detalhe na documentação de usuário.
  - Não escrever os textos na voz passiva.
  - Não perder muito tempo pra decidir em usar estereótipos "include" e "extends".

43

Diagrama de Caso de Uso

- Exemplo:
- Para identificar atores:
  - Examinamos o modelo de domínio em busca de elementos que interagem com o sistema
  - Os seguintes questionamentos auxiliam na identificação de atores:
    - Quem utiliza o sistema?
    - Que outros sistemas se comunicam com o sistema modelado?
    - Quem obtêm informações por meio do sistema modelado?
    - Quem provê informações ao sistema modelado?
  - Os atores:
    - Principal
    - Secundário

44

Diagrama de Caso de Uso

- Atores:
  - Principais
    - Usuário
    - Professor
    - Aluno
    - Bibliotecário
    - Atendente
  - Secundários
    - Editora

45

Diagrama de Caso de Uso

- Para identificar casos de uso:
  - Para cada ator principal identificado temos que identificar qual é o seu objetivo particular no sistema.
  - À partir do objetivo particular de cada ator são identificados o que ele precisa fazer para alcançar o seu objetivo que originarão os casos de uso.
- Usuário:
  - Objetivo: Locar Item
- Atendente
  - Objetivo: Auxiliar no gerenciamento da biblioteca
- Gerente
  - Objetivo: Gerenciar a biblioteca

46

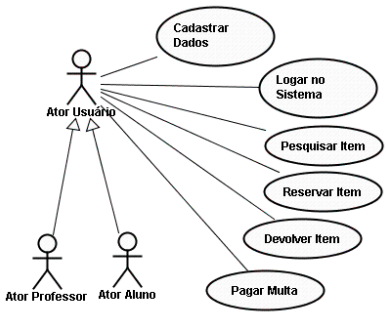
Diagrama de Caso de Uso

- Caso de Uso
  - Usuário: Locar Item
    - Cadastrar Dados
    - Logar no Sistema
    - Pesquisar Item
    - Reservar Item
    - Devolver Item
    - Pagar Multa

47

Diagrama de Caso de Uso

- Exemplo:



48

Detalhamento do Caso de Uso

- Para detalhar casos de uso:
  - Para cada caso de uso identificado temos que fazer o detalhamento, que é a descrição do fluxo de eventos do caso de uso em detalhes:
    - Contexto
    - Atores
    - Cenário principal
    - Cenário alternativo

49

Detalhamento do Caso de Uso

- Exemplo:
- Caso de Uso: Logar no Sistema
  - Contexto
    - Por meio deste caso de uso os usuários da biblioteca passam a ter acesso a funcionalidade do sistema.
  - Ator
    - Usuários
  - Pré-condição **(Erro 1)**
    - Usuário não esta logado no sistema
  - Pós-condição **(Erro 1)**
    - Usuário logado no sistema

Erro 1: Escrever textos com informações desnecessárias

50

Detalhamento do Caso de Uso

- Cenário Principal:
  1. O sistema exibe uma tela **(Erro 2)**
  2. O usuário entra com dados **(Erro 3 e Erro 4)** **(Erro 6)**
  3. O método validar dados é executado **(Erro 5)**

Erro 2: Evitar nomes genéricos para os objetos de interface

Erro 3: Escrever casos de uso concisamente

Erro 4: Descrever casos de uso sem ser específico sobre como as ações do usuário executarão em suas telas

Erro 5: Descrever métodos ou atributos na descrição do caso de uso

Erro 6: Descrever somente interações do usuário

51

Detalhamento do Caso de Uso

- Cenário Alternativo:
  a. Se os dados forem inválidos o sistema exibe uma mensagem de erro **(Erro 7)**

Erro 7: Omitir descrições no cenário alternativo

52

Detalhamento do Caso de Uso

- Exemplo:
- Caso de Uso: Logar no Sistema
  - Contexto
    - Por meio deste caso de uso os usuários da biblioteca passam a ter acesso a funcionalidade do sistema.
  - Ator
    - Usuários
  - Cenário Principal:
    1. O caso de uso inicia quando o usuário clica em login na tela principal **(Solução 2) (Solução 4)**

Solução 1: Retirar pré e pós-condição

Solução 4: Especificar as ações do usuário

Solução 2: Nome explícito para o objeto de interface

53

Detalhamento do Caso de Uso

- Cenário Principal (cont.):
  2. O sistema exibe a tela de login **(Solução 6)**
  3. O usuário entra com nome e senha e clica em Ok **(Solução 3)**
  4. O sistema valida o nome e a senha do usuário **(Solução 5)**
  5. O sistema exibe a tela principal
  6. O caso de uso termina com sucesso

Solução 6: Descrever as respostas do sistema

Solução 3: Escrever detalhes das ações dos usuários

Solução 5: Descrever o que o sistema fará

54

1 2 3 4 5 6

Detalhamento do Caso de Uso

- Cenário Alternativo:
  - 4a. Nome e senha de usuário inválido
    - 4a1. O sistema exibe uma mensagem de erro
    - 4a2. O fluxo volta ao passo 3 do cenário principal

(Solução 7)

Solução 7: Descrever todos os detalhes do cenário alternativo

1 2 3 4 5 6

Detalhamento do Caso de Uso

- Cenário Alternativo:
  - a. A qualquer momento o usuário pode cancelar a pesquisa
  - 4a. O sistema não valida a palavra-chave
    - 4a1. O sistema exibe uma mensagem
    - 4a2. O fluxo volta ao passo 3 do cenário principal
  - 5a. O sistema não recupera uma lista de itens para a palavra-chave
    - 5a1. O sistema exibe uma mensagem
    - 5a2. O fluxo volta ao passo 3 do cenário principal

1 2 3 4 5 6

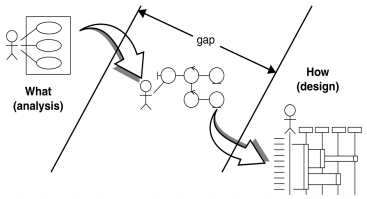
Detalhamento do Caso de Uso

- Exemplo:
- Caso de Uso: Pesquisar Item
  - Contexto
    - Por meio deste caso de uso os usuários da biblioteca podem escolher um item a ser locado. A pesquisa é feita por palavras-chave.
  - Ator
    - Usuários
  - Cenário Principal:
    1. O caso de uso inicia quando o usuário clica na pesquisa na tela principal
    2. O sistema exibe a tela de pesquisa
    3. O usuário entra com a palavra-chave e clica em Ok

1 2 3 4 5 6

Executar a Análise de Robustez

- É a ponte entre a fase de análise e a de projeto, assegurando que as descrições de casos de uso estão corretas, além de descobrir novos objetos através do fluxo de ação.



1 2 3 4 5 6

Detalhamento do Caso de Uso

- Cenário Principal (cont.):
  4. O sistema valida a palavra-chave
  5. O sistema recupera uma lista de itens
  6. O sistema exibe a lista de itens
  7. O usuário seleciona um item
  8. O sistema mostra os detalhes do item (sumário, título, editora, ano de publicação, autor)
  9. O usuário e o sistema repetem os passos 8 e 9 várias vezes
  10. O caso de uso termina com sucesso

1 2 3 4 5 6

Executar a Análise de Robustez

- O Diagrama de Comunicação representa as interações trocadas entre os objetos do sistema e os atores.
- Os elementos do diagrama:
  - Atores
  - Objetos
  - Mensagens

1 2 3 4 5 6

Executar a Análise de Robustez

- Como construir?
  - Analisar as narrativas de texto (o detalhamento) de caso de uso, identificando um conjunto de objetos que participarão de cada caso de uso.

61

1 2 3 4 5 6

Executar a Análise de Robustez

- Os objetos são classificados em três tipos:
  - **Objetos Limite ou interface (Boundary Objects)** – são usado pelos atores (por exemplo, os usuários) para se comunicarem com o sistema.
  - **Objetos de Controle (Control Objects)** - são objetos que controlam a lógica de negócio, eles fazem a conexão entre os objetos interface e objetos entidade.
  - **Objetos entidade (Entity Objects)** são responsáveis para realizar algum tipo de persistência. Geralmente eles vêm do modelo de domínio.

Representação dos tipos de objetos

62

1 2 3 4 5 6

Executar a Análise de Robustez

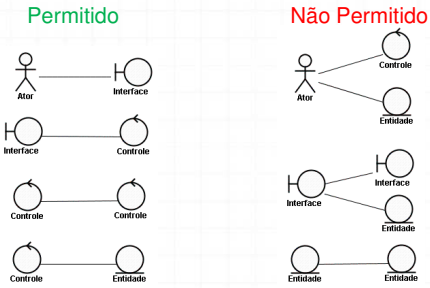
- Regras para comunicação entre os objetos:
  - Atores somente podem se comunicar com objetos interface
  - Objetos interface somente podem se comunicar com atores e controladores
  - Objetos entidade somente podem se comunicar com objetos controladores
  - Objetos controladores somente podem se comunicar com objetos entidade e interface, também com outros controladores, mas não com atores

63

1 2 3 4 5 6

Executar a Análise de Robustez

- Regras para comunicação entre os objetos:

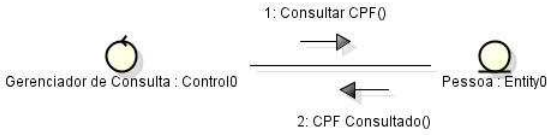


64

1 2 3 4 5 6

Executar a Análise de Robustez

- Mensagens:
  - As mensagens são numeradas indicando a ordem em que ocorrem;
  - Uma mensagem é representada por uma seta indicativa da direção para onde a mensagem foi enviada, ou seja, a ponta da seta indicará qual objeto recebeu a mensagem, enquanto o objeto na direção oposta será o responsável por seu envio.



65

1 2 3 4 5 6

Executar a Análise de Robustez

- Dicas:
  - Não violar as regras de comunicação entre os objetos.
  - Incluir os cenários alternativos no diagrama de robustez.
  - Não colocar o comportamento das classes no diagrama de robustez.

66

Executar a Análise de Robustez

- Dicas (cont.):
  - Não incluir poucos controladores. É bom ter entre dois e cinco controladores no diagrama de robustez, se tivermos apenas um controlador no diagrama de robustez o caso de uso não foi bem detalhado. Por outro lado se tivermos mais de dez controladores um caso de uso, temos que refazer esse caso de uso.
  - Manter a consistência entre o detalhamento do caso de uso e o diagrama de robustez.
  - Atualizar o modelo de domínio com os novos objetos identificados.

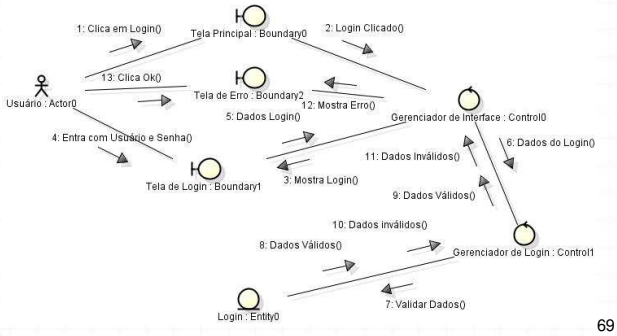
67

Diagrama de Robustez

- Exemplo:
- Para identificar e classificar os objetos e as mensagens trocadas entre eles:
  - Para cada caso de uso examinamos o seu detalhamento
- Objetos do caso de uso: Logar no Sistema
  - Objeto Interface: Tela Principal, Tela de Login, Tela de Erro
  - Objeto Controle: Gerenciador de Login, Gerenciador de Interface
  - Objeto Entidade: Login

68

Diagrama de Comunicação



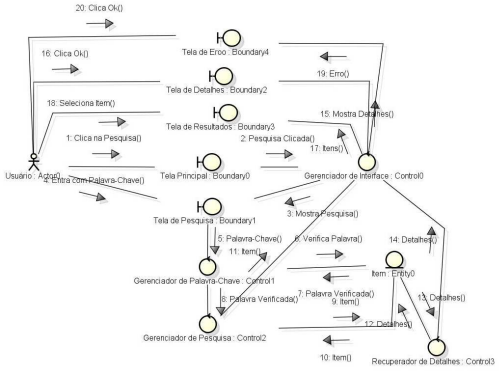
69

Diagrama de Comunicação

- Exemplo:
- Objetos do caso de uso: Pesquisar Item
  - Objeto Interface: Tela Principal, Tela de Pesquisa, Tela de Erros, Tela de Resultados, Tela de Detalhes
  - Objeto Controle: Gerenciador de Pesquisa, Navegador, Recuperador de Detalhes, Verificador de palavra-chave
  - Objeto Entidade: Item

70

Diagrama de Comunicação



71

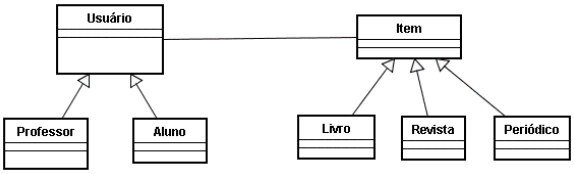
Atualização dos Casos de Uso e do Modelo de Domínio

- O detalhamento dos casos de uso tem que estar de acordo com o diagrama de robustez construído, se necessário efetue correções nos detalhamentos dos casos de uso.
- Os objetos identificados durante a análise de robustez são acrescentados ao modelo de domínio com exceção dos objetos de interface, pois fazem parte da solução do problema.

72

Modelo de Domínio

- Exemplo:



73

Alocar Comportamentos para as Classes

- O objetivo é construir um modelo dinâmico (diagrama de sequência) entre o usuário e o sistema.
  - Alocar comportamentos para os objetos entidade, controle e interface identificados durante a execução da análise de robustez.
  - Mostrar as interações detalhadas que ocorrem na seqüência entre os objetos associados com o texto de caso de uso. Os objetos interagem trocando mensagens entre si. Uma mensagem estimula um objeto a executar uma ação desejada. Para cada comportamento de um caso de uso, temos que identificar as mensagens necessárias e os métodos.
  - Finalizar a distribuição dos métodos entre as classes.

74

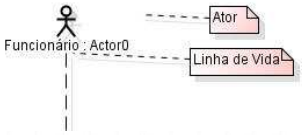
Alocar Comportamentos para as Classes

- O diagrama de sequência determina a sequência de eventos que ocorrem em um determinado processo, identificando quais mensagens devem ser disparadas entre os elementos envolvidos e em que ordem.
- Os elementos do diagrama de sequência:
  - Atores
  - O **texto** que descreve os fluxos de ação para os casos de uso;
  - Os **objetos** que interagem entre si;
  - **Mensagens** são as setas entre os objetos;
  - **Métodos**;
  - **Linha de Vida;**

75

Alocar Comportamentos para as Classes

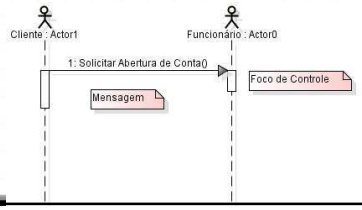
- Atores
  - São os mesmos do diagrama de casos de uso e do diagrama de robustez.
- Linha de Vida
  - Representa o tempo em que um objeto existe durante o processo.



76

Alocar Comportamentos para as Classes

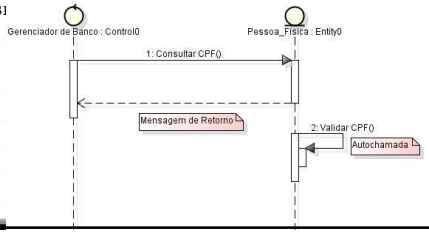
- Foco de Controle
  - Identifica os momentos em que um objeto está executando um ou mais métodos utilizados em um processo específico.
- Mensagens
  - Entre atores, entre atores e objetos e entre objetos.



77

Alocar Comportamentos para as Classes

- Mensagens de Retorno
  - É uma resposta a uma mensagem para o objeto ou ator que a chamou.
- Autochamadas
  - São mensagens que um objeto envia para si mesı



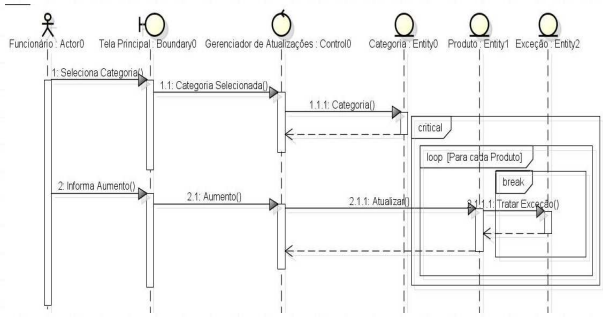
78

Alocar Comportamentos para as Classes

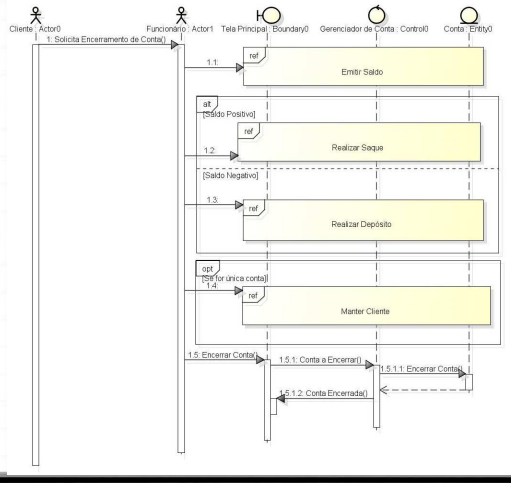
- Ocorrências de Interação
  - Ref - Serve para referenciar um diagrama de sequência já existente. Assim, faz-se uma referência a esse diagrama por meio de uma ocorrência de interação.
- Fragmentos Combinados e Operadores de Interação
  - Alt (Alternativa) – Este operador de interação define que o fragmento combinado representa uma escolha entre dois ou mais comportamentos. A condição é representada entre colchetes.
  - Opt (Opção) – Este operador de interação define que o fragmento combinado representa uma escolha de comportamento, onde esse comportamento será ou não executado.

79

Alocar Comportamentos para as Classes



82



80

Alocar Comportamentos para as Classes

- Como construir:
  - Para cada detalhamento de caso de uso e diagrama de robustez correspondente é elaborado um diagrama de sequência
    - Colocar o cenário principal e o alternativo no diagrama de seqüência;
    - Adicionar os atores, os objetos interface e os objetos entidade do diagrama de robustez;
    - Converter os objetos controladores do diagrama de robustez para um conjunto de métodos e mensagens, que envolvem o comportamento desejado dos objetos. Ocasionalmente um controlador pode se tornar um objeto real de controle.

83

Alocar Comportamentos para as Classes

- Fragmentos Combinados e Operadores de Interação
  - Par (Paralelo) – Este operador de interação define que o fragmento combinado representa uma execução paralela de dois ou mais comportamentos.
  - Loop (Laço) – Este operador de interação define que o fragmento combinado representa um laço que poderá ser repetido diversas vezes.
  - Break (Quebra) - – Este operador de interação indica uma quebra na execução normal do processo. É usado no tratamento de exceções.
  - Critical Region (Região Crítica) – Este operador de interação identifica uma operação atômica que não pode ser interrompida por outro processo até ser totalmente concluída.

81

Alocar Comportamentos para as Classes

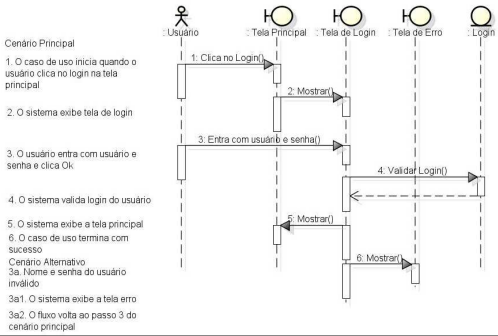
- Dicas:
  - Elaborar um diagrama de sequência para cada caso de uso;
  - Colocar o detalhamento do caso de uso (cenário principal e alternativo) no diagrama de sequência;
  - Identificar todos os objetos necessários à partir do diagrama de robustez;
  - Cada seta de mensagem deve estar alinhada com a descrição de sua ação, isso torna mais fácil a compreensão e garante a identificação de possíveis problemas;
  - Alocar o comportamento das classes corretamente;
  - Focalizar em métodos importantes, não perder tempo com get() e set();
  - Definir bem as mensagens que são trocadas entre os objetos;

84

1 2 3 **4** 5 6

Diagrama de Sequência

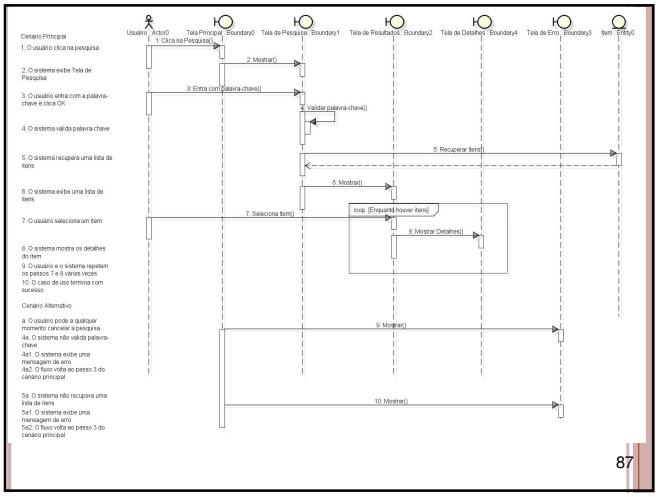
- Exemplo Logar no Sistema:



1 2 3 **4** 5 6

Diagrama de Sequência

- Exemplo Pesquisar Item:



1 2 3 4 **5** 6

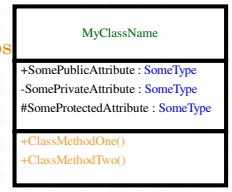
Finalizar o Modelo Estático

- O propósito deste passo é finalizar a elaboração do diagrama de classe que já vem sendo elaborado desde o primeiro passo do Processo ICONIX.
- Diagrama de Classe permite a visualização das classes que comporão o sistema com seus respectivos atributos e métodos, bem como em demonstrar como as classes do diagrama se relacionam, complementam e transmitem informações entre si.
- É composto de classes (com atributos e métodos) e seus relacionamentos definindo a estrutura do sistema.

1 2 3 4 **5** 6

Finalizar o Modelo Estático

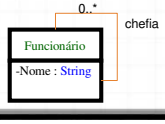
- Classes podem ter três partes:
  - Nome
  - Atributos e seus tipos de dados
  - Operações
- As classe podem mostrar visibilidade e tipos
- Apenas o nome é obrigatório.



1 2 3 4 **5** 6

Finalizar o Modelo Estático

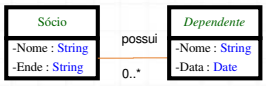
- Classes
  - Abstratas: não pode ser instanciada
  - Concretas: pode ser instanciada
- Relacionamentos ou Associações
  - Permitem que elas compartilhem informações entre si e colaborem para a execução de processos executados pelo sistema.
    - Associação Unária ou Reflexiva
      - Relação de um objeto de uma classe com objetos da mesma classe.



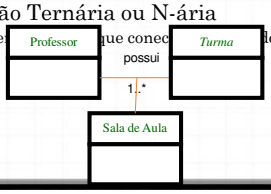
1 2 3 4 5 6

Finalizar o Modelo Estático

- Associação Binária
  - Relação entre objetos de duas classes distintas.



- Associação Ternária ou N-ária
  - Relação e[illegible]que conec[illegible]le mais de duas classes.

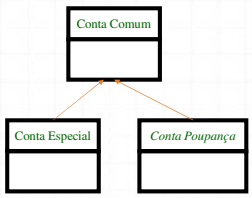


91

1 2 3 4 5 6

Finalizar o Modelo Estático

- Generalização / Especialização
  - Define a hierarquia das classes (superclasses e subclasses)

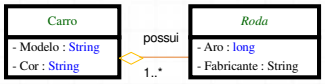


94

1 2 3 4 5 6

Finalizar o Modelo Estático

- Agregação
  - As informações de um objeto (objeto todo) precisam ser complementadas pelas informações contidas em um ou mais objetos de outra classe (objetos parte)
  - Um objeto contém uma lista de outros objetos.
  - Os objetos contidos podem existir sem serem parte do objeto que os contém.
  - Exemplo: Carro -> Rodas. Você pode tirar as rodas do carro antes de destruí-lo e elas podem ser colocadas em outro carro.



92

1 2 3 4 5 6

Finalizar o Modelo Estático

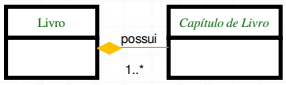
Multiplicidade	Significado
0..1	No mínimo 0 e no máximo 1. Indica que os objetos das classes associadas não precisam obrigatoriamente estar relacionadas, mas se houver relacionamento indica que apenas uma instância da classe relaciona-se com as instâncias da outra classe.
1..1	1 e somente 1. Indica que apenas um objeto da classe relaciona-se com os objetos da outra classe.
0..*	No mínimo nenhum e no máximo muitos. Indica que pode ou não haver instâncias da classe participando do relacionamento.
*	Muitos. Indica que muitos objetos da classe estão envolvidos na associação.
1..*	No mínimo um e no máximo muitos. Indica que há pelo menos um objeto envolvido no relacionamento, podendo haver muitos objetos envolvidos.
3..5	No mínimo três e no máximo cinco. Indica que existe pelo menos três instâncias envolvidas no relacionamento e que podem ser quatro ou cinco as instâncias envolvidas, não mais do que isso

95

1 2 3 4 5 6

Finalizar o Modelo Estático

- Composição
  - Os objetos parte têm de estar associados a um único objeto todo. Os objetos parte não podem ser destruídos por um objeto diferente do objeto todo ao qual estão relacionados
  - Um objeto contém uma lista de outros objetos.
  - Os objetos contidos não fazem sentido fora do contexto do objeto que os contém.
  - Exemplo: Pedido -> Itens. Se você destruir o pedido, os itens são destruidos junto, eles não tem sentido fora do pedido.

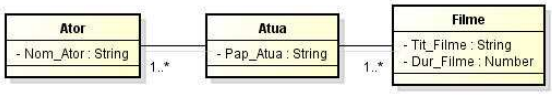


93

1 2 3 4 5 6

Finalizar o Modelo Estático

- Classe Associativa
  - São aquelas produzidas quando da ocorrência de associações que tenham multiplicidade muitos (*) em todas as suas extremidades.



96

Finalizar o Modelo Estático

- Dependência
  - Identifica o grau de dependência de uma classe em relação a outra classe.
- Realização
  - Identifica classes responsáveis por executar funções para outras classes.

97

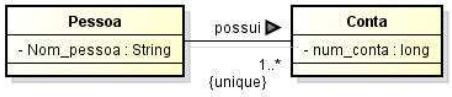
Finalizar o Modelo Estático

- Restrição
  - Informações extras que definem condições a serem validadas durante a implementação dos métodos de uma classe, das associações entre as classes ou mesmo de seus atributos.

98

Finalizar o Modelo Estático

- Tipos de Restrição:
  - Unique
    - Determina que um elemento na coleção não pode se repetir
  - Bag
    - Permite que um mesmo elemento apareça mais de uma vez
  - Sequence
    - Representa uma seqüência



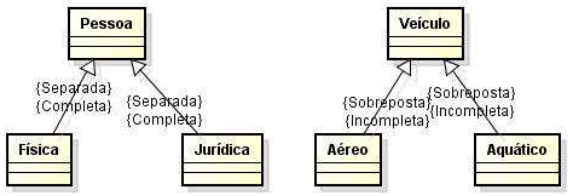
99

Finalizar o Modelo Estático

- Tipos de Restrição:
  - Completa
    - Quando todas as subclasses possíveis forem derivadas da classe geral.
  - Incompleta
    - Quando ainda é possível derivar novas subclasses.
  - Separada ou disjunta
    - Quando as subclasses são mutuamente exclusivas
  - Sobreposta
    - Quando o fato de pertencer a uma subclasse não impede que pertença a outras.

100

Finalizar o Modelo Estático



101

Finalizar o Modelo Estático

- Estereótipos
  - <<boundary>>
  - <<control>>
  - <<entity>>

102

1 2 3 4 **5** 6

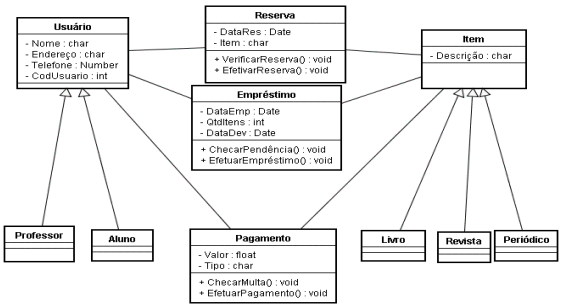
Finalizar o Modelo Estático

- Como construir:
  - Baseado nos diagramas de seqüência elaborados:
    - Adicionar atributos e métodos que foram identificados durante a alocação de comportamentos para as classes;
    - Acrescentar a visibilidade nas classes;
    - Podem ser utilizados padrões de projeto;
    - Verificar se o projeto satisfaz a todos os requisitos identificados.

1 2 3 4 **5** 6

Diagrama de Classe

- Exemplo:



1 2 3 4 5 **6**

Codificação

- O propósito deste passo é traduzir de forma correta os artefatos produzidos para um produto de software.
- O produto de software deve refletir as funcionalidades requeridas e identificadas durante a elaboração dos diagramas.

Bibliografia

- José Anízio Maia, **"Construindo Softwares com Qualidade e Rapidez usando ICONIX"**, www.jugmanaus.com
- Cristina Bona, **"Avaliação de Processos de Softwares: Um Estudo de Caso em XP e ICONIX"**, Dissertação apresentada ao Programa de Pós-Graduação em Engenharia de Produção da Universidade Federal de Santa Catarina, 2002.
- ROSENBERG, D., STEPHENS, M. and COLLINS-COPE, M. **"Agile Development with ICONIX Process"**, Includes Index, 2005.
- ROSENBERG, D. and SCOTT, K. **"Applying Use Case Driven Object Modeling with UML: An Annotated e-Commerce Example"**, Addison Wesley, 2001.
- ROSENBERG, D., STEPHENS, M. and COLLINS-COPE, M. **"Agile Development with ICONIX Process: People, Process and Pragmatism"**, Apress, 2005.

Links Interessantes

- http://www.iconixsw.com

Dúvidas?