Certified Tech Developer
The Ultimate Degree

infraestrutura II

Artefatos

Artefato é um objeto ou uma série de objetos produzidos a partir dessas compilações. Esses resultados podem, por sua vez, ser usados para uma segunda ou última compilação. Temos como exemplo arquivos binários: arquivos **dll, jar, war, ear, msi, exe,** etc.

Vamos navegar entre os artefatos para entender o que são e como são gerenciados!

Navegando entre artefatos

Os aplicativos são fragmentados em diferentes componentes individuais que podem ser montados em um produto completo. Isso geralmente acontece durante a fase de compilação, quando fragmentos menores são criados. Isso normalmente é feito para fazer um uso mais eficiente dos recursos, reduzir os tempos de compilação, controlar melhor a depuração binária e assim por diante.

O tipo de artefato dependerá do produto com o qual o aplicativo está programado.

Vamos ver o tipo de artefato por produto:

- Java: jar, ear, war, etc.
- .Net: dll, exe, etc.
- Python: .py, sdist.

Também pode haver tipos de artefatos não relacionados ao produto específico, mas necessários para a operação. Por exemplo, arquivos, YAML, XML, TXT, MD, lib, so, bin, etc.

Um momento! Para tornar a imagem um pouco mais complexa, um artefato também pode ser um script, um diagrama, um modelo de dados e assim por diante.

Mas então, um artefato é qualquer tipo de arquivo? Em termos estritos, **é tudo o que resulta de um processo build.**

Administração

Um produto de repositórios de artefatos basicamente nos permite realizar as seguintes operações:

- mover
- copiar
- excluir

... artefatos para manter a consistência em cada repositório.

Quando um artefato é movido, copiado ou excluído, o produto deve atualizar automaticamente os chamados "descritores de metadados".

Exemplos disso podem ser: maven-metadata.xml, RubyGems, Npm, etc.

Metadados

Metadados são dados sobre dados! Cada artefato conterá metadados essenciais para a reutilização do código.

Um recurso existente é permitir que os desenvolvedores compartilhem código e usem componentes de terceiros. Nesse sentido, os metadados desempenham um papel fundamental na colaboração. **Por quê?**

Por exemplo, verificando os dados associados a cada artefato, podemos ver se eles foram alterados. Isso será útil para validar sua descrição, por exemplo, a versão de um produto. Poderemos saber - sobre essa mudança - quem a fez, quando, a que horas exatamente, que dependências tem, etc.

No entanto, a combinação de muitos tipos de metadados pode tornar o processo de colaboração mais complexo. Uma boa estratégia inicial é vital para evitar cair nesse tipo de situação!

Armazenamento

Onde todo esse código e seus respectivos artefatos são armazenados?

Neste ponto, está claro que o código geralmente é armazenado em sistemas de controle de versão como GitHub, GitLab ou BitBucket.

Mas este não é o caso de artefatos que podem - por exemplo - ser arquivos binários. E talvez não faça muito sentido usar um repositório de projeto para armazenar arquivos binários que são ilegíveis para humanos e podem ter um tamanho muito grande.

É aqui que **os repositórios binários** são uma parte vital do processo de integração contínua.

O repositório binário pode permitir hospedar tudo isso em um único lugar, tornando sua administração mais simples. Exemplos de produtos típicos que podemos encontrar no mercado hoje são: Artifactory, Nexus, Harbor, etc.

Também encontraremos aqueles que são baseados em nuvem: Azure Artifact, Artifact Registry, etc.

Acessando nossos artefatos

Como qualquer produto de software, sua sintaxe irá variar, mas sua propriedade será a mesma: oferecer uma maneira simples de realizar consultas que especificam critérios de pesquisa, filtros, opções de classificação e parâmetros de saída.

Isso é feito expondo uma API RESTful, pois por se tratar de objetos que devem ser utilizados imediatamente para fornecer dados de saída, o tempo de acesso e resposta deve ser extremamente rápido e com baixo consumo de memória.

Tudo em torno de uma filosofia

A importância de um tipo de artefato pode ser entendida em relação à filosofia DevOps: "desenvolver aplicativos melhores, mais rápido e com entrega constante de funções ou produtos de software".

Uma prática comum na integração contínua é construir o binário apenas uma vez, carregá-lo em um repositório binário e chamá-lo de lá para implementá-lo nos diferentes ambientes. Dessa forma, garantimos que o código base que já funciona no ambiente de desenvolvimento seja a mesma base que foi introduzida na produção.

Conclusão

Artefatos são subprodutos de um processo de desenvolvimento de software. Eles são os **componentes** com os quais um software é feito.

Eles são extremamente necessários durante o desenvolvimento, operações diárias, manutenção e atualização de software.

Sua correta administração é fundamental para o correto funcionamento de um produto gerenciado por meio de uma filosofia DevOps.