EMPREGAR OS DIAGRAMAS DE COMPONENTES E DE IMPLANTAÇÃO

DIAGRAMA DE COMPONENTES

O diagrama de componentes apresenta o relacionamento entre os diferentes componentes de um sistema de software. Na UML, o termo "componente" refere-se a um módulo que representa sistemas ou subsistemas com capacidade de interagir. Para isso, existe uma abordagem de desenvolvimento em torno de componentes.

Nesse diagrama, são representados os diferentes componentes de software necessários para que o sistema funcione corretamente.

Em uma abordagem de programação orientada a objetos, o desenvolvedor usa o diagrama de componentes para agrupar classes com base em um objetivo comum. Assim, é função do diagrama de componentes documentar como os componentes estão estruturados, organizados e como se relacionam, permitindo uma melhor compreensão e facilitando a sua reutilização. Também serve para modelar arquivos de processos de negócio, descrevendo as partes que participam dele e suas relações.

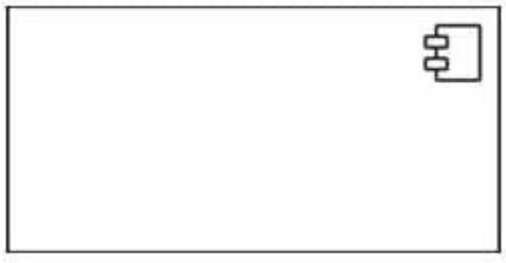
Um diagrama de componentes suporta tanto a especificação dos componentes lógicos, como, por exemplo, componentes de negócio, de processo, entre outros. Os principais benefícios na construção desse diagrama são:

Permitir visualizar a estrutura lógica e física do sistema.
Identificar os componentes do sistema e como eles se relacionam.
Perceber o comportamento do serviço quanto à interface.

Componente

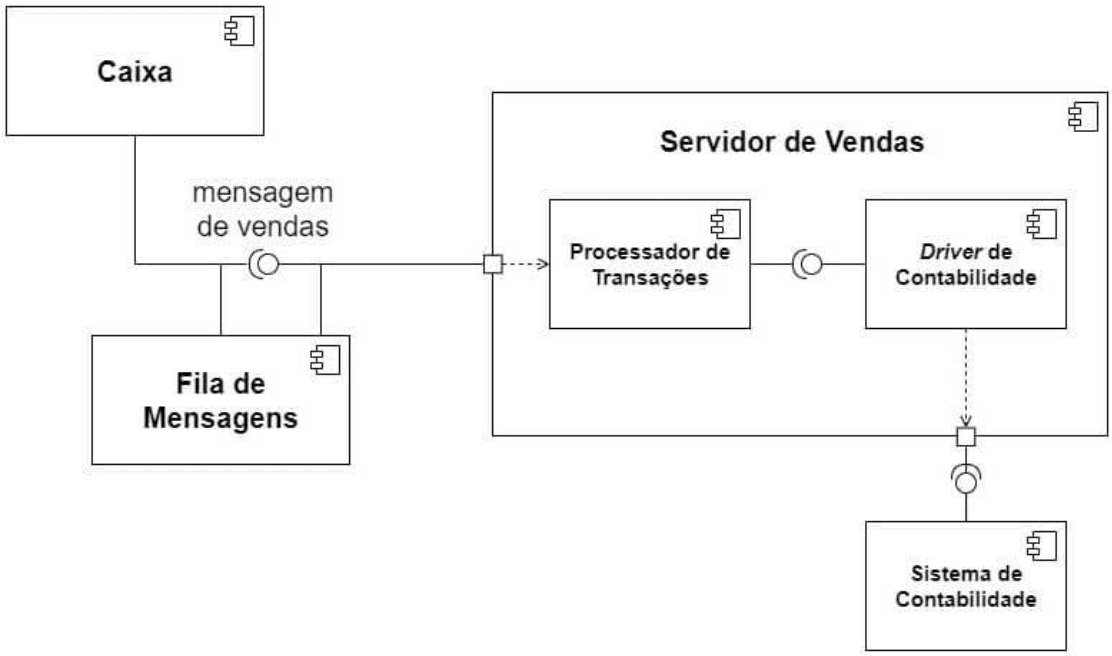
Um componente nada mais é do que uma caixa preta (artefato do sistema) que possui vida autônoma e que, somente por meio de suas interfaces, oferece ou requer serviços de outras caixas pretas, sem que para isso tenha que conhecer seus conteúdos. Um componente é modelado ao longo do ciclo de vida de desenvolvimento, incluindo a concepção da funcionalidade do sistema por meio de casos de uso, definição das classes, entre outros.

Na UML, os componentes também fazem uso de estereótipos. Os mais utilizados são arquivos, tabela, banco de dados, executável e biblioteca. Assim sendo, podemos dizer que um componente é qualquer arquivo que seja necessário à execução do sistema de informação.



Os componentes são elementos de alto nível que permitem o agrupamento de várias interfaces sem considerá-las um pacote. Cada componente é representado por um retângulo contendo um nome que deve traduzir sua função. Se necessário, pode conter um estereótipo. No canto superior do retângulo, há um ícone que é também um retângulo com dois menores sobrepostos.

Os componentes também podem ser descritos por meio da visão externa usando símbolos de interface colados na caixa do componente. Nesse caso, é interessante anexar uma máquina de estados para interfaces, portas e para o próprio componente para permitir uma descrição mais precisa.



Exemplo de diagrama de componentes. Elaborado na ferramenta Astah UML.

Quando mais detalhes são necessários para a perfeita especificação de um componente, podemos definir uma estrutura interna de partes e conectores, usando o que podemos chamar de visão interna ou “caixa branca”. Nessa estrutura, geralmente estão contidos outros componentes. Esses componentes podem ser implementados por diferentes tipos de arquivo. No diagrama de componentes, esses tipos podem ser destacados por meio de estereótipos, representados por rótulos acima do nome do componente. Nas diretrizes da UML definidas pela OMG, um artefato é dito como sendo definido pelo usuário e representa um elemento concreto do mundo físico. Ele pode ter propriedades ou operações e pode ser instanciado ou associado com outros artefatos.

Interfaces

Interfaces são coleções de operações que especificam serviços de um componente. É por meio delas que os componentes se comunicam com o mundo externo, seja para oferecer ou receber serviços. Interfaces descrevem o comportamento visível externamente e não especificam qualquer estrutura (não podem incluir atributos), nem qualquer implementação.

As interfaces fazem as associações entre os componentes do software.

As interfaces podem ser chamadas quando requerem ou esperam serviços. Podem ser opcionalmente por meio de portas onde é possível haver informações adicionais, tais como requisito de desempenho ou políticas que determinem que a interface seja realizada de maneira adequada.

Conectores

São utilizados nos diagramas de componentes para incluir interfaces baseadas em restrições e notações. Existem dois tipos principais de conectores:

Clique nas informações a seguir.

Conector*Assembly*

Conecta dois componentes para definir que um deles fornece os serviços que o outro requer.

Conector*Delegate*

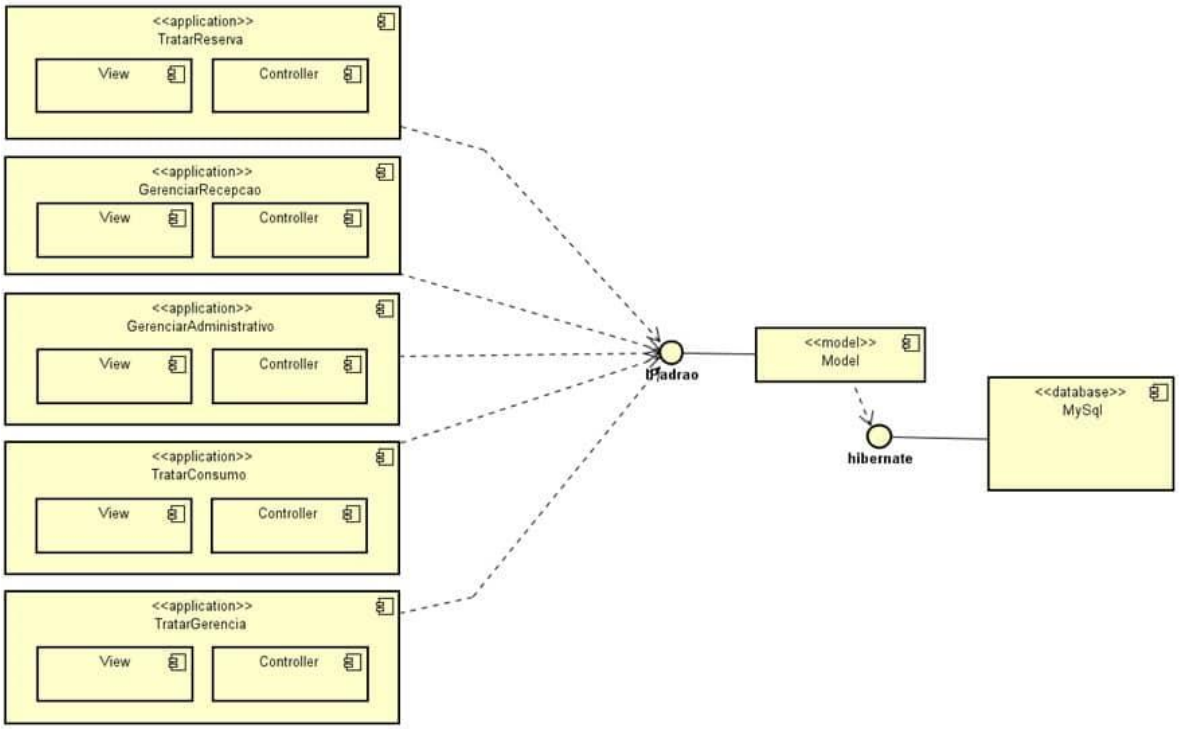
Indica que um sinal que chega para um componente será transferido para o outro para tratamento ou manipulação. Isso significa que quando colocamos um conector de delegação, estamos dizendo que naquela instância do componente o trabalho não será realizado, mas sim por outro componente. Esse tipo de conector é bastante usado para modelar decomposição hierárquica em serviços.

Camadas

A arquitetura de sistemas em camadas permite isolar a camada de apresentação das de regras de negócio e de armazenamento de dados. Isso permite aumentar o desempenho do sistema, além de facilitar a manutenção, atualização e substituição de componentes. Isso é muito interessante nos sistemas atuais que se caracterizam muitas vezes por atender grande número de usuários e áreas de negócio.

Um exemplo clássico de arquitetura em camadas é o padrão de projeto MVC *(Model – View – Controller)*, em que os componentes da aplicação consistem na interface *(View)* que invoca o controlador *(Controller)*, o qual, por sua vez, acessa o banco de dados *(Model)*.

Essa separação das camadas de software permite a modularização e o reuso dos componentes do sistema. Um exemplo é ilustrado na figura a seguir, em uma aplicação hipotética (note que diversas outras formas de componentização poderiam ser adotadas, a critério do projetista).

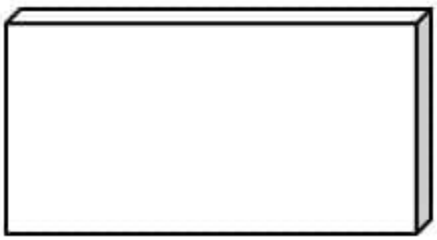


Exemplo de diagrama de componentes usando MVC. Elaborado na ferramenta Astah UML.

DIAGRAMA DE IMPLANTAÇÃO

O diagrama de implantação especifica um conjunto de artefatos de infraestrutura que podem ser utilizados para definir a arquitetura física de execução de sistemas de informação. Ele foca a organização da arquitetura sobre a qual o software irá ser implantado e executado em termos de hardware, software básico e redes de comunicação, ou seja, quais máquinas (computadores, servidores, switches etc.), quais programas de software básico (sistema operacional, sistema gerenciador de banco de dados, browser) serão necessários para suportar o sistema, bem como definir como essas máquinas serão conectadas, com qual velocidade de conexão e quais protocolos de comunicação são utilizados.

O diagrama de implantação descreve a implementação física de informações geradas pelo programa de software em componentes de hardware. Na UML, um diagrama de implantação representa a estrutura física do sistema em si, o que, assim como o diagrama de componentes, pode ser feito de várias formas.



Caixas tridimensionais representam os elementos básicos de software básico ou hardware, ou nós no sistema.

As linhas de nós a nós indicam relacionamentos.

As formas menores contidas dentro das caixas tridimensionais representam os artefatos de software contidos nos nós.

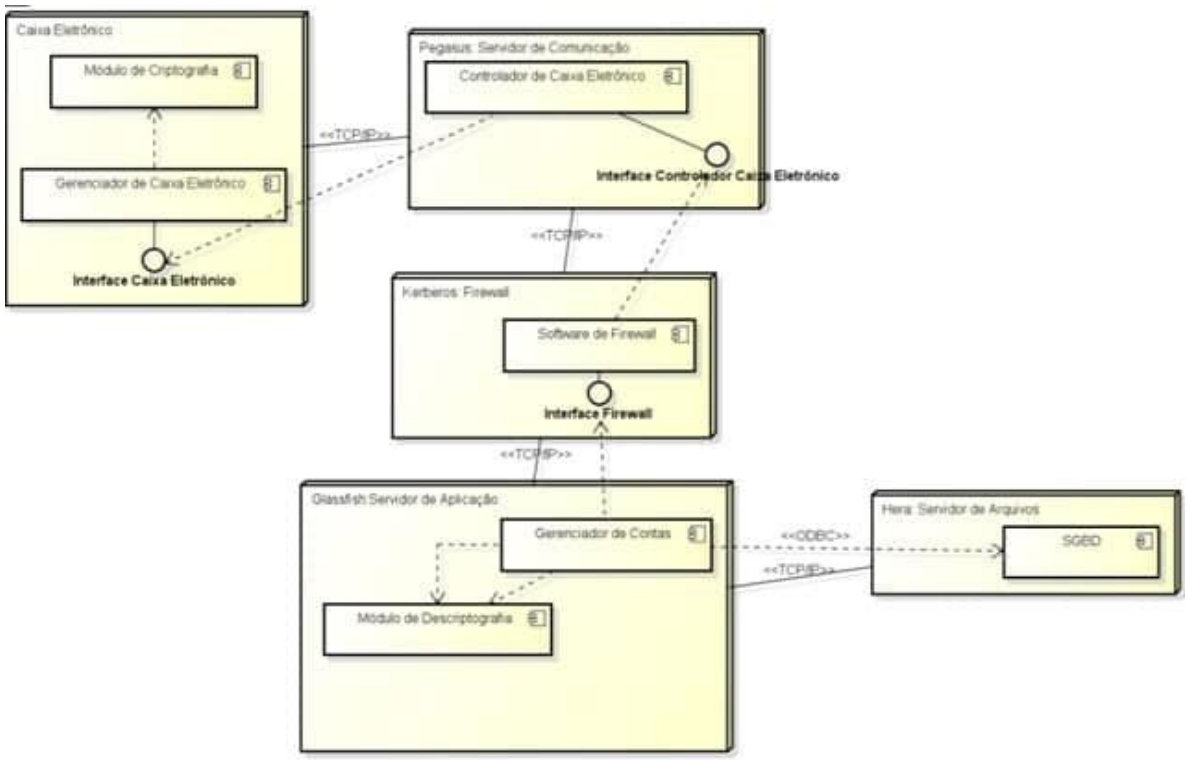
Nós do diagrama de implantação

São definidos de maneira aninhada e representam dispositivos de hardware e ambientes de execução de software. São os elementos mais básicos do diagrama de implantação; podem representar computadores pessoais e/ou servidores de bancos de dados, de aplicação ou de internet, sistemas operacionais, browser, sistemas gerenciadores de bancos de dados etc. Eles podem ser compostos de outros nós, sendo comum encontrar um nó que representa um item de hardware contendo outros nós que representam ambientes de execução.

O nó representa um elemento físico capaz de oferecer recursos computacionais e, geralmente, possui pelo menos memória e processador. Também serve para representar topologias de rede por meio de ligações entre instâncias de nós. Podem ser conectados para representar uma topologia de redes usando caminhos de comunicação que podem ser definidos como servidor de aplicação, servidor web, entre outros.

Atenção

A elaboração do diagrama de implantação depende do porte do sistema a ser desenvolvido. Por exemplo, não faz muito sentido para um sistema que será executado dentro de um único computador. Em outros casos, porém, ele pode ser muito útil, principalmente para ser utilizado como meio de comunicação entre a equipe de desenvolvimento e de infraestrutura, pois servirá de base para a equipe de infraestrutura instalar e configurar servidores, gerenciadores de bancos de dados, entre outros.
A figura a seguir ilustra um diagrama de implantação.



Exemplo de diagrama de implantação. Elaborado na ferramenta Astah UML.

Podem ser usados estereótipos que são, como em outros diagramas da UML, uma maneira gráfica ou textual de diferenciar componentes dos diagramas (classes, casos de uso, associações etc.). Podem ser representados por meio de textos escritos entre dois sinais de maior e menor quando não modificam o modelo original do componente (<<>>) e, também, por meio de um novo componente visual, porém padronizado pela UML, como os nós que podem conter os seguintes
estereótipos: **<<device>>**, **<<artifact>>**, **<<execution environment>>**.
Clique na barra para ver as informações.

DEVICE

Representa um dispositivo computacional com capacidade de processamento em que artefatos podem ser implantados para execução. Os dispositivos podem ser complexos, isto é, podem ser constituídos por outros dispositivos. Para representar um device, utilizamos o estereótipo <<device>>.

ARTEFATOS

Representam objetos concretos do mundo físico, usados e/ou resultantes de um processo de desenvolvimento. São exemplos de artefatos: programas fontes e executáveis, uma tabela de bancos de dados, scripts, um documento do sistema, uma mensagem de correio eletrônico etc. Para representar um artefato, é utilizado o estereótipo <<artifact>>. Artefatos são sempre implantados em nós.

AMBIENTE DE EXECUÇÃO

Representam uma subclasse de um nó. São exemplos de ambiente de execução: sistemas operacionais, sistemas gerenciadores de bancos de dados etc. Para representar um ambiente de execução utilizamos o estereótipo <<execution environment>>.

ASSOCIAÇÃO

Os nós são conectados por meio de caminhos de comunicação para criar sistemas em rede, ou seja, possuem ligações entre si de forma que possam se comunicar e trocar informações. Essas ligações são chamadas de associações e são representadas por segmentos de reta ligando um nó a outro. Uma associação pode conter estereótipos utilizados para determinar, por exemplo, o tipo de protocolo de comunicação usado entre os nós.