Capítulo 11

Educação Continuada

Not everything that counts can be counted, and not everything that can be counted counts.

Albert Einstein

OBJETIVOS

- Fazer uma breve retrospectiva da programação orientada a objetos.
- Discutir o contexto atual do uso da programação orientada a objetos combinada com o uso da linguagem C++.
- Apresentar um conjunto de aplicações que você pode estudar para aprofundar o conteúdo apresentado neste texto.
- Indicar um conjunto de informações que você pode consultar, além de servir de base de leitura complementar este material.

Os capítulos anteriores apresentaram a linguagem C++ e como ela pode ser empregada para elaborar programas orientados a objetos. Um conjunto de características do paradigma de programação orientada a objetos foi apresentado com exemplos para facilitar o entendimento e ilustrar o uso. Essa apresentação procurou balancear características da programação orientada a objetos e da linguagem C++. Este último capítulo tem por objetivo fazer uma breve análise do conteúdo deste livro e de assuntos correlatos a fim de motivá-lo a continuar aprimorando seus conhecimentos.

11.1. RETROSPECTO

Um aspecto de grande importância no desenvolvimento de programas grandes é sua organização, a qual é representada como uma estrutura organizacional do código que consiste em componentes computacionais e em conectores que permitem a

interação entre esses componentes. Essa percepção corresponde à visão arquitetural do software em um projeto de sistema.

É importante considerar que uma questão antecedente ao desenvolvimento de um programa para a solução de um problema diz respeito à natureza do software, bem como ao seu crescimento em termos de tamanho e complexidade.

O desenvolvimento de técnicas de abstração tem sido uma das principais fontes de avanço em termos de desenvolvimento e programação de sistemas de software. A década de 1980 foi notoriamente marcada pelo surgimento do paradigma de orientação a objetos.

Observe que uma das principais características do ser humano quando se depara com um problema é buscar uma solução baseada em soluções existentes de problemas similares. Entretanto, quando as soluções existentes não são suficientes para o problema, busca-se estender essas soluções a fim de solucionar um novo problema. Adicionalmente, o ser humano tem feito uso da abstração como forma de lidar com situações de complexidade.

Assim, à medida que os sistemas crescem, também cresce sua complexidade e torna-se mais difícil satisfazer a um número cada vez maior de requisitos, muitas vezes conflitantes. Atualmente, o paradigma de orientação a objetos tem se mostrado como o mais adequado, comparativamente aos demais, para ser empregado no desenvolvimento de sistemas de software complexos e de grande porte.

Nesse sentido, a programação orientada a objetos (POO) tem papel fundamental. A programação orientada a objetos foi desenvolvida devido às limitações encontradas nas abordagens anteriores de programação. Para entender o que POO faz, é preciso entender quais são essas limitações e como elas surgiram nas linguagens de programação tradicionais. Pascal, C, Basic e Fortran são linguagens *procedimentais*, isto é, cada declaração na linguagem procedimental informa que o computador deve realizar alguma tarefa, como, por exemplo, ler um dado de entrada, adicionar uma constante, dividir um número por outro e exibir o resultado. Um programa em uma linguagem procedimental é visto como uma *lista de instruções* organizadas em procedimentos (funções).

Note ainda que, para programas pequenos, nenhum princípio organizacional (ou paradigma) é necessário. O programador simplesmente cria uma lista de instruções, e o computador as executa. Entretanto, à medida que os programas se tornam maiores e com grande número de funcionalidades, fica muito mais difícil tratar isso em uma única lista de instruções. Poucos programadores podem compreender um

programa que possua mais de poucas centenas de instruções, a menos que o programa seja *quebrado* ou dividido em unidades menores, denominadas funções.

Por essa razão, a função foi adotada como uma forma de tornar os programas mais compreensíveis para seus criadores, isto é, os seres humanos. Vale salientar que o termo *função* é usado tanto na linguagem de programação C++ quanto na C. Em outras linguagens, o mesmo conceito pode ser referido como uma sub-rotina, um subprograma ou um procedimento. Sabe-se que um programa é dividido em funções, e (idealmente, no mínimo) cada função tem uma proposta claramente definida, bem como uma interface bem definida com as outras funções do programa.

A ideia de dividir um programa em funções pode ser adicionalmente estendida através do agrupamento de várias funções em uma entidade maior, denominada *módulo*. Porém, o princípio é similar, ou seja, um grupo de componentes que executa tarefas específicas. Nesse sentido, dividir um programa em funções e módulos é um dos fundamentos da *programação estruturada*. Trata-se de um paradigma de programação que tem sido utilizado na organização de programas. No entanto, à medida que os programas se tornam maiores e mais complexos, até mesmo a abordagem de programação estruturada começa a mostrar limitações.

Por exemplo, considere a situação em que muitas funções têm acesso aos dados. Como resultado, a forma através da qual os dados são armazenados torna-se crítica. A organização dos dados não pode ser modificada sem modificar todas as funções que têm acesso a eles. Se novos dados forem adicionados, será necessário modificar todas as funções que têm acesso aos dados para que elas também possam ter acesso a esses novos dados. Torna-se difícil achar tais funções e até mesmo mais difícil modificá-las adequada e corretamente.

Tais circunstâncias exigem uma forma de restringir o acesso ao dado, escondê-lo de todas exceto de algumas poucas funções críticas. Isso dará proteção aos dados, simplificará a manutenção implicará outros benefícios.

Dessa forma, considera-se importante a introdução da programação orientada a objetos (POO). A ideia por trás das linguagens de programação orientadas a objetos, ou linguagens OO, é combinar em uma única entidade tanto os dados quanto as funções que operam sobre esses dados. Tal entidade é denominada *objeto*. As funções de um objeto, chamadas funções membros em C++, tipicamente oferecem uma única forma de acesso a seus dados.

Assim, caso seja necessário ler dados em um objeto, basta chamar a função membro desse objeto. Os dados serão lidos, e o valor retornado ao objeto que invo-

cou a função. Perceba que os dados não podem ser acessados diretamente. Os dados são ocultados. Assim, os dados estão seguros quanto a qualquer alteração acidental.

Dados e funções são ditos *encapsulados* em uma única entidade denominada *objeto*. Encapsulamento e ocultação de dados são aspectos importantes na descrição de linguagens orientadas a objetos. Além disso, se o programador precisar modificar o(s) dado(s) de um objeto, ele terá de saber exatamente quais funções interagem com aquele objeto (isto é, funções membros).

Outras funções não podem ter acesso ao(s) dado(s). Como resultado, obtém-se uma simplificação da escrita, bem como depuração e manutenção do programa. Assim, um programa C++ tipicamente consiste em muitos objetos, os quais se comunicam entre si através da chamada de funções membros do(s) outro(s) objeto(s). Note que chamar a função membro de um objeto é, às vezes, também referido como *enviar uma mensagem* para o objeto.

Devido ao exposto e à grande importância que a POO tem dentro do contexto atual, dentre os paradigmas de programação considera-se essencial conhecer a linguagem C++ e saber implementar os recursos oferecidos pelo paradigma de orientação a objetos. Não importa a área na qual você se encontre (computação, engenharia e áreas afins), é de suma importância conhecer essa ferramenta, ou seja, a programação orientada a objetos. Com ela, você pode explorar a solução de diversos problemas.

11.2. PROSPECTO

C++ continua sendo uma das linguagens preferidas pelos profissionais que desenvolvem sistemas de software. Não obstante, é também uma linguagem que começa a ser cada vez mais utilizada nos cursos de engenharia devido aos diversos recursos que o paradigma de orientação a objetos oferece, permitindo que soluções mais eficientes sejam implementadas.

Cada vez mais, há uma percepção de que a linguagem C++ é nada mais que uma ferramenta que os engenheiros, profissionais de computação e áreas afins utilizam na solução de problemas. Nesse sentindo, a programação orientada a objetos é uma abordagem de programação que serve de elo entre os problemas existentes e as soluções computacionais apresentadas no campo da programação.

Vale ressaltar que, antes da POO, havia um obstáculo conceitual para os profissionais quando eles tentavam adaptar as entidades reais às restrições impostas pelas linguagens e técnicas de programação tradicionais. Em uma situação real, o ser humano tende a raciocinar em termos dos objetos ou entidades reais.

Mas, antes da POO, os profissionais eram ensinados a raciocinar sobre os problemas em termos de blocos de código ou procedimentos e da forma que eles atuavam sobre os dados. Observe que essas duas abordagens são distintas e constituem um problema se há necessidade de desenvolver um sistema complexo e/ou de grande porte.

A POO apresenta-se como uma linguagem de programação que permite aos programadores raciocinar e solucionar problemas em termos de objetos, os quais estão diretamente associados às entidades ou coisas reais. Como resultado desse mapeamento natural, utilizando a POO um profissional pode concentrar-se nos objetos que compõem o sistema, em vez de tentar vislumbrar o sistema como um conjunto de procedimentos e dados.

Cabe também destacar que a POO é uma forma natural e lógica pela qual os seres humanos, especificamente os estudantes e profissionais, raciocinam.

Os benefícios resultantes de empregar a POO como abordagem de ferramenta de programação (e solução de problemas) não se restringem a raciocinar e resolver problemas em termos de objetos ou entidades reais, mas também na reutilização de código.

Além disso, conceitos adicionais como classes, encapsulamento, polimorfismo e herança podem ser implementados. Além da identificação de uma técnica de programação adequada para a solução de problemas e a implementação de sistemas, há a necessidade de selecionar uma linguagem de programação. A linguagem C++ desenvolvida nos laboratórios da Bell por Bjarne Stroustrup é bastante utilizada por estudantes e profissionais das áreas de computação, engenharias e áreas afins.

Embora a C++ mantenha algumas semelhanças com a linguagem C, ela oferece menor possibilidade de erros comparativamente. Além disso, C++ é uma linguagem padrão que possibilita escrever programas portáveis que podem ser executados em mais de uma plataforma. Portabilidade e reusabilidade são duas propriedades essenciais muito desejadas na solução de problemas e implementação de sistemas. Ambas são suportadas pela linguagem C++, juntamente com as outras características da programação orientada a objetos.

Adicionalmente, três prováveis tendências na área de programação englobam: (i) a crescente necessidade de integração de subsistemas e componentes de software, (ii) o uso continuado da programação orientada a objetos para prover suporte à reusabilidade e à portabilidade, (iii) a adoção crescente da linguagem C++ em aplicações de engenharia como ferramenta de soluções de problemas (que demandam eficiência).

11.3. APLICAÇÕES NA ENGENHARIA

O paradigma orientado a objetos pode ser aplicado em problemas de engenharia? A linguagem C++ pode ser utilizada para implementar soluções de problemas de engenharia?

A resposta é sim, para as duas questões. Nas engenharias, investigamos a solução de problemas. O primeiro passo da solução é fazer a modelagem do problema, e em seguida esse modelo inicial é refinado para um modelo matemático.

Durante a construção do modelo matemático, faz-se levantamento de dados, realizando medidas, escolhendo parâmetros, de modo a detalhar ao máximo o modelo que se tem em mãos. Quando se tem um modelo suficientemente detalhado, ele é transformado em um algoritmo, que nada mais é do que um conjunto de instruções que permite obter uma solução para uma dada entrada.

Perceba que você pode buscar, dentre os métodos existentes, se há algum que ofereça solução. Aqui, você se encontra no domínio que envolve os métodos computacionais numéricos que auxiliam na solução de diversos problemas de engenharia. Esses métodos numéricos, assim como outras soluções de natureza algorítmica que retratam modelos de solução de problemas, podem ser implementados com a linguagem C++.

Nesse sentido, você tem em mãos processos numéricos para a resolução de problemas (algoritmos), os quais visam a máxima economia e confiabilidade em termos de fatores envolvidos, como tempo de execução, memória utilizada e erros de arredondamento.

Entretanto, o algoritmo adequado para determinada aplicação depende, por exemplo, do tamanho da entrada e correção da solução. Nesse caso, um algoritmo é dito correto se, para toda a instância, ele termina com a saída correta, ou seja, produz uma solução correta.

De modo geral, os algoritmos numéricos são fundamentais na resolução de problemas de engenharia que exigem processamento numérico, otimização, integração e derivação numéricas, além de uso de equações diferenciais na modelagem e solução de problemas envolvendo o fluxo de corrente elétrica em circuitos, a dissipação de calor em objetos, a propagação e detecção de ondas sísmicas, o aumento ou diminuição de populações, entre outros.

Perceba que a oportunidade de aplicação é grande para essa nova ferramenta que você aprendeu.

11.4. SUGESTÕES DE LEITURA

A seguir, um conjunto de recomendações de fontes adicionais de leitura e pesquisa é proporcionado ao leitor. Essas sugestões servem como um complemento ao seu estudo.

The C++ Programming Language, site de Bjarne Stroustrup (criador da linguagem C++)
http://www.research.att.com/~bs/C++.html

C++ FAQ LITE – Frequently Asked Questions
http://www.parashift.com/c++-faq-lite/

Ambiente do Compilador Dev-C++
http://www.bloodshed.net/dev/

Ambiente do Compilador Borland C++ Builder
http://www.borland.com/bcppbuilder/

Site de referência sobre a linguagem C++
http://www.cppreference.com/wiki/

Jeff Sutherland's Object Technology Web Site
http://jeffsutherland.com/

Linguagem C++ na Wikipédia
http://en.wikipedia.org/wiki/C%2B%2B

Programação orientada a objetos na Wikipédia
http://en.wikipedia.org/wiki/Object-oriented_programming