



Apresentando a orientação a objetos 1

- ✓ Definição de programação orientada a objetos;
- ✓ Comparação entre modelos orientados a objeto e modelos estruturados;
- ✓ Definição de objetos;
- ✓ Conceito de sistema orientado a objetos;
- ✓ Vantagens da orientação a objetos.

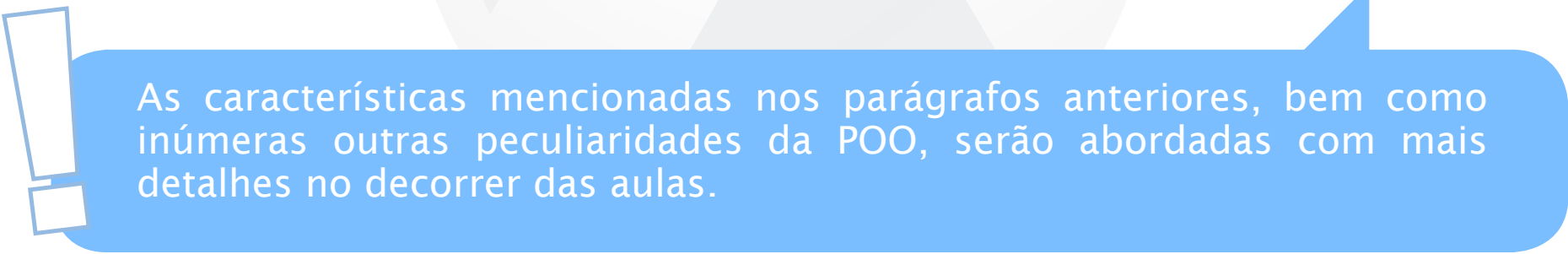
1.1. Introdução

Originada em meados da década de 1960 – mas utilizada em larga escala apenas a partir do início dos anos 1990 –, a linguagem orientada a objetos (POO) é um modelo de programação que emprega objetos no desenvolvimento de aplicações. Ela é baseada em vários conceitos, como modularidade, herança, encapsulamento e polimorfismo, os quais são amplamente aplicados na etapa de desenvolvimento.

Em comparação à visão tradicional de programação, em que o programa pode ser visto como uma lista de instruções para o computador, na POO, cada um dos objetos relaciona-se com os demais. Essa habilidade é demonstrada em diferentes características, como:

- Processamento de dados;
- Recebimento de mensagens de outros objetos;
- Envio de mensagens para outros objetos da aplicação.

Uma das vantagens de utilizar a programação orientada a objetos é que ela permite alterar ou substituir partes de um sistema sem que haja riscos de ocorrência de erros. Além disso, ela oferece uma visão mais natural, uma forte arquitetura e um alto grau de reusabilidade do código.



As características mencionadas nos parágrafos anteriores, bem como inúmeras outras peculiaridades da POO, serão abordadas com mais detalhes no decorrer das aulas.

1.1.1. Um pouco de história

Resgatando um pouco a respeito da origem da POO, especialmente os fatores que conduziram ao seu surgimento na década de 1960, constata-se um período em que o software e o hardware tornaram-se bastante complexos.

A qualidade do software deveria ser mantida a qualquer custo – isso era o que almejavam os profissionais da época. Justamente nesse contexto, em que se chegou a discutir até mesmo a ideia de uma crise relacionada ao software, surgiu a POO. Ela pretendia combater o problema, principalmente por meio da modularidade em software, ou seja, unidades separadas de lógica de programação.

A primeira linguagem de programação que introduziu os conceitos que fundamentam a POO foi a linguagem **Simula**. Esses conceitos, que incluem objetos, classes, subclasses, métodos virtuais, corrotinas, garbage collection e simulação de eventos discretos, foram introduzidos como um superconjunto da linguagem Algol (ALGOritmic Language).

A primeira linguagem de programação a ser chamada de orientada a objetos foi a linguagem **Smalltalk**.

1.2. Modelos orientados a objeto X modelos estruturados

A orientação a objetos e a programação estruturada são modelos diferentes de desenvolvimento de sistemas. O conceito de programação estruturada diz respeito a uma estratégia baseada na ideia de que um sistema é dividido em duas partes principais: os dados e as funcionalidades.

Em projetos estruturados, geralmente os dados são representados por um modelo de dados que define os registros estruturados para serem armazenados nas bases de dados. As funcionalidades são definidas por um modelo de processamento de fluxo de dados no qual é determinado como os dados devem entrar e sair dos processos e como devem ser armazenados no sistema.

Portanto, nas aplicações que utilizam uma abordagem estruturada, os dados ficam separados das funcionalidades do sistema. Já em um sistema de software que se baseia no conceito de orientação a objetos, os dados ficam armazenados em cada objeto existente. Em termos simples, objeto é algo capaz de ser apresentado e que possui algum significado ou sentido.

Em vez de os sistemas orientados a objetos serem definidos estruturalmente como duas partes separadas, as funcionalidades são nada mais que conjuntos de objetos interativos. Esses objetos geralmente possuem ações que interferem no comportamento de um sistema e informações que armazenam dados. Em termos de modelagem, os objetos são representados por classes.

1.3. Objetos

A orientação a objetos visa representar, de maneira idêntica, as situações do mundo real nos sistemas computacionais. Para ela, os sistemas operacionais não devem ser vistos como uma coleção estruturada de processos, mas sim como uma coleção de objetos que interagem uns com os outros organizadamente, assim como ocorre no mundo real.

Introdução à Programação Orientada a Objeto (online)

10

Em uma concepção abrangente, um objeto pode ser entendido como sendo um elemento físico, por exemplo, uma pedra, uma cadeira ou um animal, ou como um elemento abstrato, como uma conta bancária.



ILUSTRAÇÃO: EDUARDO JOSÉ DE SOUZA ENGELMANN

Dentro do ambiente de TI, os objetos computacionais são objetos que estão localizados dentro de sistemas de computador. A intenção ao especificar esses objetos é retratar as mesmas características dos objetos do mundo real e programar comportamentos que se aproximem dos encontrados na realidade. Como o comportamento desses objetos já está programado, um programador não precisa entender o funcionamento interno desse comportamento para interagir com ele. Para isso, basta ativar os comportamentos programados.

Um exemplo comum desse processo são os aplicativos para desenvolvimento de softwares que utilizam recursos de arrastar e soltar. O programador pode criar uma tela arrastando e soltando botões, caixas de textos, listas, conexões com banco de dados para uma tela, sem se preocupar com a forma como esses objetos funcionam internamente.

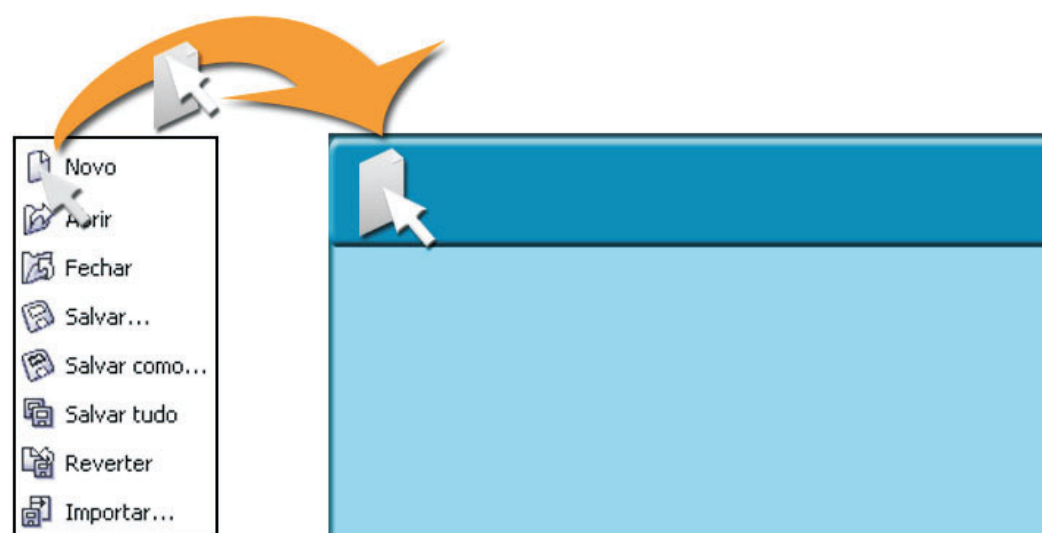


ILUSTRAÇÃO: EDUARDO JOSÉ DE SOUZA ENGELMANN

Ao arrastar um botão, o programador espera que o software, ao ser executado, desenhe aquele botão, e que este, ao ser pressionado, funcione como um botão do mundo real, afundando dentro do painel no qual está contido e executando alguma ação. O botão é um objeto e o fato dele responder a um clique é um evento. Esse tipo de programação em que objetos são desenhados por um aplicativo e o programador escreve as ações a serem executadas ficou conhecido como **Programação Orientada a Eventos**.

1.3.1. Objetos computacionais

A representação dos objetos computacionais no computador retrata apenas uma imagem dos objetos do mundo real, ou seja, são modelos abstratos dos objetos reais. No entanto, o comportamento desses modelos pode ser igual ao da realidade. Para que ocorra a interação dentro de um sistema de computador, os objetos computacionais devem saber como reagir diante de uma ação.

Conforme mencionado anteriormente, o comportamento desses objetos já está previamente programado, portanto, eles possuem a informação necessária para saber como reagir. Isso significa que o seu comportamento não depende do sistema.

Para facilitar a compreensão acerca da relação entre os objetos computacionais e o comportamento associado a eles, consideremos, como exemplo, quatro tipos de objetos existentes no ambiente computacional:

- **Objetos visuais**

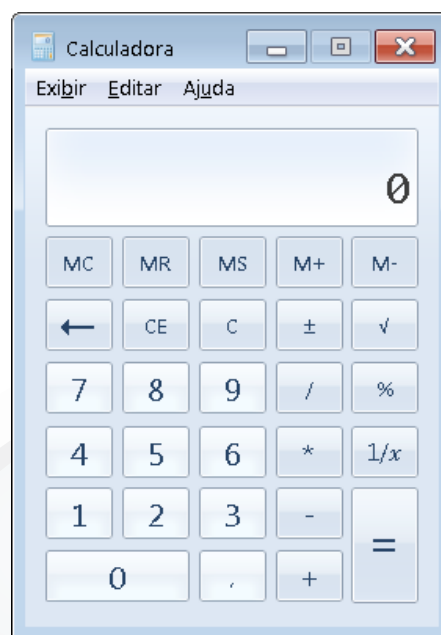
Interagem diretamente com o usuário, permitindo que ele aperte botões, mova peças e selecione itens dentro de caixas. Neste caso, a interação do usuário com esses objetos ocorre em um monitor, com a utilização de um mouse ou teclado. Mesmo assim, existe um nível de abstração e interação muito bom em relação ao mundo real.

Em geral, esse tipo de objeto é manuseado por ambientes de programação visual. Utilizar os objetos desses ambientes é uma tarefa simples: basta selecioná-los em uma palheta e, em seguida, arrastá-los até a janela da aplicação.

Um fator importante e que exige a nossa máxima atenção é a maneira como utilizaremos esses objetos, pois cada objeto já possui o conhecimento de como deve funcionar, sem que o programador escreva um código.

Alguns exemplos de objetos visuais são: Menus, Caixas de Texto, Botões e Listas.

A calculadora do Windows utiliza diversos objetos visuais:



É possível identificar nessa imagem da calculadora: um menu, uma caixa de texto e diversos botões.

- **Objetos de domínio de trabalho**

São objetos que estão envolvidos ou são criados durante o desenvolvimento de um sistema de computador. Além disso, eles se relacionam diretamente como trabalho desenvolvido e, muitas vezes, não são percebidos pelos olhos do usuário final.

Outra característica dos objetos de domínio de trabalho é que eles estão envolvidos diretamente com o sistema de informações.

Por exemplo, um software controlador de estoque pode conter alguns objetos internos, como Cliente, Produto, Fornecedor, Venda ou Compra.

- **Objetos com tarefa relacionada**

São arquivos que também possuem comportamentos inclusos, como objetos do tipo documentos e arquivos multimídia. Apesar de apresentarem esses comportamentos, em muitos casos é necessário ter softwares aplicativos para abri-los.



ILUSTRAÇÃO: EDUARDO JOSÉ DE SOUZA ENGELMANN

Por exemplo, um software de compra de ingressos para o cinema pode ter uma funcionalidade que permite ao usuário ver o trailer de um filme. Se esse trailer for um arquivo do tipo **MOV**, um aplicativo chamado **QuickTime**, da Apple, deverá estar instalado no computador, além do aplicativo que é responsável pela exibição correta de imagem e som.

- **Objetos multimídia**

A maioria das pessoas já está familiarizada com os objetos multimídia, os quais podem conter som, imagem, animação ou vídeo. Esses objetos sabem de que forma devem se comportar diante de uma situação.

Por representarem os objetos reais de forma bastante idêntica, os objetos multimídia são de fácil utilização.



ILUSTRAÇÃO: EDUARDO JOSÉ DE SOUZA ENGELMANN

1.4. Concepção de um sistema orientado a objeto

O conceito de orientação a objetos consiste em criar um sistema computacional como um sistema orgânico, que seja formado por objetos que interajam uns com os outros.

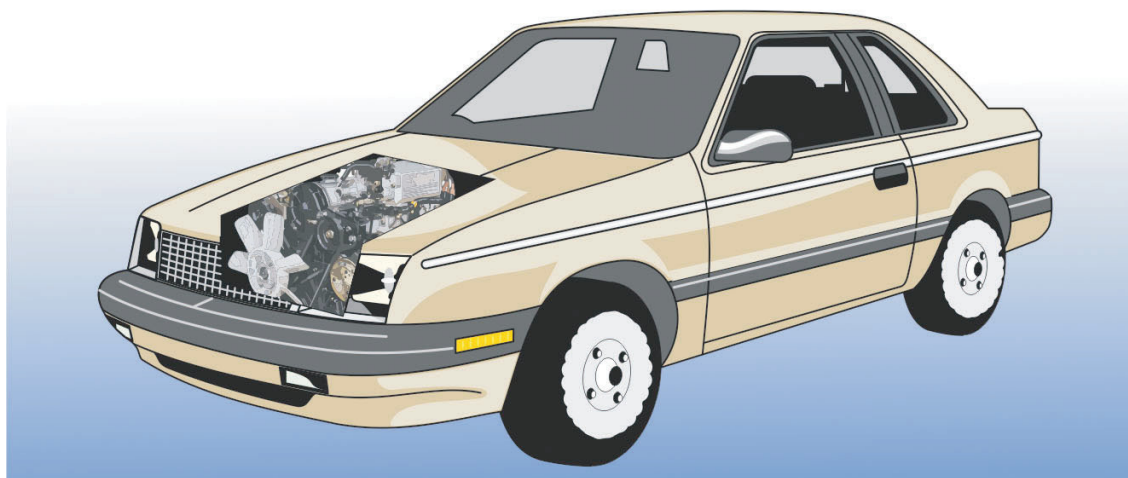


ILUSTRAÇÃO: EDUARDO JOSÉ DE SOUZA ENGELMANN

Esse conceito pode ser empregado na análise de sistemas e na programação, fazendo com que uma das principais vantagens da orientação a objetos seja a possibilidade de utilizar a mesma técnica, tanto para a definição lógica do sistema quanto para a sua implementação.

1.4.1. Análise

A análise orientada a objetos consiste em definir os objetos que pertencem a um sistema e a maneira como eles se comportam. O processo dessa análise consiste em identificar os seguintes elementos:

- Os objetos que existem dentro do ambiente que desejamos automatizar;
- Os atributos desses objetos, ou seja, que tipos de informação esses objetos devem conter;
- As ações que esses objetos podem executar.

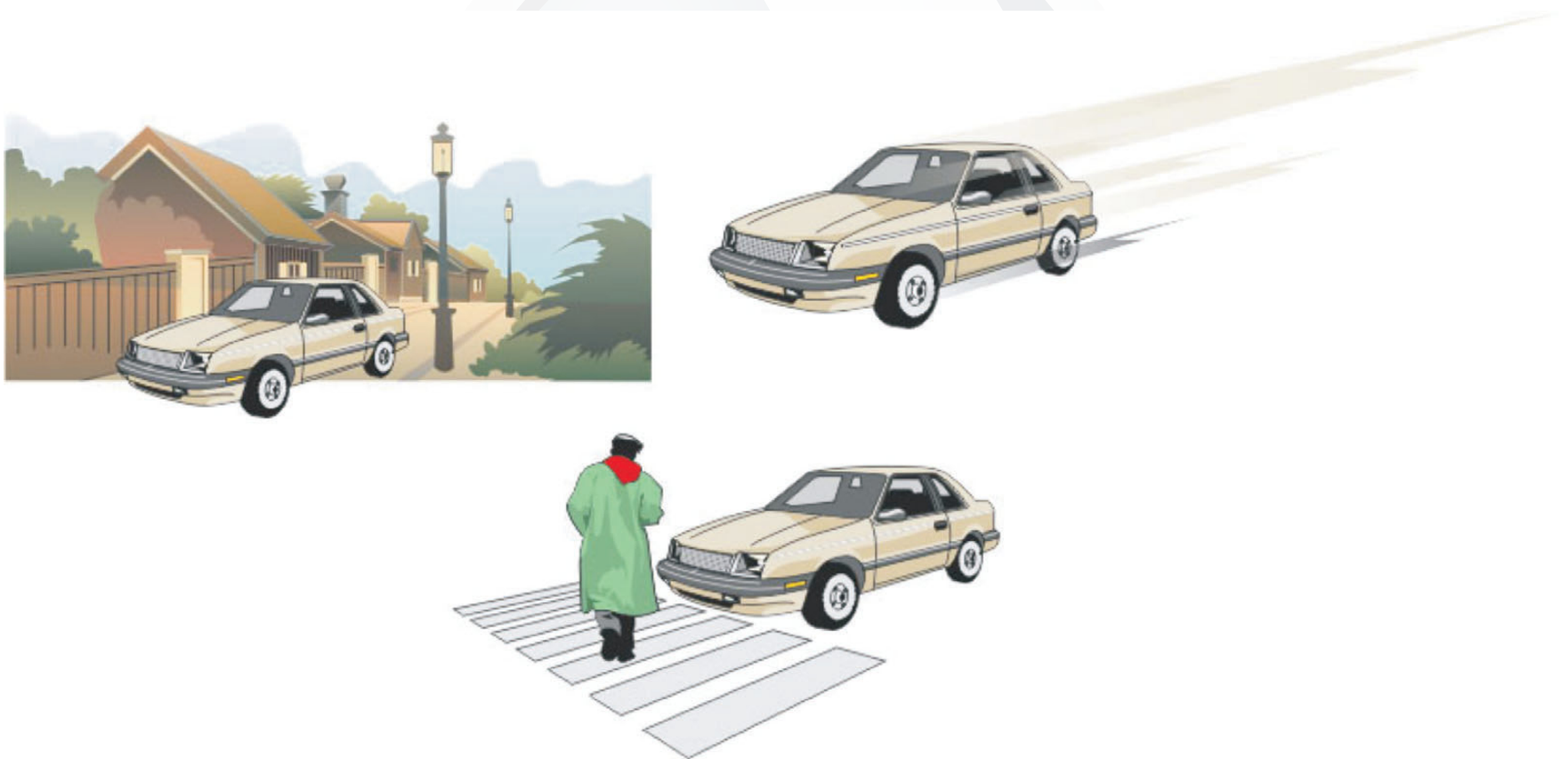


ILUSTRAÇÃO: EDUARDO JOSÉ DE SOUZA ENGELMANN

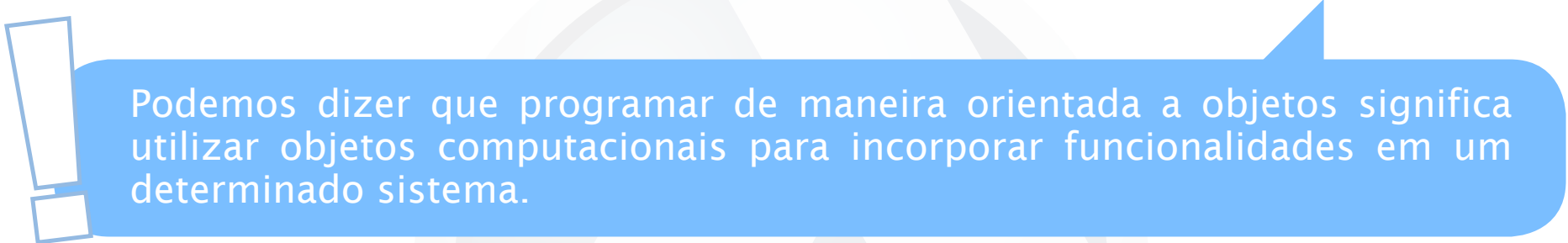
Por meio da análise orientada a objetos, um analista pode compartilhar o conhecimento que ele tem em relação ao funcionamento do sistema com um usuário comum.

1.4.2. Programação

Consiste em utilizar estruturas de dados que façam uma simulação do comportamento dos objetos. Dessa forma, a programação é realizada com mais facilidade com a utilização de uma única técnica tanto para a análise quanto para a programação.

Na programação orientada a objetos, os objetos identificados pelo analista em seu diagrama podem ser implementados exatamente da forma que foram projetados. Por outro lado, na análise estruturada é necessário traduzir os diagramas da análise para estruturas de dados e de programação.

Depois de fazer uma análise orientada a objetos, é possível implementá-la em uma linguagem tradicional e, também, implementar sistemas analisados com outras técnicas, utilizando linguagens orientadas a objetos.



Podemos dizer que programar de maneira orientada a objetos significa utilizar objetos computacionais para incorporar funcionalidades em um determinado sistema.

1.5. Vantagens

A seguir veremos algumas vantagens da utilização da técnica de orientação a objetos:

- **Organização**

A partir do momento que os desenvolvedores de sistemas começam a utilizar a técnica da orientação a objetos, eles passam a usufruir de diversas vantagens, dentre as quais a principal é a possibilidade de reunir em uma mesma estrutura os dados e os processos que são executados sobre esses dados. Consequentemente, há um aumento no nível de organização e simplicidade do programa.

- **Produtividade**

Outra vantagem é o ganho de produtividade, pois, quando um desenvolvedor cria um objeto para realizar uma tarefa difícil ou altera a maneira como esse objeto a executa internamente, os demais desenvolvedores que venham a interagir com esse objeto precisarão apenas acessá-lo para executar as mesmas tarefas, sem a necessidade de saber de que forma ocorre a execução e sem precisar alterar seus programas para continuar tendo acesso ao novo comportamento.

- **Redução do custo**

Além das vantagens mencionadas anteriormente, também há a redução no custo de desenvolvimento e no risco de ocorrência de erros. Isso ocorre porque, depois de criar uma estrutura eficaz de objetos, será possível incluir módulos adicionais no sistema, que reutilizem funcionalidades já desenvolvidas, ou seja, utilizem o mesmo código em sistemas diferentes. Consequentemente, não há necessidade de reprogramação.

- **Reaproveitamento**

Mais uma vantagem encontrada é a possibilidade de transformar objetos semelhantes, com a finalidade de aproveitá-los em novos sistemas. Isso significa que podemos fazer pequenas alterações em objetos que possuem características parecidas com as especificações necessárias de um novo objeto, a fim de que esse novo objeto resultante da alteração seja utilizado em um novo sistema.

- **Facilidade de manutenção**

Quando é preciso realizar alguma alteração em um sistema, a flexibilidade oferecida pela técnica de orientação a objetos permite que o desenvolvedor adapte, exclua ou inclua novos objetos no sistema rapidamente, sem comprometer o funcionamento do mesmo.

- **Trabalho em equipe**

Usando orientação a objetos, é fácil dividir tarefas entre diversas equipes de programação. Uma parte da equipe pode trabalhar no código relacionado aos bancos de dados e outra parte pode trabalhar na interação com o usuário. Esse tipo de programação é conhecido como **Programação em Camadas**.

Enfim, como a programação orientada a objetos lida com objetos pré-prontos, ela não apenas simplifica a criação de novos sistemas como tem a finalidade de maximizar a reutilização dos objetos.

