=======================================================

***Introdução a Diagrama de Deployment***

=======================================================

# **Definições**

A linguagem **UML** prevê os chamados diagramas de **Deployment** para representar uma estrutura física **(normalmente de hardware)**, onde um conjunto de artefatos de software são instalados para compor uma configuração de um sistema.

* A estrutura é construída por Nós, conectados vias de comunicação.
* Artefatos representam elementos concretos do mundo físico, resultados do processo de desenvolvimento de software, ou para a instalação ou operação de um sistema computacional.
* Um nó é um recurso computacional onde artefatos podem ser instalados para posterior execução.

# **Artefatos**

Exemplos de artefatos incluem arquivos de modelos, arquivos de código-fonte, scripts, arquivos executáveis, tabelas em bancos de dados, documento de texto, mensagem de e-mail ou qualquer outro resultado de um processo de desenvolvimento.

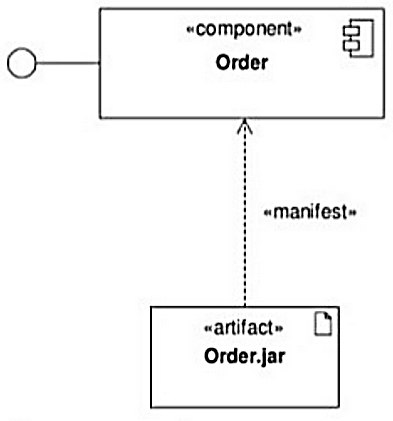
Um artefato representa, portanto, um elemento concreto do mundo físico. Uma instância particular do artefato (ou cópia do artefato) é instalada em uma instância de um nó.

Artefatos podem manter associações com outros artefatos que podem estar aninhados dentro de si próprio. Diversos estereótipos estão previstos na norma, especificando o tipo de artefato.

Por exemplo, **«source»** ou **«executable»**. Estes estereótipos podem ser ainda especializados mais ainda em um profile. Por exemplo, o estereótipo **«jar»** poderia ser definido como uma subclasse de **«executable»** de forma a designar arquivos Java executáveis.

# **Artefatos na UML**

Na linguagem **UML**, um artefato é apresentado utilizando-se o retângulo de uma classe ordinária, com o uso da palavra-chave **«artifact»**. Alternativamente, este retângulo pode acrescentar ainda um pequeno ícone no canto superior direito do retângulo.



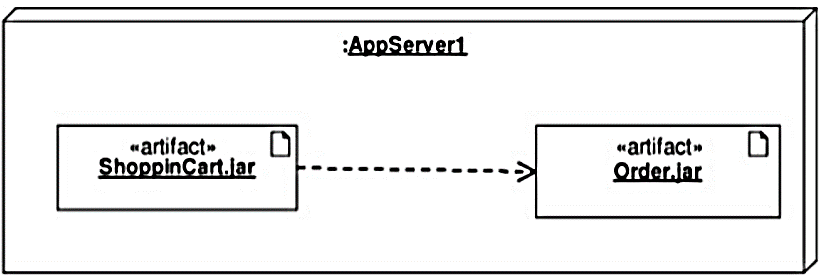
# **Nós**

Um nó é um recurso computacional onde artefatos podem ser instalados para posterior execução. Nós podem ser interconectados por meio de conexões que definem estruturas de redes.

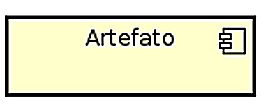
Estas conexões representam caminhos de comunicação possíveis entre os nós. Topologias específicas de redes podem ser definidas por meio dessas conexões. Nós hierárquicos (ou seja, nós dentro de nós) podem ser definidos utilizando-se associações do tipo composição, ou definindo-se uma estrutura interna para aplicações de modelagem avançada.

Arcos tracejados com o uso do keyword **«deploy»** podem ser utilizados para representar a capacidade de um nó de suportar um determinado tipo de artefato.

Alternativamente, isso pode ser representado utilizando-se artefatos aninhados dentro do nó. Em ambos os casos, isso significa que o artefato correspondente se encontra instalado no nó.



Sabemos que todo diagrama de **Deploy** (Implantação) é basicamente formado por Nós e Artefatos:

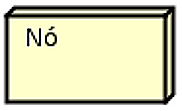


Onde os artefatos incluem arquivos de modelos, arquivos de código-fonte, scripts, arquivos executáveis, tabelas em bancos de dados, documento de texto, mensagem de e-mail ou qualquer outro resultado de um processo de desenvolvimento.

Um artefato representa, portanto, um elemento concreto do mundo físico. Uma instância particular do artefato (ou cópia do artefato) é instalada em uma instância de um nó.

Artefatos podem manter associações com outros artefatos que podem estar aninhados dentro de si próprio. Diversos estereótipos estão previstos na norma, especificando o tipo de artefato.

Por exemplo, **«source»** ou **«executable»**. Estes estereótipos podem ser ainda especializados mais ainda em um profile. Por exemplo, o estereótipo **«jar»** poderia ser definido como uma subclasse de **«executable»** de forma a designar arquivos Java executáveis.



Um nó é um recurso computacional onde artefatos podem ser instalados para posterior execução.

Nós podem ser interconectados por meio de conexões que definem estruturas de redes. Estas conexões representam caminhos de comunicação possíveis entre os nós.

Topologias específicas de redes podem ser definidas por meio dessas conexões. Nós hierárquicos (ou seja, nós dentro de nós) podem ser definidos utilizando-se associações do tipo composição, ou definindo-se uma estrutura interna para aplicações de modelagem avançada.

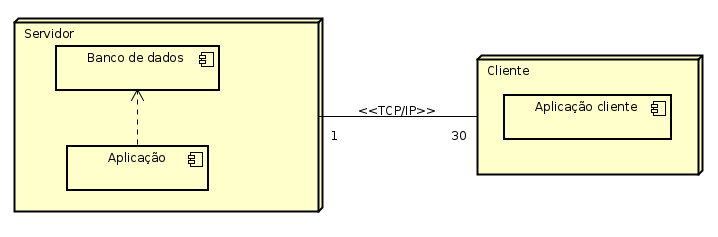
=======================================================

***Atividade  1***

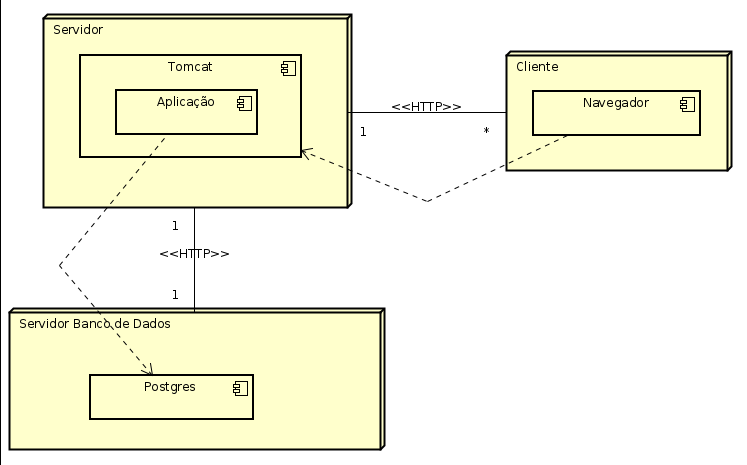
=======================================================

**1 - Desenvolva um diagrama de Deploy para cada uma das arquiteturas abaixo:**

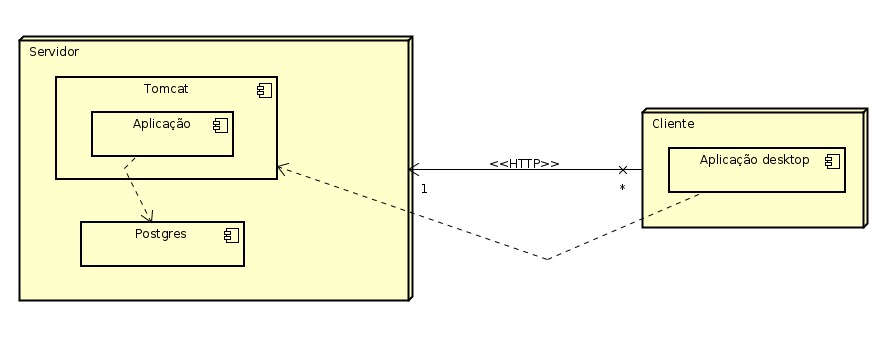
***A)*** Uma aplicação DESKTOP com um único servidor de aplicação e banco de dados, com a possibilitando conexão de no máximo 30 usuários.



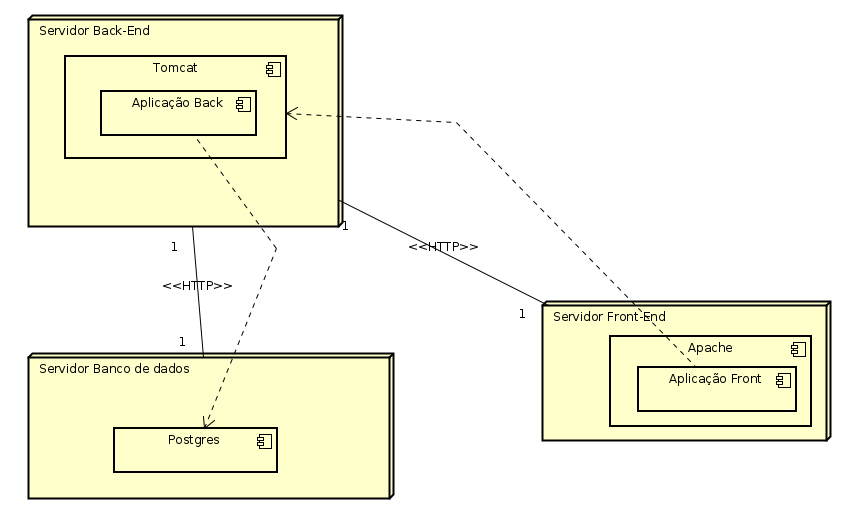
***B)*** Uma aplicação WEB com um servidor para aplicação e outro servidor para o banco de dados, com a possibilidade de um número indefinido de conexões por clientes.



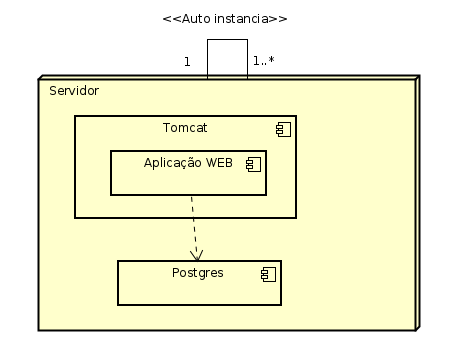
***C)*** Uma aplicação WEB com um único servidor para aplicação e banco de dados, cujo sua função é apenas receber informações de várias aplicações DESKTOP via webservice.



***D)*** Uma aplicação WEB com um servidor para front-end, outro servidor para regras de negócio e outro para o banco de dados, onde pode ser aplicado uma escalabilidade na horizontal.



***E)*** Uma aplicação WEB com apenas um servidor para aplicação e banco de dados, onde pode ser aplicado uma escalabilidade na vertical.



=======================================================

***Atividade  2***

=======================================================

**1) Qual a função do diagrama de Deployment na linguagem UML?**

Representar uma estrutura física (normalmente de hardware), onde um conjunto de artefatos de software são instalados para compor uma configuração de um sistema, podendo descrever o limite, a estrutura e o comportamento do sistema e os objetos nele contidos.

**2) Do que é composto um diagrama de Deployment? Explique.**

Um componente é modelado durante o ciclo de desenvolvimento e refinado sucessivamente durante a instalação e execução do sistema, podendo ser  modelado durante o ciclo de desenvolvimento e refinado sucessivamente durante a instalação e execução do sistema.

**3) O que um NÓ representa no diagrama de Deployment?**

Uma entidade física que executa um ou mais componentes, subsistemas ou executáveis, podendo ser um hardware ou elemento de software.

**4) O que um Artefato representa no diagrama de Deployment?**

Unidades físicas de execução, como por exemplo arquivos executáveis, bibliotecas, componentes de software, documentos e bancos de dados.

**5) Como um Artefato é representado graficamente em um diagrama de deployment?**

Um artefato é exibido como um retângulo que contém o nome do artefato, o retângulo também contém o estereótipo «artifact» e o ícone do artefato.

**6) Como um NÓ é representado graficamente em um diagrama de deployment?**

É exibido como um retângulo tridimensional que contém o nome e o ícone do nó.



**7) Como os dois componentes (Artefato e NÓ) se interagem no diagrama de deployment?**

Na modelagem UML, implemente artefatos em nós para indicar que o nó suporta a implementação de instâncias desses artefatos quando o sistema é executado.

**Antes de Iniciar**

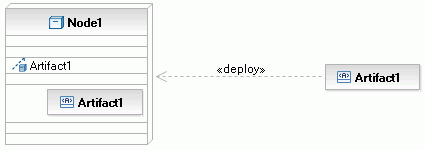
Um diagrama deve ser aberto no editor de diagramas e deve conter pelo menos um nó e um artefato.

**Por Que e Quando Desempenhar Esta Tarefa**

Para especificar a implementação de um artefato em um nó, no editor de diagramas, clique em um artefato e arraste-o sobre os compartimentos Textual Requeridas ou Gráficos de Implementações do nó. Também é possível criar um relacionamento de Implementação do artefato até o nó.

**Resultados**

Como a seguinte figura ilustra, o artefato, chamado Artifact1, é conectado por uma linha tracejada ao nó, chamado Node1. Uma ponta de seta aberta aponta do artefato para o nó. O artefato também é exibido nos compartimentos Textual Requeridas e Gráficos de Implementações do nó, se esses compartimentos estiverem mostrados no diagrama.



**8) Assinale V para Verdadeiro e F para Falso:**

**( V )** Um NÓ pode ser adicionado dentro de outro NÓ.

**( V )** O Artefato representa apenas software, porém o NÓ também pode ser usado para tal função.

**( F )** É possível projetar a implantação de um software com qualquer arquitetura de desenvolvimento utilizando o diagrama de Deployment.

**( V )** É possível um NÓ interagir com outro NÓ utilizando relacionamento N para N.

**( F )** Um Artefato não pode interagir com outro Artefato.

**(V )** Podemos adicionar vários artefatos dentro de um NÓ.