ANÁLISE E PROJETO DE SISTEMAS

Priscila Gonçalves

Diagramas de componentes

Objetivos de aprendizagem

Ao final deste texto, você deve apresentar os seguintes aprendizados:

- Reconhecer conceitos sobre o diagrama de componentes.
- Descrever os elementos de um diagrama de componentes.
- Identificar um diagrama de componentes por meio de exemplos práticos.

Introdução

Os diagramas de componentes suportam o desenvolvimento com base em componentes nos quais um sistema de software é dividido em componentes e interfaces que são reutilizáveis e substituíveis. Esses diagramas são utilizados para modelar sistemas de software em alto nível ou para apresentar componentes em nível de pacote mais baixo. Apresentam estruturas do sistema de software, que descrevem os componentes do software, suas interfaces e dependências.

Neste capítulo, você vai estudar conceitos sobre diagramas de componentes, saberá identificar e descrever seus componentes e aprenderá aplicações práticas de diagramas de componentes.

Diagrama de componentes e seus conceitos

Em UML (linguagem de modelagem unificada), os diagramas de componentes apresentam estruturas do sistema de software, que descrevem os componentes do software, interfaces e suas dependências. Além disso, ajudam a enxergar a estrutura geral do sistema e o comportamento de serviço que as partes fornecem e consumi-las por meio de interfaces.

Os diagramas de componentes suportam o desenvolvimento com base em componentes no qual um sistema de software é dividido em componentes e interfaces que são reutilizáveis e substituíveis. Utilizam-se desses diagramas

para modelar sistemas de software em um alto nível ou para apresentar componentes em nível de pacote mais baixo.

Aplicações e utilizações dos diagramas de componentes

Os diagramas de componentes têm diversas utilidades, algumas das quais são apresentadas a seguir.

- Definem aspectos executáveis e reutilizáveis de um sistema de software.
- Apresentam problemas de configuração de software por meio de relacionamentos de dependência.
- Apresentam uma representação precisa de um aplicativo de software antes de fazer alterações ou aprimoramentos.
- Descrevem as seguintes peças físicas de um sistema de software:
  - Arquivos de código-fonte desenvolvidos em ambiente de desenvolvimento integrado.
  - Arquivos executáveis necessários para fornecer um sistema em execução.
  - Bancos de dados físicos que armazenam informações nas tabelas de um banco de dados relacional ou em páginas de um banco de dados orientado a objetos.
  - Sistemas adaptáveis que possuam componentes que migrem para equilíbrio de carga e recuperação de defeitos.
- Descrevem um design implementado em qualquer idioma ou estilo, sendo necessário identificar partes do design que interagem com as outras partes de um projeto por meio de um conjunto de entradas e saídas.

Ainda há mais benefícios em relação ao desenho de diagramas de componentes, entre os quais se pode citar o pensamento de design em relação a blocos principais que ajudam a equipe de desenvolvedores a compreender de uma melhor forma e criar um novo design.

Deve-se pensar no sistema como uma coleção de vários componentes, com interfaces bem definidas e obrigatórias, de modo que se pode melhorar a separação entre esses componentes, facilitando a compreensão e a alteração de design quando requisitos sofrerem alterações. Pode-se ainda utilizar um diagrama de componente como forma de representar o design, independentemente da linguagem ou plataforma que será utilizada.

Saiba mais

Um sistema de software completo pode ser considerado um componente. Este, por sua vez, é uma unidade modular substituível dentro de seu ambiente. Possui uma ou mais interfaces fornecidas bem definidas por meio das quais suas funções podem ser acessadas, e pode ter interfaces obrigatórias, que definem quais funções ou serviços são exigidos de outros componentes. Ao conectar interfaces fornecidas e obrigatórias de vários componentes, um componente de grande proporção pode ser construído.

Elementos de um diagrama de componentes

Para melhor entender de que se trata um diagrama de componente, apresenta-se a seguir mais a respeito de seus elementos.

Componentes

Na modelagem UML (linguagem de modelagem unificada), componentes são elementos de um modelo que representam partes independentes e intercambiáveis de um sistema. Estão sempre em conformidade e realizam uma ou mais interfaces que são fornecidas e requeridas, determinando o comportamento desses componentes.

Instâncias de componentes

São elementos de modelo que representam entidades reais em um sistema.

Pacotes

Tem a função de agrupar elementos de modelos relacionados de todos os tipos, incluindo outros pacotes.

Artefatos

São elementos de modelo que representam as entidades físicas em um sistema de software. Eles representam unidades físicas de execução, podendo ser arquivos executáveis, bibliotecas, componentes de software, documentos e bancos de dados.

Interfaces

Trata-se de elementos do modelo que servem para definir conjuntos de operações que outros elementos do modelo, como classes ou componentes, devem implementar.

Relacionamentos em diagramas de componentes

Um relacionamento é uma conexão entre elementos de modelo, um tipo de elemento de modelo que inclui semântica, define a estrutura e o comportamento entre elementos de modelo.

A Figura 1, a seguir, descreve como ocorre a leitura dos diagramas de componente, e é seguida pelo Quadro 1, que apresenta uma explicação referente à Figura 1.

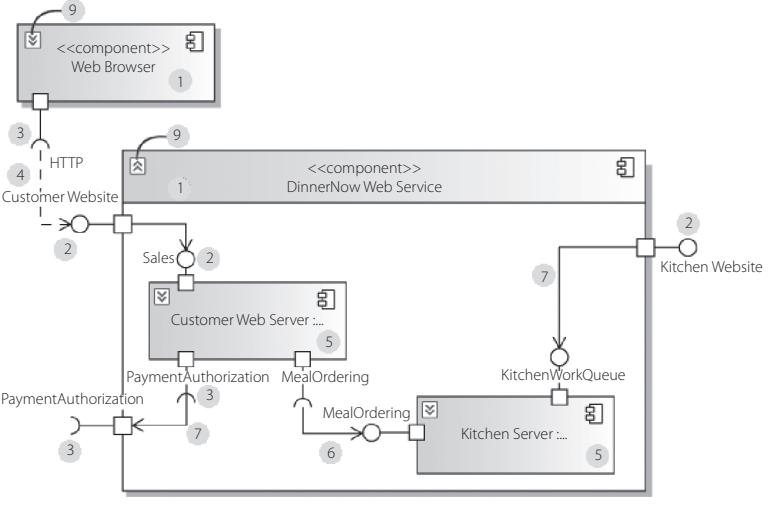


Figura 1. Diagrama de componente.

Fonte: Visual Studio 2015 (c2018).

Quadro 1. Quadro explicativo referente à Figura 1

Forma	Elemento	Descrição e propriedades principais
1	**Componente**	Um pedaço reutilizável de funcionalidade do sistema. Um componente fornece e consome o comportamento por meio de interfaces e pode usar outros componentes. Você pode ocultar ou mostrar as partes internas de um componente usando o controle expandir/recolher (9). Um componente é um tipo de classe. ■ **é instanciado indiretamente**. Se true (padrão), o componente existe apenas como um artefato de design. Em tempo de execução, apenas suas partes existem.
2	**Fornecida a porta de interface**	Representa um grupo de mensagens ou chamadas que implementa um componente e que outros componentes ou sistemas externos podem usar. Uma porta é uma propriedade de um componente que tem uma interface como seu tipo.
3	**Porta de interface necessária**	Representa um grupo de mensagens ou chamadas que o componente envia a outros componentes ou sistemas externos. O componente é projetado para ser acoplado a componentes que fornecem pelo menos essas operações. A porta possui uma interface como seu tipo.
4	**Dependência**	Pode ser usado para indicar que uma interface obrigatória em um componente pode ser satisfeita por uma interface fornecida em outro. Dependências também podem ser usadas mais geral entre elementos de modelo, para mostrar que o design de um depende do design do outro.

(Continua)

(Continuação)

Quadro 1. Quadro explicativo referente à Figura 1

Forma	Elemento	Descrição e propriedades principais
5	**Parte**	Um atributo de um componente, cujo tipo é um componente geralmente outro. Uma parte é usada no design interno do componente pai. Partes são exibidas graficamente, aninhados no componente pai. Para criar uma parte de um tipo de componente existente, arraste o componente do Gerenciador de modelos UML para o componente do proprietário. Para criar uma parte de um novo tipo, clique o **componente** ferramenta e, em seguida, clique no componente do proprietário. Por exemplo, um componente Car tem partes engine:CarEngine, backLeft:Wheel, frontRight:Wheel, e assim por diante. Mais de uma parte pode ter o mesmo tipo e componentes diferentes podem ter partes do mesmo tipo. ■ **Tipo**. O tipo de parte, que é definido em outro lugar no modelo. Normalmente, o tipo é outro componente. ■ **Multiplicidade**. O padrão é 1. Você pode defini-lo como **0** para indicar que a parte pode ter o valor **nulo**, * para indicar que a parte é uma coleção de instâncias do tipo em questão ou com qualquer expressão que possa ser avaliada como um intervalo de números.
6	**Assembly de Partes**	Uma conexão entre as portas de interface obrigatória de uma parte e as portas de interface fornecida de outra. A implementação de um assembly de parte pode variar de um componente para outro. As partes de conectado devem ter o mesmo componente pai.

(Continua)

(Continua)

Quadro 1. Quadro explicativo referente à Figura 1

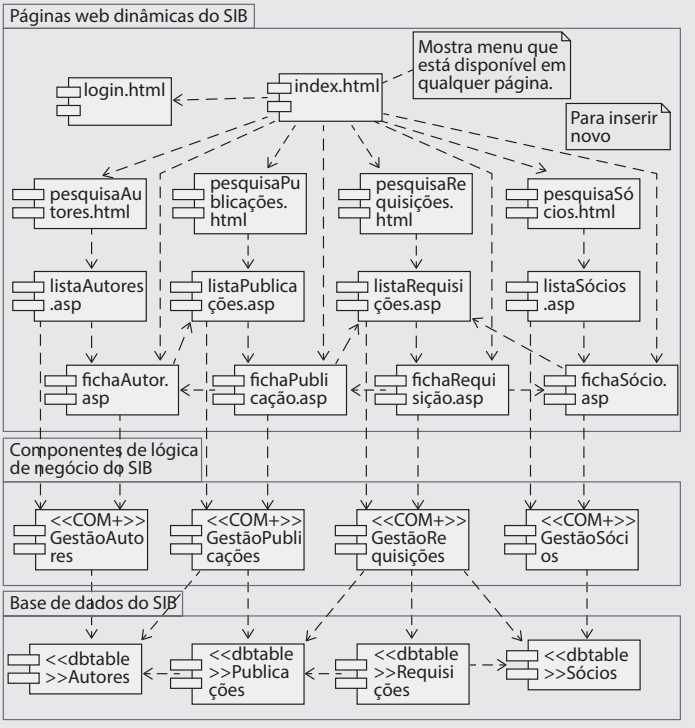
Forma	Elemento	Descrição e propriedades principais
7	**Delegação**	Vincula uma porta para uma interface de uma das partes do componente. Indica que as mensagens enviadas para o componente são tratadas pela parte ou que as mensagens enviadas da parte são enviadas do componente pai.
(não mostrado)	**Generalização**	Indica que um componente herda de outro componente. Partes e interfaces são herdadas.
9	Recolher/ expandir controle	Use isto para ocultar ou mostrar as partes internas do componente.
(não mostrado)	**Comentário**	Para obter notas adicionais. Você pode vincular um comentário para qualquer número de elementos no diagrama usando o **conector** ferramenta.

Fonte: Adaptado de Visual Studio 2015 (c2018).

No quadro apresentado, consta a explicação de como realizar a leitura de cada um dos elementos do diagrama. Segundo Scott Ambler (2004), o principal objetivo desse diagrama é possibilitar a construção de artefatos para o perfil de arquitetura da solução, seja para a arquitetura técnica ou a de negócios.

Exemplo

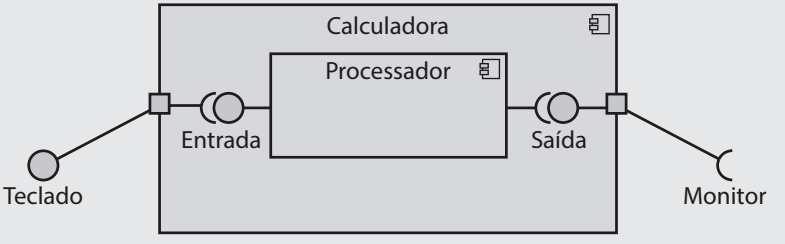
O exemplo a seguir pertence ao diagrama de componente referente a um caso de estudo de um sistema de gestão da biblioteca.



Fonte: Adaptado de Gemo e Saugene (2012).

Saiba mais

Confira a seguir as portas — elemento que permite que subcomponentes de um componente maior se comuniquem com elementos externos.



Fonte: Adaptada de UML diagrama de componentes (2013).

Diagrama de componentes: exemplos práticos

Diagramas de componentes são conectados por meio de interfaces implementadas, geralmente utilizando notação de bola e soquete, assim como ocorre em diagramas de classes. Há a possibilidade de decompor os componentes utilizando diagramas de estrutura composta.

Um ponto importante sobre os componentes é que eles representam peças que podem ser adquiridas e/ou atualizadas de forma independente. O uso de diagramas deve ocorrer quando você precisar dividir seu sistema em componentes e necessitar mostrar relacionamentos por meio de interfaces.

Outra questão característica para o diagrama de componente que constava em UML 1 era um símbolo característico que foi retirado em UML 2, permitindo que fosse anotada uma caixa de classe com um ícone de aparência semelhante. Assim, é possível utilizar a palavra-chave <<componente>> apresentada na Figura 2, a seguir.

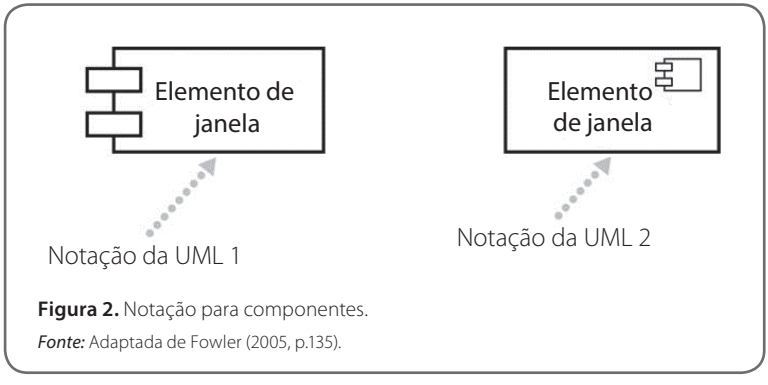


Figura 2. Notação para componentes.
Fonte: Adaptada de Fowler (2005, p.135).

A figura ilustra a retirada do ícone e a caixa de classe com o ícone de aparência semelhante pelo qual foi substituído na notação UML 1.

Na Figura 3, apresenta-se um modelo de elaboração de componente de projeto, no qual, a partir de uma análise da classe, o serviço de impressão é definido como componente na arquitetura de software e possui duas interfaces: calcularServico e iniciarServico, representadas pelo símbolo em forma de pirulito.

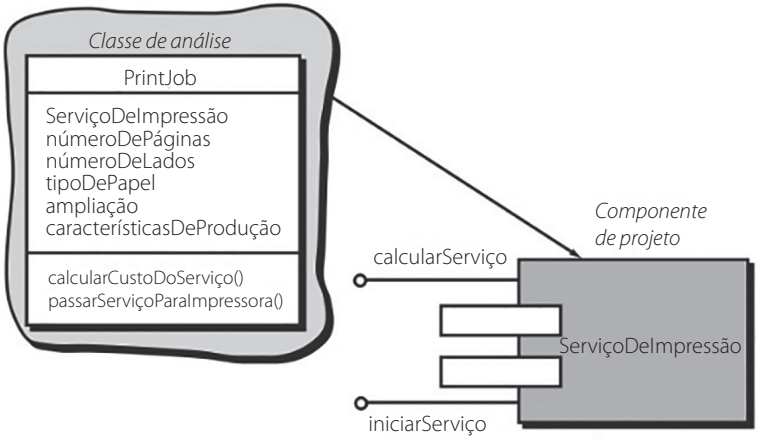


Figura 3. Elaboração de um componente de projeto.

Fonte: Pressman e Maxim (2016, p. 287).

Na Figura 4, é apresentado um modelo de diagrama de componentes, no qual uma caixa registradora conecta-se a um servidor de vendas, utilizando uma interface de mensagens de vendas. Para tornar a rede mais segura, ocorre a introdução de um componente de fila de mensagens a fim de que a caixa se comunique com o servidor se a rede estiver disponível. Dessa forma, a fila de mensagens fornece a interface de mensagens de vendas para comunicar-se com a caixa, e a interface é exigida para comunicar-se com o servidor. Este, por sua vez, é dividido em dois componentes principais: o processador de transações, que implementa a interface de mensagens de vendas, e o driver de contabilidade, que se comunica com o sistema de contabilidade.

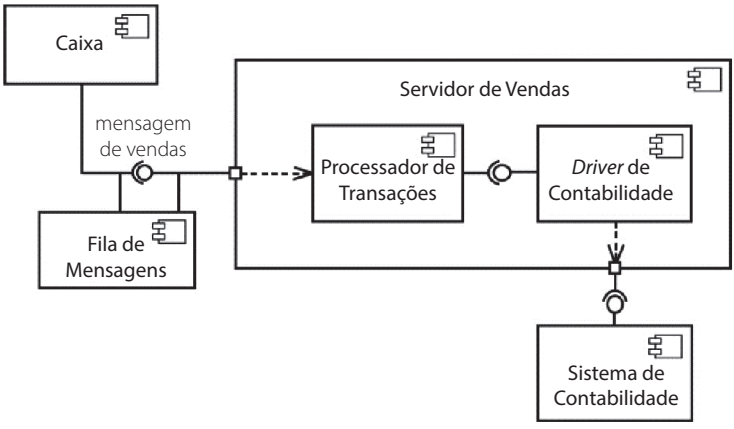


Figura 4. Exemplo de diagrama de componentes.
Fonte: Fowler (2011, p. 135).

Saiba mais

Acesse o link ou código a seguir para conhecer melhor os diagramas de componentes.

https://goo.gl/Xa62cn

Referências

AMBLER, S. W. *The object primer*: agile model driven development with UML 2.0. 3rd ed. Cambridge, UK: Cambridge University Press, 2004.

FOWLER, M. *UML essencial*: um breve guia para a linguagem padrão de modelagem de objetos. 3. ed. Porto Alegre: Bookman, 2005.

GEMO, E.; SAUGENE, Z. *UML – diagramas de componentes e modelação da arquitectura física*. 2012. Disponível em: <https://pt.slideshare.net/Portal_do_estudante_ADS/diagramas-de-componentes>. Acesso em: 01 jul. 2018.

PRESSMAN, R. S.; MAXIM, B. R. *Engenharia de software*: uma abordagem professional. 8th ed. Porto Alegre: AMGH, 2016.

UML: diagrama de componentes. *Revista Brasileira de Web*: tecnologia, 2013. Disponível em: <http://www.revistabw.com.br/revistabw/uml-diagrama-de-componentes/>. Acesso em: 01 jul. 2018.

VISUAL STUDIO 2015. *Diagramas de componente UML*: referência. c2018. Disponível em: <https://msdn.microsoft.com/pt-br/library/dd409390.aspx>. Acesso em: 01 jul. 2018.

Leituras recomendadas

AMBLER, S. W. *UML 2 component diagramming guidelines*. c2014. Disponível em: <http://agilemodeling.com/style/componentDiagram.htm>. Acesso em: 01 jul. 2018.

AMBLER, S. W. *UML 2 component diagrams*: an agile introduction. c2014. Disponível em: <http://agilemodeling.com/artifacts/componentDiagram.htm>. Acesso em: 01 jul. 2018.

DIAGRAMA de componentes. c2006. Disponível em: <http://mds.cultura.gov.br/core.base_rup/guidances/guidelines/component_diagram_CC93D7B3.html>. Acesso em: 01 jul. 2018.

OBJECT MANAGEMENT GROUP (OMG). *OMG Unified Modeling Language (OMG UML), version 2.5*. March, 2015. Disponível em: <www.omg.org/spec/UML/2.5/PDF>. Acesso em: 01 jul. 2018.

Encerra aqui o trecho do livro disponibilizado para esta Unidade de Aprendizagem. Na Biblioteca Virtual da Instituição, você encontra a obra na íntegra.

Conteúdo: