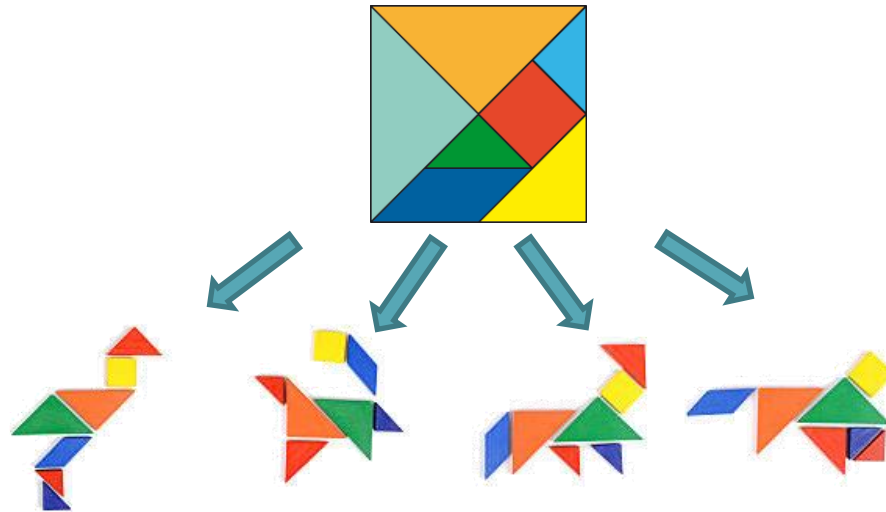


Engenharia de Software

Reutilização de software

Profa. Dra. Jacilane Rabelo



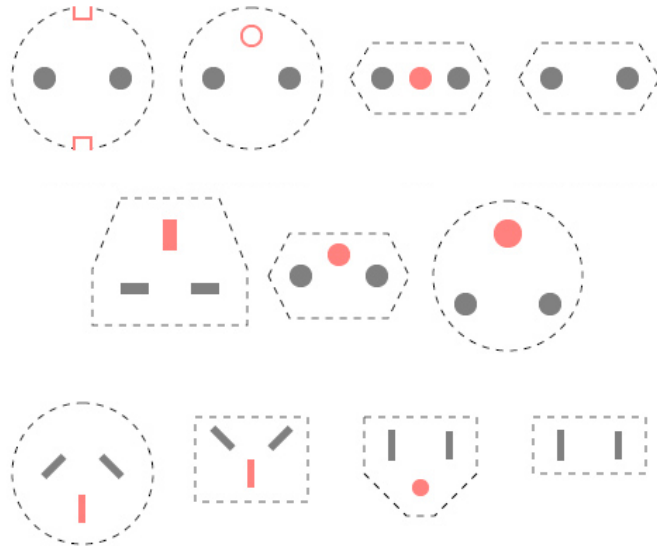
Introdução

- © Você já precisou ligar seu laptop na tomada num país estrangeiro (ou até mesmo aqui na faculdade)?



Introdução

© Existem diversas tomadas...



Introdução

© Você compraria um novo carregador toda vez que você viajasse?



Introdução

- ◎ Uma solução... Mais barata =)
- ◎ Reusar componentes (unidades ou partes) para resolver um problema diferente.

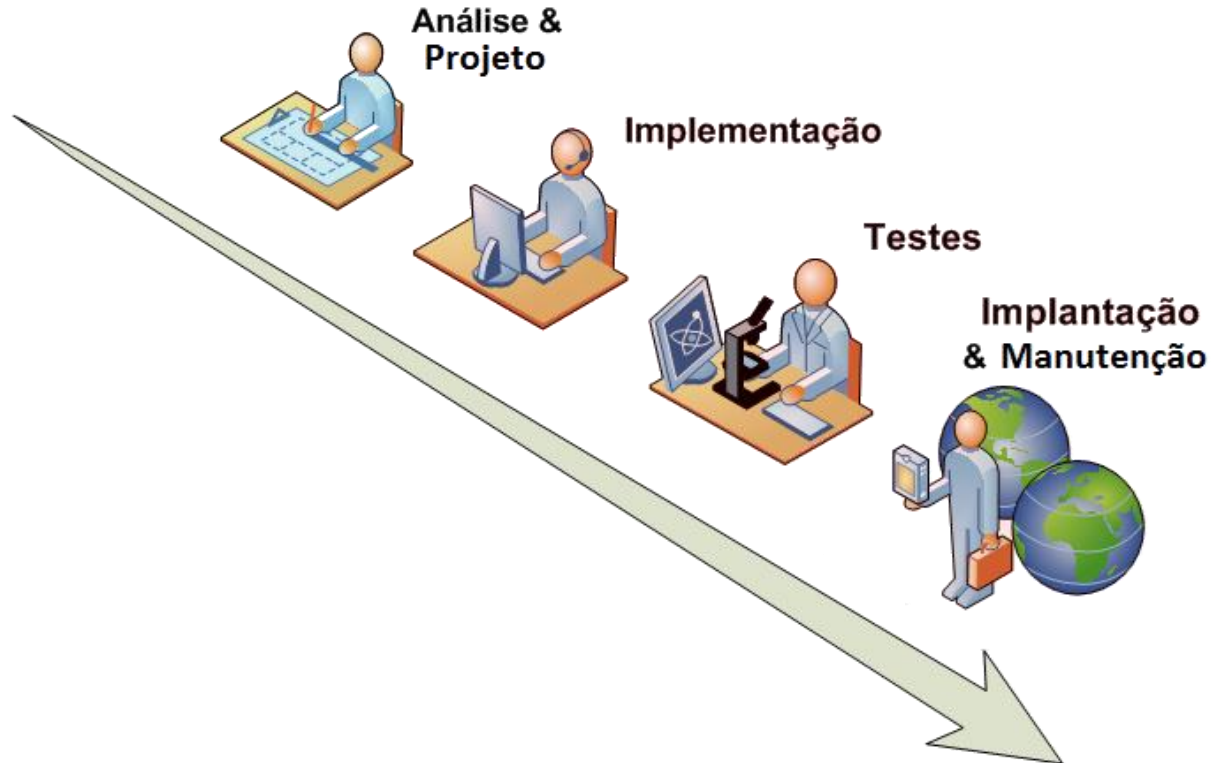


E em software?



Relembrando as aulas passadas...

No desenvolvimento tradicional



No processo de desenvolvimento...

A engenharia de software, até então, tinha como base o desenvolvimento tradicional.

Já pensou em:

- ⦿ ter que construir o seu código **do zero**?
- ⦿ ter que implementar algoritmos que já foram **pensados por alguém**?
- ⦿ propor soluções para problemas que **já foram resolvidos**?



No processo de desenvolvimento...

Como podemos alcançar software com **mais qualidade**, de forma **mais rápida** e com **baixo custo**?



já foram pensados por alguém?

propor soluções para problemas que já foram resolvidos?

E agora?



Reuso

- ◎ A **Reutilização** é inerente ao processo de solução de problemas utilizado pelos seres humanos
- ◎ Na medida em que soluções são encontradas, estas são utilizadas em **problemas similares**
- ◎ Nossa capacidade de abstração **garante a adaptação** necessária ao novo contexto
- ◎ O **problema**, portanto, **não é a falta de reutilização** na Engenharia de Software, mas **conhecer formas de aplicá-la**



Como podemos fazer reuso?

Segundo Sommerville (2007), pode ser feito:

- ◎ **Reuso de Sistemas** - Uma aplicação pode ser reutilizada incorporando-se à outros sistemas sem necessidade de mudança ou com algumas configurações
- ◎ **Reuso de Componentes** - Componentes de software que implementam um conjunto de funções podem ser reutilizados
- ◎ **Reuso de Funções** - Componentes de software que implementam uma única função podem ser reutilizados

Por que Reusar?



Por que Reusar?

- ◎ **Maior confiabilidade** - Componentes já utilizados e testados em outros sistemas são mais confiáveis que novos componentes
- ◎ **Redução dos riscos de processo** - Menos incerteza nos custos de reuso comparado aos custos de desenvolvimento



Por que Reusar?

- ◎ **Uso efetivo de especialistas** - O especialista desenvolve software reutilizável encapsulando seu conhecimento, ao invés de desenvolver as mesmas funcionalidades repetidas vezes em diferentes projetos
- ◎ **Uso efetivo de padrões** - O uso de padrões organizacionais agiliza o desenvolvimento, pois estabelece uma base comum de comunicação e garante a consistência



Por que Reusar?

- © **Desenvolvimento acelerado** - Evitando o desenvolvimento de produtos “originais” é possível acelerar a produção e a validação



Sempre Funciona?



Dificuldades do Reuso

- ⊙ Identificação, recuperação e modificação de artefatos reutilizáveis
- ⊙ Compreensão dos artefatos recuperados
- ⊙ Qualidade de artefatos reutilizáveis
- ⊙ Composição de aplicações a partir de componentes
- ⊙ Barreiras psicológicas, legais e econômicas
- ⊙ Necessidade da criação de incentivos à reutilização



Dificuldades do Reuso

- © Identificação, recuperação e modificação de artefatos

Ainda assim, devido ao potencial do reuso de software, vale a pena entender e conhecer como funciona, para aproveitar seus benefícios



- © Necessidade da criação de incentivos à reutilização



Técnicas de Reuso



Técnicas de Reuso

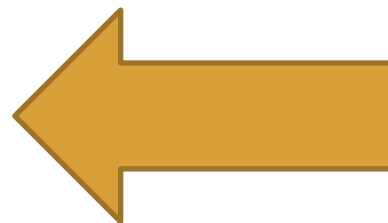
Algumas técnicas que propiciam o reuso de projeto e implementação em sistemas são:

- ◎ Padrões de Projeto (Design Patterns)
- ◎ Frameworks
- ◎ Linhas de Produto
- ◎ Bibliotecas de Software

Técnicas de Reuso

Algumas técnicas que propiciam o reuso de projeto e implementação em sistemas são:

- ◎ **Padrões de Projeto (Design Patterns)**
- ◎ Frameworks
- ◎ Linhas de Produto
- ◎ Bibliotecas de Software



A decorative background featuring various colored circles (green, yellow, orange, red, blue) and a dashed line forming a large circle around the text. A blue circle with white quotation marks is positioned above the text.

“

Cada padrão descreve um problema que ocorre repetidas vezes em nosso ambiente, e então descreve o núcleo da sua solução para aquele problema, de tal maneira que seja possível usar essa solução milhões de vezes sem nunca fazê-la da mesma forma duas vezes.

Christopher Alexander sobre padrões em arquitetura de construções



“

Os padrões de projeto são descrições de objetos que se comunicam e classes que são customizadas para resolver um problema de projeto genérico em um contexto específico.

Gamma, Helm, Vlissides & Johnson, sobre padrões de projeto em software

Padrões de Projeto

O que é?

- ⦿ Uma nova categoria de conhecimento
- ⦿ Conhecimento não é novo, mas falar sobre ele é
- ⦿ O objetivo é conhecer o que você já conhece

Como?

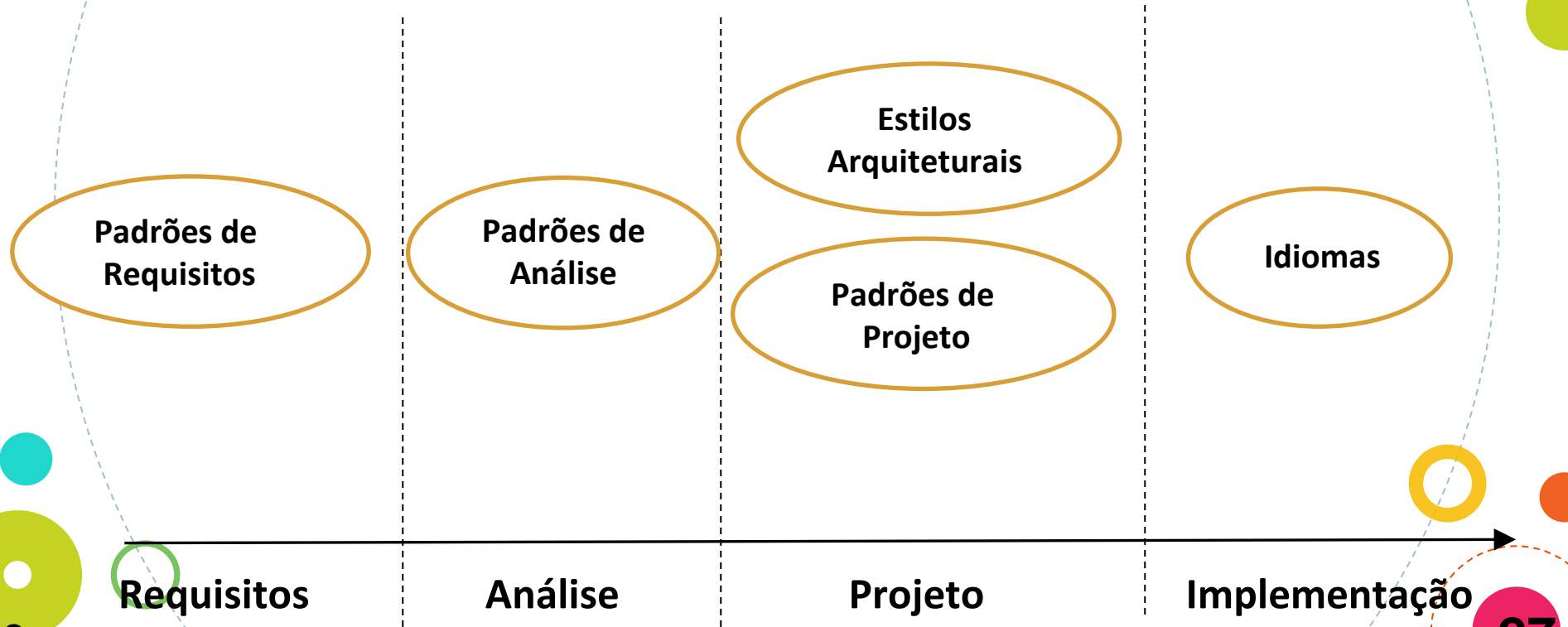
- ⦿ Partindo de problemas e soluções recorrentes em diferentes áreas do conhecimento

Padrões de Projeto

Por que padrões de software?

- ⦿ engenheiros de software não iniciam o seu projeto do nada
- ⦿ ao contrário, nós reutilizamos “ideias” que já vimos antes
- ⦿ as mesmas técnicas são utilizadas repetitivamente
- ⦿ a indústria de software necessita documentar o que nós fazemos

Classificação dos Padrões



Padrões de Requisitos

- ⦿ Documentam as necessidades do usuário e o comportamento genérico do sistema em um alto nível de abstração
- ⦿ Ações que os desenvolvedores de software podem tomar para melhorar os requisitos não-funcionais
- ⦿ Mostram os relacionamentos entre o usuário ou o operador e o sistema

Padrões de Requisitos - Exemplo

Fault-tolerant telecommunication patterns

- ◎ Visa a manutenção dos sistemas de comutação
- ◎ Medidas apropriadas para serem tomadas no estágio de desenvolvimento de requisitos
- ◎ Padrões relacionados a confiabilidade (mensagens do sistema e falhas do sistema)

“Five minutes of no escalation messages”

Padrões de Requisitos - Exemplo

IDENTIFICAÇÃO:

Nome do Padrão: Construindo as coisas certas.

Problema: Como você captura, comunica e valida requisitos de software, tal que você possa construir sistemas de sucesso que "fazem as coisas certas"?

CONTEXTO:

Causas:

- clientes freqüentemente não expressam adequadamente seus requisitos;
- desenvolvedores tem dificuldade de entender o que os clientes precisam;
- requisitos mudam, ou eles são incompletos, não bem entendidos ...

....

SOLUÇÃO:

5. Identificação: USAR MÉTODOS E ATIVIDADES DE DEFINIÇÃO DE REQUISITOS

Motivo: determinar qual o melhor uso e ligar, ou ordenar, as três atividades chaves — análise de domínio, especificação de requisitos e avaliação de requisitos — para produzir a especificação que melhor comunicará e modelará os requisitos.

Recomendação: isto é extremamente importante para incluir o uso de <protótipos> como parte chave da análise de requisitos.

Pré-condição: estabelecer e manter um acordo com o cliente: a análise de requisitos só inicia quando a investigação estiver completa e todos os envolvidos forem identificados.

Entradas: Requisitos.

Saídas: Especificação de requisitos validados; uma ou mais simulações de produtos.

Padrões de Requisitos - Exemplo

- **Nome do Padrão:** Um nome descritivo que permita a fácil comunicação do padrão.
- **Forças:** As razões para a existência do padrão.
- **Contexto:** A área de aplicabilidade do padrão.
- **Solução:** A descrição do padrão.
- **Padrões Relacionados:** Outros padrões que possam ser aplicados em conjunto com o padrão descrito ou que possam ajudar a compreendê-lo melhor.

Padrões de Requisitos - Exemplo

- **Nome do Padrão:** Cliente Pretende Comprar Produto
- **Forças:** Uma organização é procurada pelos seus clientes para fornecer bens ou serviços. Se essa procura não é satisfeita, existe o risco de o cliente escolher outra organização. Por vezes, a o produto está indisponível para venda imediata.
- **Contexto:** Um padrão para receber pedidos dos clientes, processar a encomenda e emitir a factura.
- **Solução:**

Partes do padrão:

Relação entre os objectos que participam no padrão:

- **Padrões Relacionados:** Cliente Cancela Pedido; Cliente Efectua Pagamento; Cliente Devolve Produto com Falhas.

Padrões de Análise

- ◎ Inicialmente apresentados como complementos aos padrões de projeto
- ◎ Um passo antes do projeto
- ◎ São grupos de conceitos úteis na modelagem de domínios de negócio
- ◎ Podem se aplicar a um único domínio ou a vários domínios
- ◎ Modelo de análise que focaliza nas estruturas conceituais
- ◎ Apoiam o reuso de ideias durante a fase de análise

Padrões de Análise - Exemplo

- ⊙ Existem padrões de análise em praticamente qualquer domínio
- ⊙ Fowler escreveu um excelente livro que cobre padrões classificados como segue:
 - ⊙ Padrões de Accountability (organização e responsabilidade)
 - ⊙ Padrões de Observações e Medições
 - ⊙ Padrões de Observações para a Finança Corporativa
 - ⊙ Padrões de Inventário e Contabilidade
 - ⊙ Padrões de Planejamento
 - ⊙ Padrões para o Comércio
 - ⊙ Padrões de Contratos de Derivativos

Padrões de Análise - Exemplo

Padrões de análise do Martin Fowler

- ◎ Domínio de conhecimento de software de negócios
- ◎ Party, quantity, subtype state machines, entre outros

Party

- ◎ **Problema:** pessoas e organizações têm responsabilidades semelhantes
- ◎ **Solução:** Crie um tipo party como um supertype de uma pessoa ou organização

Padrões de Projeto

- ◎ Estrutura repetida de elementos de projeto
- ◎ Um esquema para o refinamento de subsistemas ou de componentes de sistemas ou as relações entre eles.
- ◎ Padrões de projeto que incluem detalhes de código de baixo nível
- ◎ Aplicados a diferentes tipos de problemas

Padrões de Projeto

- ◎ Estrutura repetida de elementos de projeto
- ◎ Um esquema para o refinamento de subsistemas ou de componentes de sistemas ou as relações entre eles.
- ◎ Padrões de projeto que incluem detalhes de código de baixo nível
- ◎ Aplicados a diferentes tipos de problemas

Estilos Arquiteturais

Cada estilo descreve uma categoria de sistemas que engloba:

- ◎ Um conjunto de componentes (um banco de dados, módulos de interface) que realiza uma função requerida pelo sistema
- ◎ Um conjunto de conectores que fornecem “comunicação, coordenação e cooperação” entre os componentes

Estilos Arquiteturais - Exemplo

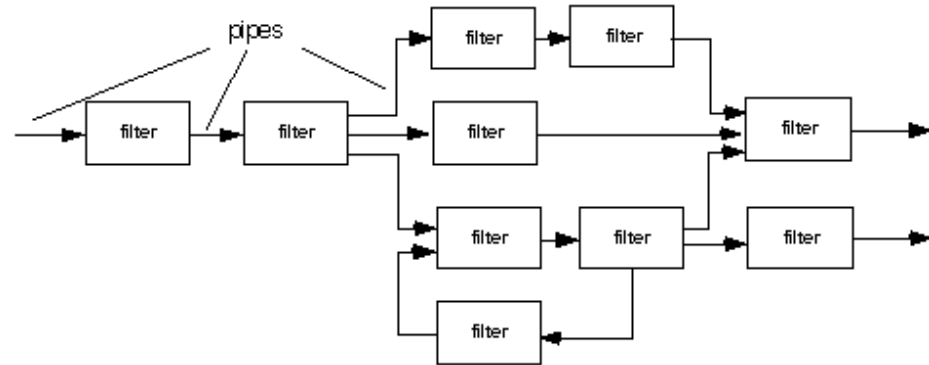
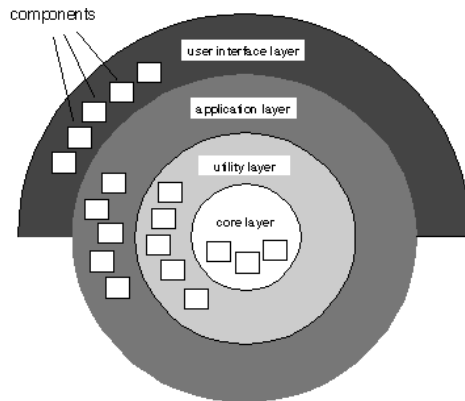
MVC

Pipes & Filters

Camadas

Repositório

Cliente Servidor



(a) pipes and filters



(b) batch sequential

Idiomas

- ◎ Relacionados com a implementação de características de projeto específicas
- ◎ Padrão de baixo nível específico para uma linguagem de programação
- ◎ Idiomas em C++
- ◎ C++ Programming Styles and Idioms, James Coplien, 1991

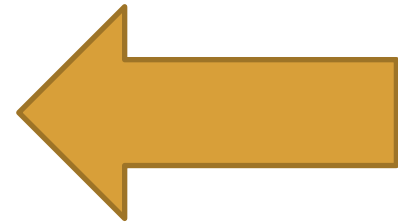
Idiomas - Exemplo

- ◎ Nome: Counted Body
- ◎ Contexto: A interface de uma classe é separada de sua implementação (respectivamente, classes handle e body)
- ◎ Problema: atribuição em C++ é definida recursivamente como membro-por-membro com cópia quando a recursão termina
- ◎ Solução: Um contador de referência é adicionado à classe body para facilitar o gerenciamento de memória
- ◎ Autor e data: James Coplien, 1994

Técnicas de Reuso

Algumas técnicas que propiciam o reuso de projeto e implementação em sistemas são:

- ◎ Padrões de Projeto (Design Patterns)
- ◎ **Frameworks**
- ◎ Linhas de Produto
- ◎ Bibliotecas de Software



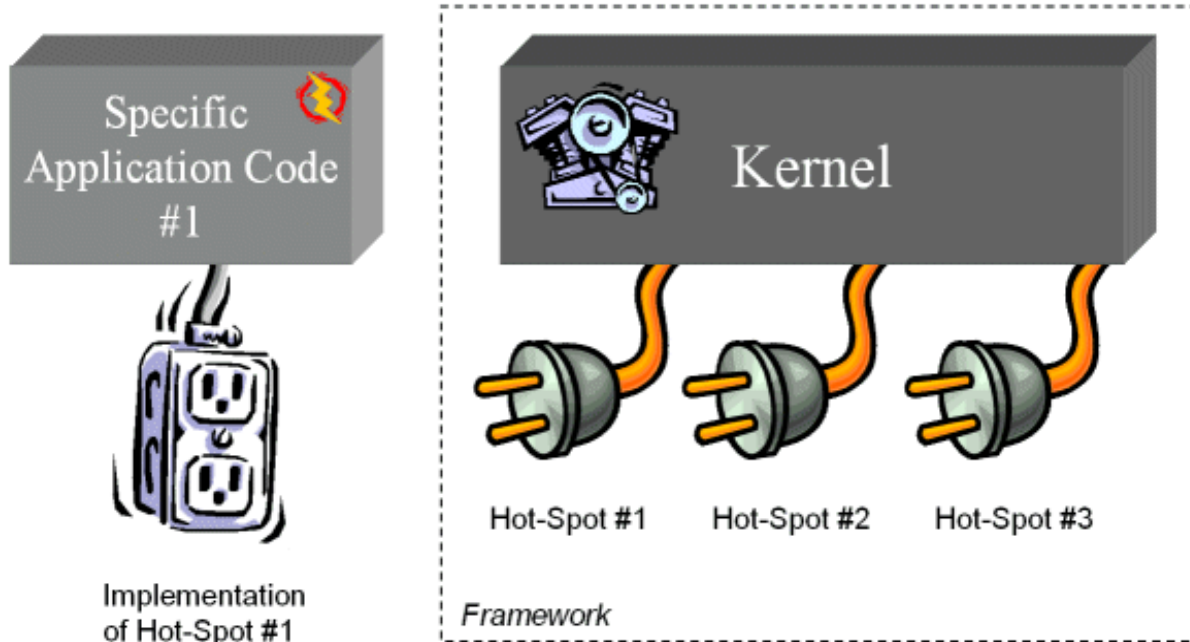
Frameworks

Framework de aplicação

- ◎ É uma estrutura genérica que deve ser estendida para criar uma aplicação específica
- ◎ Um conjunto integrado de artefatos de software (como classes, objetos e componentes) que colaboram para fornecer uma arquitetura reusável para uma família de aplicações relacionadas.
- ◎ São implementados como uma coleção de classes concretas e abstratas em uma linguagem OO. Exemplos: Hibernate, Struts
- ◎ Frameworks - entidades “relativamente” grandes que podem ser reutilizadas

Frameworks

Um framework separa o que é fixo (frozen-spots) do que é variável (hot-spots)



Frameworks

Exemplo: Padrão MVC

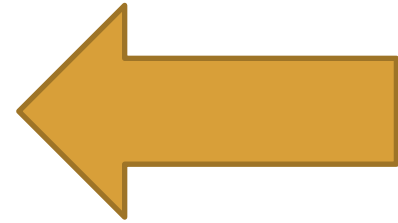
- O padrão MVC foi proposto na década de 60 para o projeto de Interface Humano-Computador.
- O Framework MVC permite representar dados de diferentes maneiras.
- O framework MVC inclui também os padrões GoF observador, estratégia e composição.



Técnicas de Reuso

Algumas técnicas que propiciam o reuso de projeto e implementação em sistemas são:

- ◎ Padrões de Projeto (Design Patterns)
- ◎ Frameworks
- ◎ **Linhas de Produto**
- ◎ Bibliotecas de Software



Linhas de Produto

- © É uma implementação de um conjunto de produtos de software que apresentam características comuns entre si.
- © Permite às organizações explorar semelhanças entre seus produtos, aumentando, assim, a reutilização de artefatos e, como consequência, tem-se uma diminuição dos custos e do tempo no desenvolvimento (Heymans e Trigaux, 2003).

A decorative background featuring various colored circles (green, yellow, orange, red, blue) and a dashed line forming a large circle around the text. A blue circle with white quotation marks is positioned above the text.

“

uma linha de produto de software é um conjunto de sistemas que usam software intensivamente, compartilhando um conjunto de características comuns e gerenciadas, que satisfazem as necessidades de um segmento particular de mercado ou missão, e que são desenvolvidos a partir de um conjunto comum de ativos principais e de uma forma preestabelecida

Clements e Northrop, 2001

Linhas de Produto

Um conjunto de sistemas de software que:

- ◎ Tem funcionalidades em comum
- ◎ São construídos de uma forma prescrita visando uma missão específica ou segmento de mercado.
- ◎ São desenvolvidos utilizando componentes e recursos (ativos) de uma base comum.
- ◎ Substancial economia de produção de software
- ◎ Aplicável em grupos de sistemas similares

Linhas de Produto

Benefícios:





- ◎ Ganhos de produtividade em larga escala
- ◎ Diminuição do tempo de entrega
- ◎ Melhoria da qualidade do produto e satisfação do usuário
- ◎ Maior eficiência no uso dos recursos humanos
- ◎ Maior presença no mercado
- ◎ Possibilidades de crescimento da empresa

Linhas de Produto

- ◎ Uma das abordagens para o desenvolvimento de linha de produto de software é o **Feature-Oriented Domain Analysis**
- ◎ Feature é todo aspecto visual proeminente ou distintivo para o usuário, qualidade ou característica de um sistema
- ◎ O Feature model tem como objetivo definir as features e suas dependências
- ◎ O Feature Model é descrito em forma de árvore e relacionamentos



Linhas de Produto

Feature Model - Elementos

Relacionamento	Tipo	Semântica	Notação
Relacionamento de Domínio	Mandatório	Se a <i>feature</i> pai é selecionada, o filho também devem ser selecionado	
	Opcional	Se a <i>feature</i> pai é selecionada, a funcionalidade filha pode ser selecionada	
	Alternativa	Se a <i>feature</i> pai é selecionada, exatamente uma <i>feature</i> filha deve ser selecionada	
	Ou	Se a <i>feature</i> pai é selecionada, pelos uma da <i>features</i> filha deve ser selecionada.	

Linhas de Produto

Feature Model - Elementos

Relacionamento	Tipo	Semântica	Notação
Dependência	Implicação	Se uma <i>feature</i> é selecionada, a <i>feature</i> implicada deve ser selecionada, ignorando a sua posição na árvore de funcionalidade	
	Exclusão	Indica que ambas as <i>feature</i> que não podem ser selecionadas na mesma configuração de produto e que são mutuamente exclusivas	

Linhas de Produto

Feature Model - Exemplo

– *Lista de Features*

- *Carro*
 - *Features mandatórias*
 - » *Motor*
 - *Features alternativas*
 - » *Transmissão manual*
 - » *Transmissão automática*
 - *Features opcionais*
 - » *Ar condicionado*

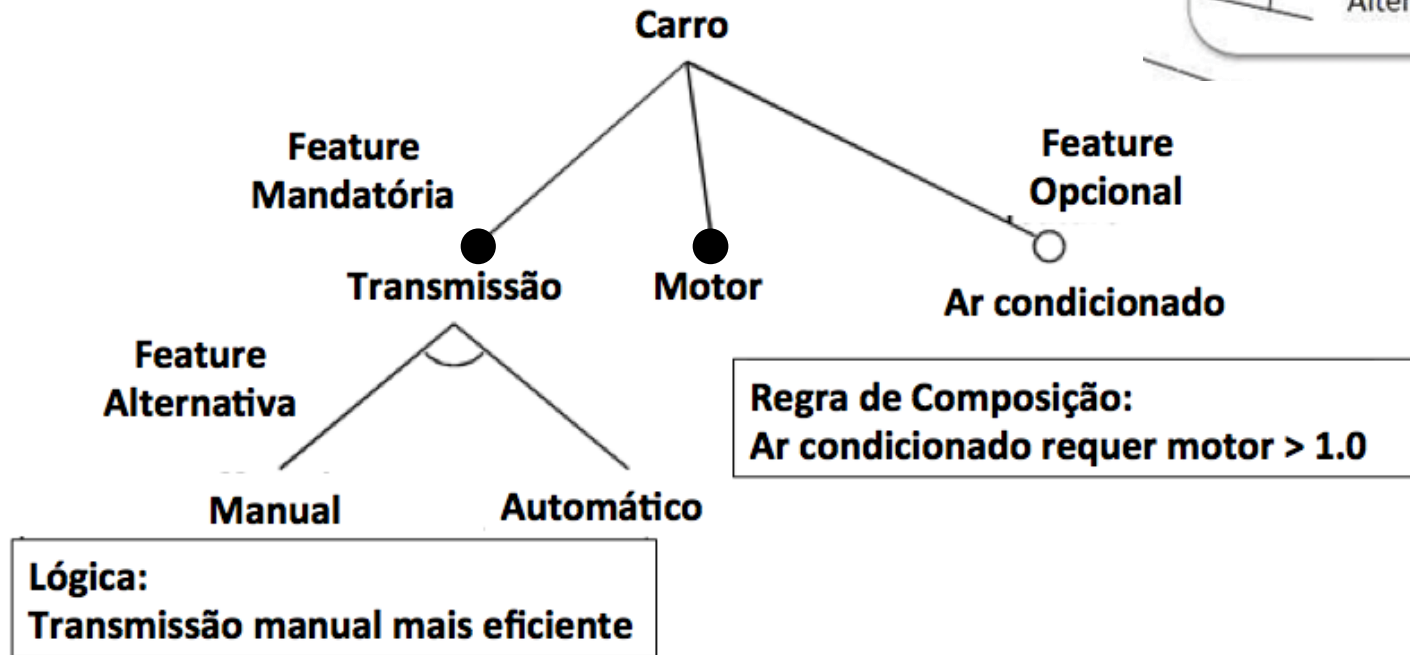
Linhas de Produto

Feature Model – Mapa de Produtos

	Modelo 1	Modelo 2	Modelo 3	Modelo 4
Transmissão Manual		X		X
Transmissão Automática	X		X	
Motor	X	X	X	X
Ar condicionado			X	X

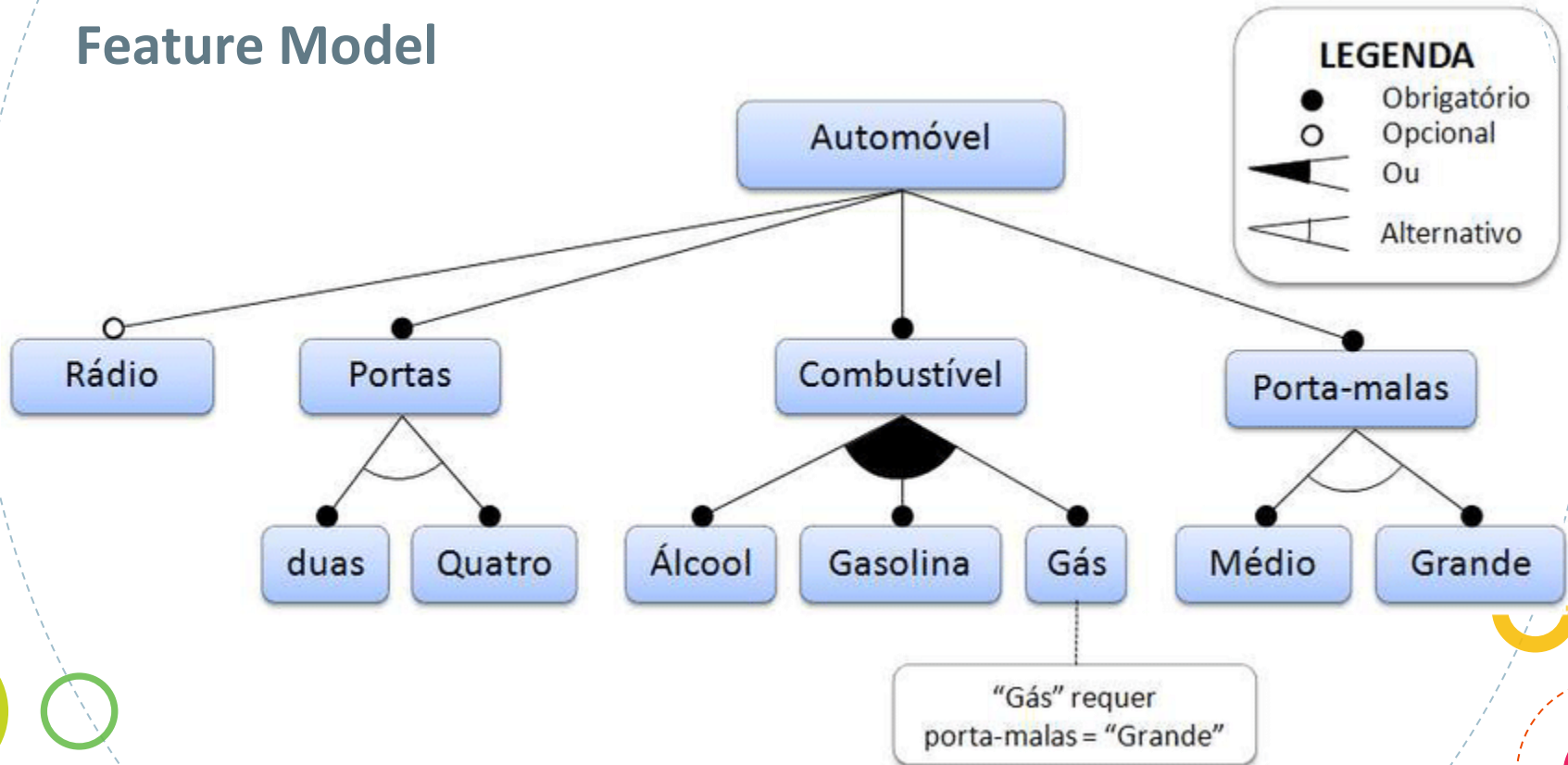
Linhas de Produto

Feature Model



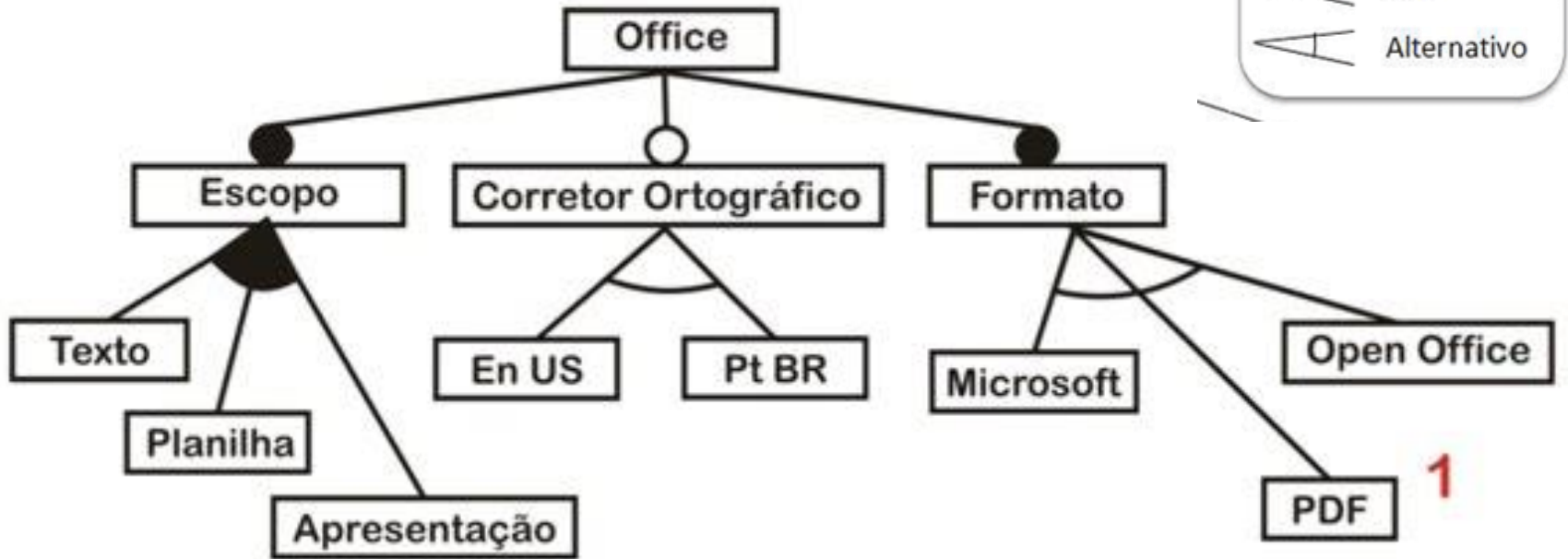
Linhas de Produto

Feature Model



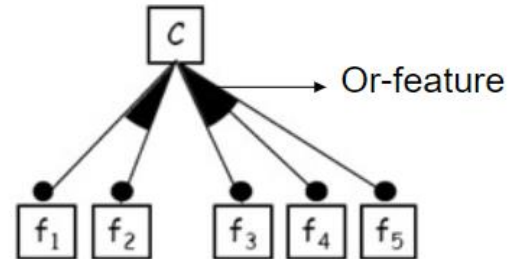
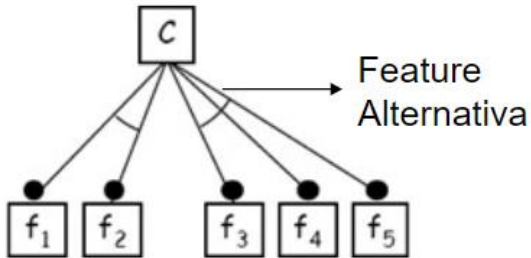
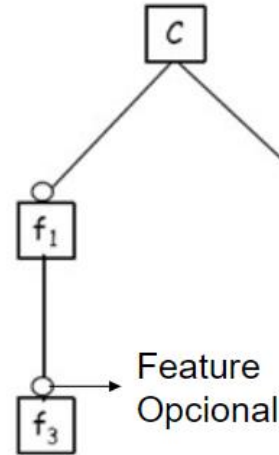
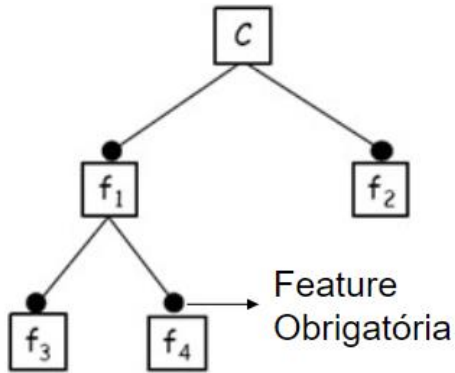
Linhas de Produto

Feature Model



Linhas de Produto

- Notação FODA



Linhas de Produto

- Ferramentas

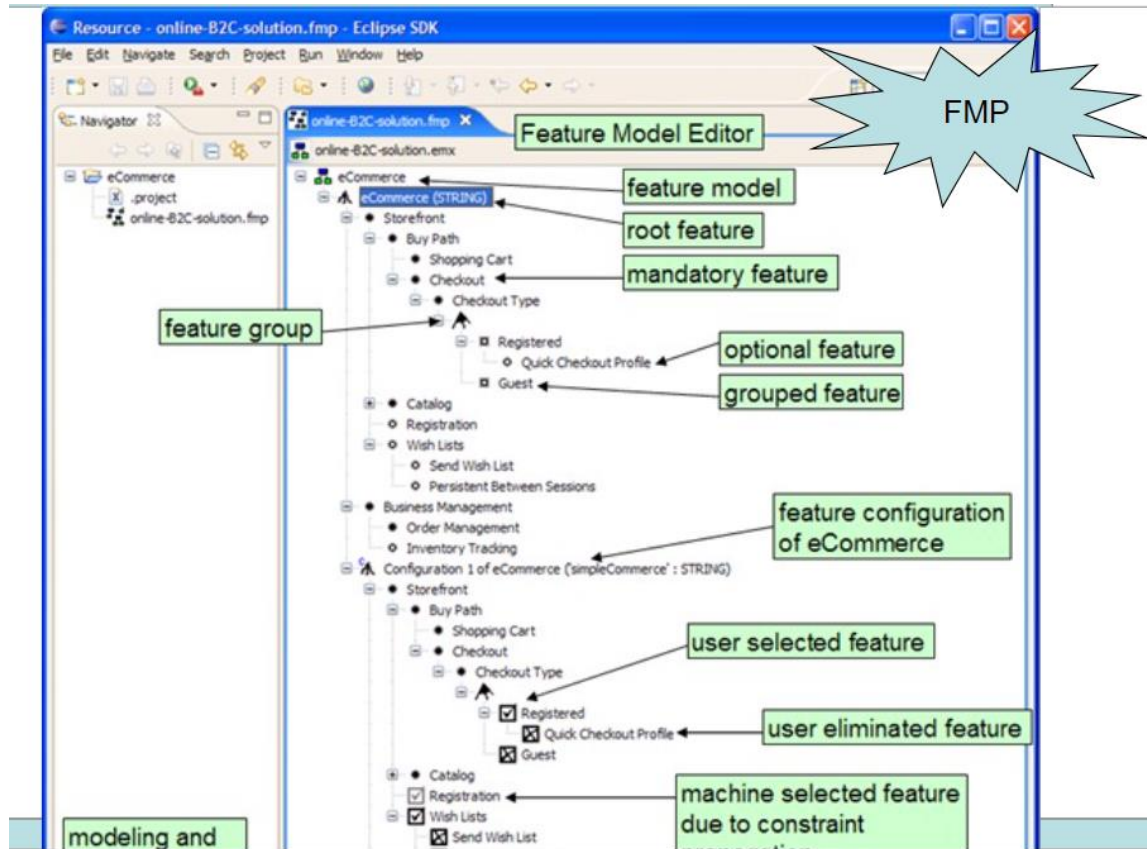
- Feature Modeling Plug-in (fmp)

- Eclipse plug-in
 - Permite edição e configuração de modelos de feature
 - Download
 - <http://gsd.uwaterloo.ca/projects/fmp-plugin/>

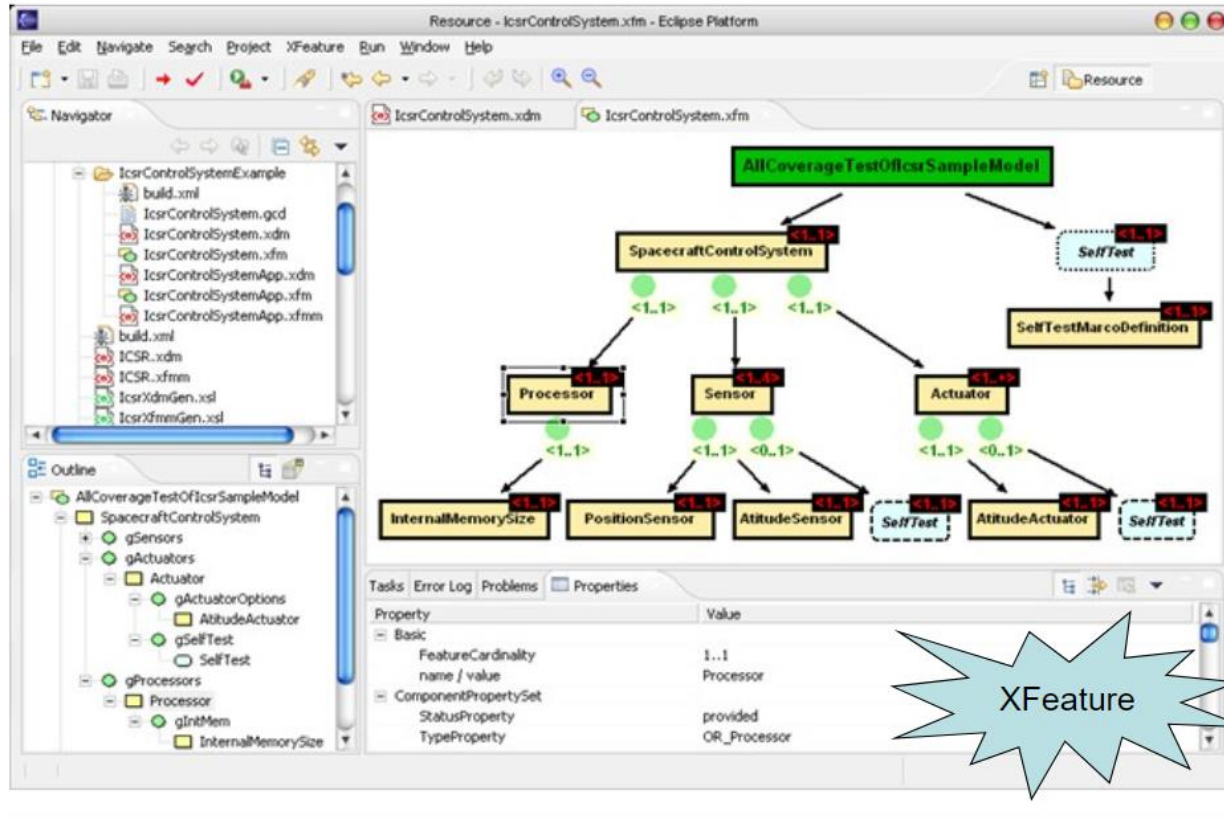
- XFeature

- Eclipse Plugin
 - Suporta a modelagem de famílias de produto e de aplicações instanciadas a partir delas
 - Permite que os usuários definam seu próprio meta-modelo
 - Download
 - <http://www.pnp-software.com/XFeature/>

Linhas de Produto



Linhas de Produto



Linhas de Produto

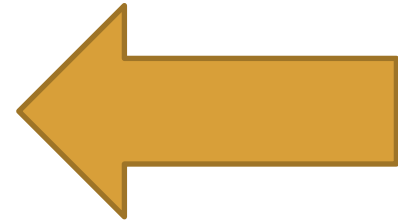
Casos de Sucesso

- **Nokia**
 - Passou a produzir de 25 a 30 modelos de celular por ano devido a abordagem de linha de produto de software
 - Antes eram produzidos 4 celulares
- **Motorola**
 - Observou uma melhoria de 400% em uma família de pages unidirecionais
- **HP**
 - Família de impressoras
 - Tempo para o mercado melhorou em 7 vezes
 - Produtividade aumentou 6 vezes

Técnicas de Reuso

Algumas técnicas que propiciam o reuso de projeto e implementação em sistemas são:

- ◎ Padrões de Projeto (Design Patterns)
- ◎ Frameworks
- ◎ Linhas de Produto
- ◎ **Bibliotecas de Software**



Bibliotecas de Software

Bibliotecas implementam serviços que podem ser usados por programas

- ⦿ É uma forma comum de reutilização
- ⦿ Disponibiliza funcionalidades comuns a diferentes tipos de sistemas

Exemplos:

- ⦿ Converter informação entre formatos conhecidos (e.g., string para inteiro)
- ⦿ Acesso a recursos, arquivos, BD, etc.
- ⦿ Tipos abstratos de dados: fila, pilha, lista

Bibliotecas de Software

Exemplo de uso de biblioteca em java:

```
import java.util.Vector;

public class Customer {

    String name;
    Vector phoneNumbers = new Vector();

    void removePhoneNumber(String c){
        phoneNumbers.removeElement(c);
    }

    void addPhoneNumber(String c){
        phoneNumbers.addElement(c);
    }

    ...
}
```

Técnicas de Reuso

Algumas técnicas que propiciam o reuso de projeto e im

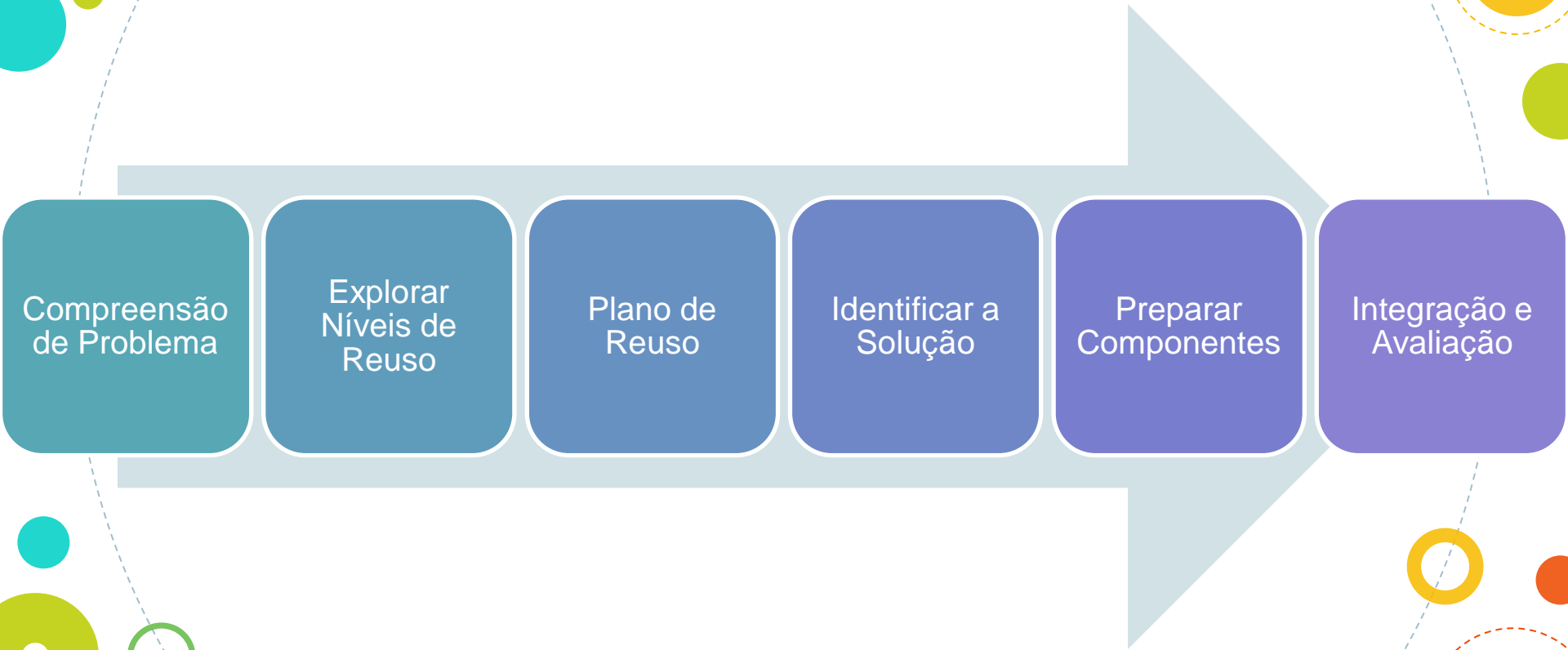
Ok
Mas como fazemos
reuso?



Processo



Processo



Vamos Reusar Bibliotecas de Software?



Exercício II

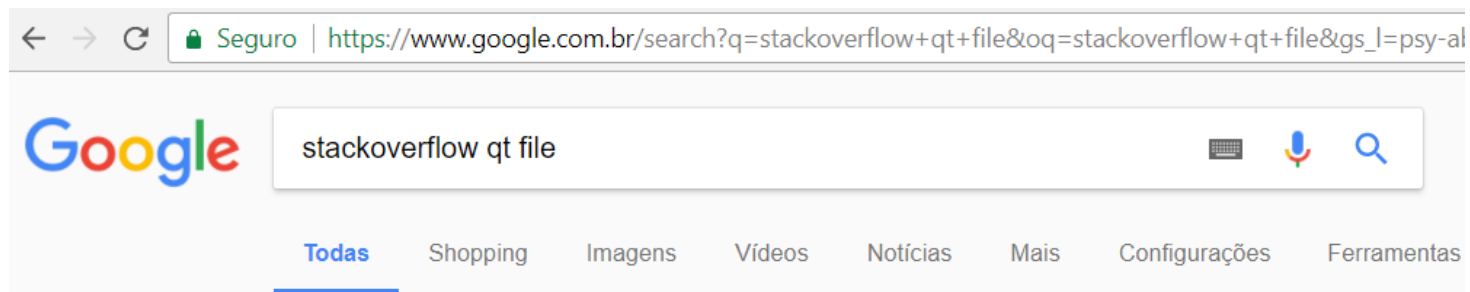
Considere o contexto descrito a seguir:

“Você foi contratado para criar um sistema que gere relatórios automaticamente ao processar arquivos individuais e ler os seus conteúdos.”

Que tipo de bibliotecas precisaríamos importar para aproveitar funções prontas que permitam manipular os dados mais rápido?

Exercício II

Stackoverflow - Este website apresenta perguntas e respostas em uma grande quantidade de tópicos de programação de computadores



Aproximadamente 334.000 resultados (0,34 segundos)

c++ - Creating/writing into a new file in Qt - Stack Overflow

<https://stackoverflow.com/.../creating-writing-into-a-new-file-in-q...> ▼ Traduzir esta página

6 de fev de 2011 - Are you sure you're in the right directory? Opening a **file** without a full path will open it in the current working directory. In most cases this is not ...

Exercício II

```
#ifndef DADOSRELATORIOCONTROLLER_H
#define DADOSRELATORIOCONTROLLER_H

#include <QString>
#include <QFile>
#include <QTextStream>
#include <QtGui>
|
class DadosRelatorioController
{
```

```
void DadosRelatorioController::atualizarDados(QString fileName)
{
    QFile inputFile(fileName);
    QString relatorioCompleto = "";

    if (inputFile.open(QIODevice::ReadOnly))
    {
        total_usuarios++;
        QTextStream in(&inputFile);
        in.setCodec("UTF-8");
        while (!in.atEnd())
        {
            QString stringAux = in.readLine();
            if (stringAux == "")
            {}
            else
            {
                relatorioCompleto.append(stringAux);
                relatorioCompleto.append("\n");
            }

            processarString(stringAux);
        }
        relatorioCompletoList.append(relatorioCompleto);
        inputFile.close();
    }

    /*qDebug() << "Mostrando Dados Após Relatorio";
    qDebug() << relatorioCompleto;
    qDebug() << "\n\n";*/
}
```

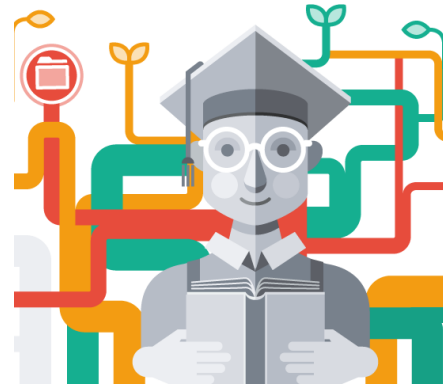
```
void DadosRelatorioControler::processarString(QString stringAux)
{
    //Dados Demograficos
    if (stringAux.contains("Idade/"))
    {
        idade_media = idade_media + stringAux.remove("Idade/").toInt();
        if (stringAux.remove("Idade/").toInt() < idade_min)
            idade_min = stringAux.remove("Idade/").toInt();
        if (stringAux.remove("Idade/").toInt() > idade_max)
            idade_max = stringAux.remove("Idade/").toInt();
    }

    if (stringAux == "Sexo/Masculino")
        sexo_masc++;
    if (stringAux == "Sexo/Feminino")
        sexo_fem++;
    if (stringAux == "Experiencia/Nenhuma - Não uso")
        exp_zero++;
    if (stringAux == "Experiencia/Baixa - Uso com pouca frequência")
        exp_baixa++;
    if (stringAux == "Experiencia/Média - Uso de vez em quando")
        exp_med++;
    if (stringAux == "Experiencia/Alta - Uso quase todo dia")
        exp_alta++;
}
```

Exercício II

Considere o contexto descrito a seguir:

E então?
O que aprendemos?



Resumo da Aula de Hoje...

- ◎ O objetivo do reuso de software é o aumento da produtividade e redução no esforço de desenvolvimento de novos produtos.
- ◎ Fatores como desconhecimento de técnicas de reuso, infraestrutura para reuso e fatores humanos são problemas adicionais que tornam complexa a realização de reuso.

Aplique reuso e torne-se mais competitivo no mercado!





Atividades para casa

Exercício 1

- *Baseado na seguinte lista de features, criar o modelo de features para um SW de supermercado:*
 - *Pagamento (OU)*
 - *Dinheiro*
 - *Cartão*
 - *Cheque*
 - *Controle de Estoque (Obrigatória)*
 - *Sistema Operacional (Alternativas)*
 - *Windows*
 - *Linux*
 - *Mac OS*
 - *Relatório Gerencial (Opcional)*

Exercício 2

- *Baseado no modelo de features criado para o SW de supermercado, criar o mapa de produtos para 5 produtos*

Exercício 3

- Criar para uma linha de produto de software para telefone celulares que contenham pelo menos os relacionamento mandatório, opcional, alternativo e de implicação.
- Artefatos a serem criados:
 - Requisitos: **Lista de features**
 - Mapa de produtos
 - Modelo de features

Exercício 4

- **Pesquisar e fazer um resumo ou mapa mental sobre:**
 - **Proposta MDD – Desenvolvimento Dirigido por Modelos**
 - Tópicos sugeridos: Abordagem; Processo MDD; Vantagens; Potenciais problemas
 - **Reuso de Produtos COTS (software pronto)**
 - Tópicos sugeridos: Conceitos; Vantagens; Desvantagens; Tipos

Referências

- ◎ Alexander, C. (1977). A pattern language: towns, buildings, construction. Oxford University Press.
- ◎ Clements, P., & Northrop, L. (2001). Software product lines: Patterns and practice. Boston, MA, EUA: Addison Wesley Longman Publishing Co.
- ◎ Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1993, July). Design patterns: Abstraction and reuse of object-oriented design. In European Conference on Object-Oriented Programming (pp. 406-431). Springer, Berlin, Heidelberg.
- ◎ Neil, T. (2014). Mobile design pattern gallery: UI patterns for smartphone apps. " O'Reilly Media, Inc.".
- ◎ Peters, J. F., & Pedrycz, W. (2001). Engenharia de software: teoria e prática. Rio de Janeiro: Campus, 681(519.683), 2.

Referências

- © Pressman, R. S. (2006) Engenharia de software. 6ª. ed. São Paulo: McGraw-Hill, xxxi, 720p.
- © Reutilização de Software - Engenharia de Software Magazine 39: Disponível em: <http://www.devmedia.com.br/reutilizacao-de-software-revista-engenharia-de-software-magazine-39/21956>
- © Sommerville, I. (2007) Engenharia de Software-8ª Edição. Ed Person Education.
- © Trigaux, J. C., & Heymans, P. (2003). Modelling variability requirements in software product lines: a comparative survey. EPH3310300R0462/215315, FUNDP–Equipe LIEL, Namur.
- © <https://slideplayer.com.br/slide/5025146/>

Obrigada!



Perguntas?

E-mail: jacilane.rabelo@ufc.br