

Aula 5: Gerenciamento de Construção

Apresentação

Focaremos nossa atenção no Gerenciamento de Construção, explicando seus fundamentos, bem como algumas boas práticas gerais específicas, incluindo a gestão de dependências.

Abordaremos duas ferramentas muito conhecidas de Gerenciamento de Construção: Maven e Gradle.

Objetivo

- Examinar os fundamentos e boas práticas gerais do Gerenciamento de Construção;
- Analisar o Gerenciamento de Dependências;
- Identificar e conhecer as ferramentas Maven e Gradle.

A Gestão de Construção e seus Fundamentos

Vimos anteriormente que o Controle de Mudanças usa um volume considerável de automação para ser executado. Tal automação é fundamental para que o Controle de Mudanças seja bem executado, afinal, o volume de itens de configuração de software sob controle somado ao já esperado e normalmente alto volume de mudanças a todo tempo fazem com que uma abordagem manual seja impraticável, senão impossível.

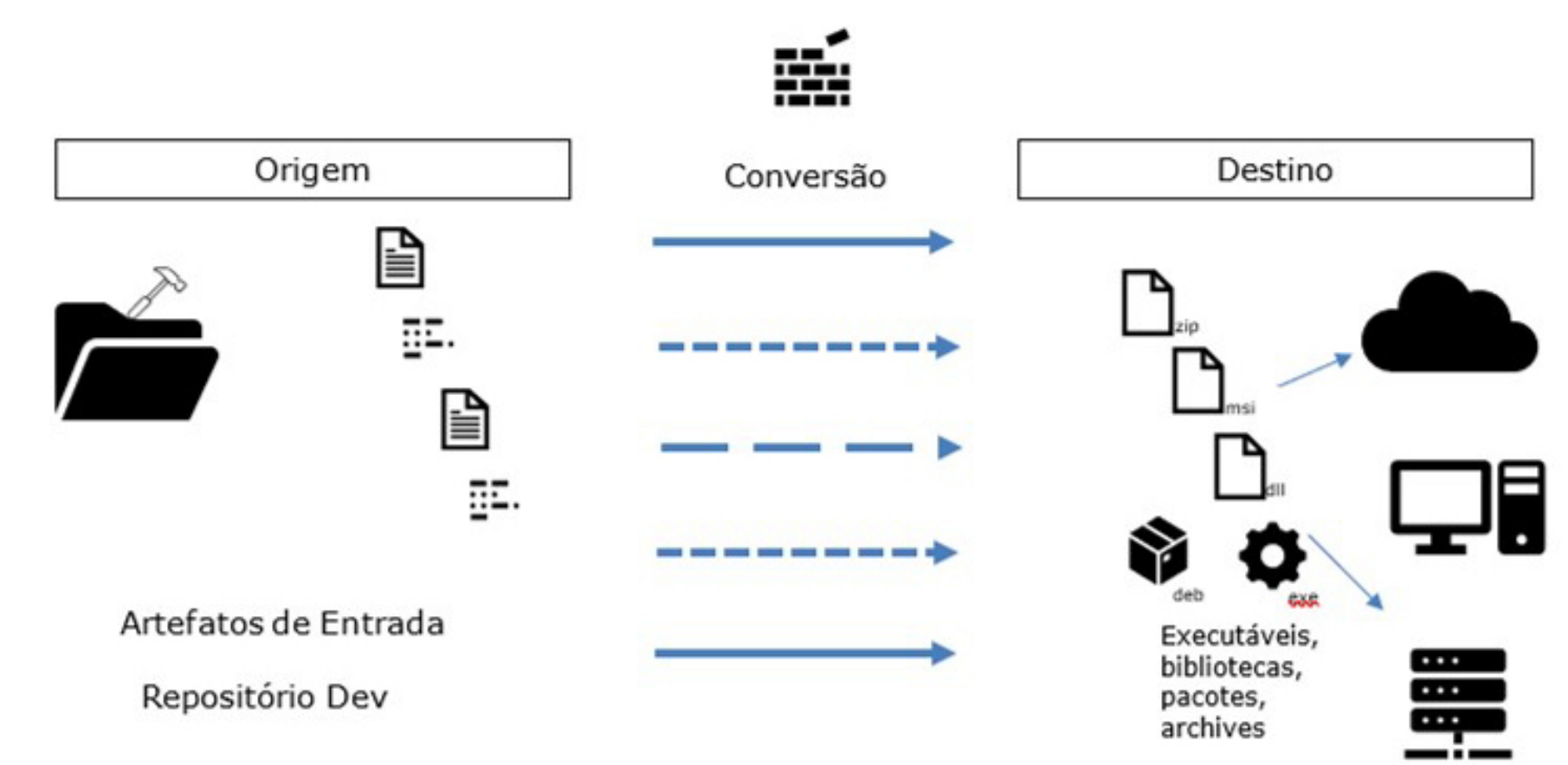
Essa necessidade por automação não é algo inerente somente ao Controle de Mudanças, mas será um aspecto doravante corriqueiro, inerente a todos os tópicos que visitaremos, a ponto de ficar claro que a Gerência de Configuração como um todo não ocorre sem forte uso de ferramentas automatizadas. O primeiro dos demais tópicos que se encaixa neste paradigma é a Gestão de Construção, já que este assunto representa uma parte importante dos ciclos de vida de desenvolvimento e teste.

A Gestão de Construção envolve justamente o passo de coletar todos os ativos e itens de configuração a serem incluídos e uni-los em uma construção (build), seja como parte de uma nova implementação, devido à correção de defeitos, em resposta a mudanças, em preparação para atender a uma liberação, concretizar uma implantação, e ainda por quaisquer outras razões pertinentes.

A palavra Gestão diz justamente respeito a automatizar e orquestrar de forma ativa tudo o que é necessário, dando um caráter mais profissional ao que eram tarefas normais do dia-a-dia.

Em outras palavras, a Gestão de Construção é toda a orquestração da automação necessária para que de fato se gere um produto de software concreto. Estão incluídas aqui as tarefas para compilar, construir e empacotar o produto, e possivelmente ainda testá-lo e implantá-lo em ambientes de teste diferentes.

Por mais que a Gerência de Construção seja automatizada, veremos mais adiante que esses passos na verdade são apenas iniciais na caminhada rumo à visão avançada promovida pela Integração Contínua e DevOps.



Atenção

O Gerenciamento de Dependências envolve a declaração, resolução e uso dessas dependências de maneira automatizada, ou seja, diretamente viabiliza as construções para os seus vários propósitos. Basicamente permitem reuso de funcionalidades, algo muito importante atualmente, permitindo ainda a criação de *frameworks* de desenvolvimento de *software* na forma de bibliotecas, componentes de terceiros etc., aumentando assim a escala e a velocidade do desenvolvimento.

O Gerenciamento de Dependências, portanto, se refere à gestão ativa das várias dependências encontradas em um esforço de desenvolvimento de software, tais como as que são encontradas entre:

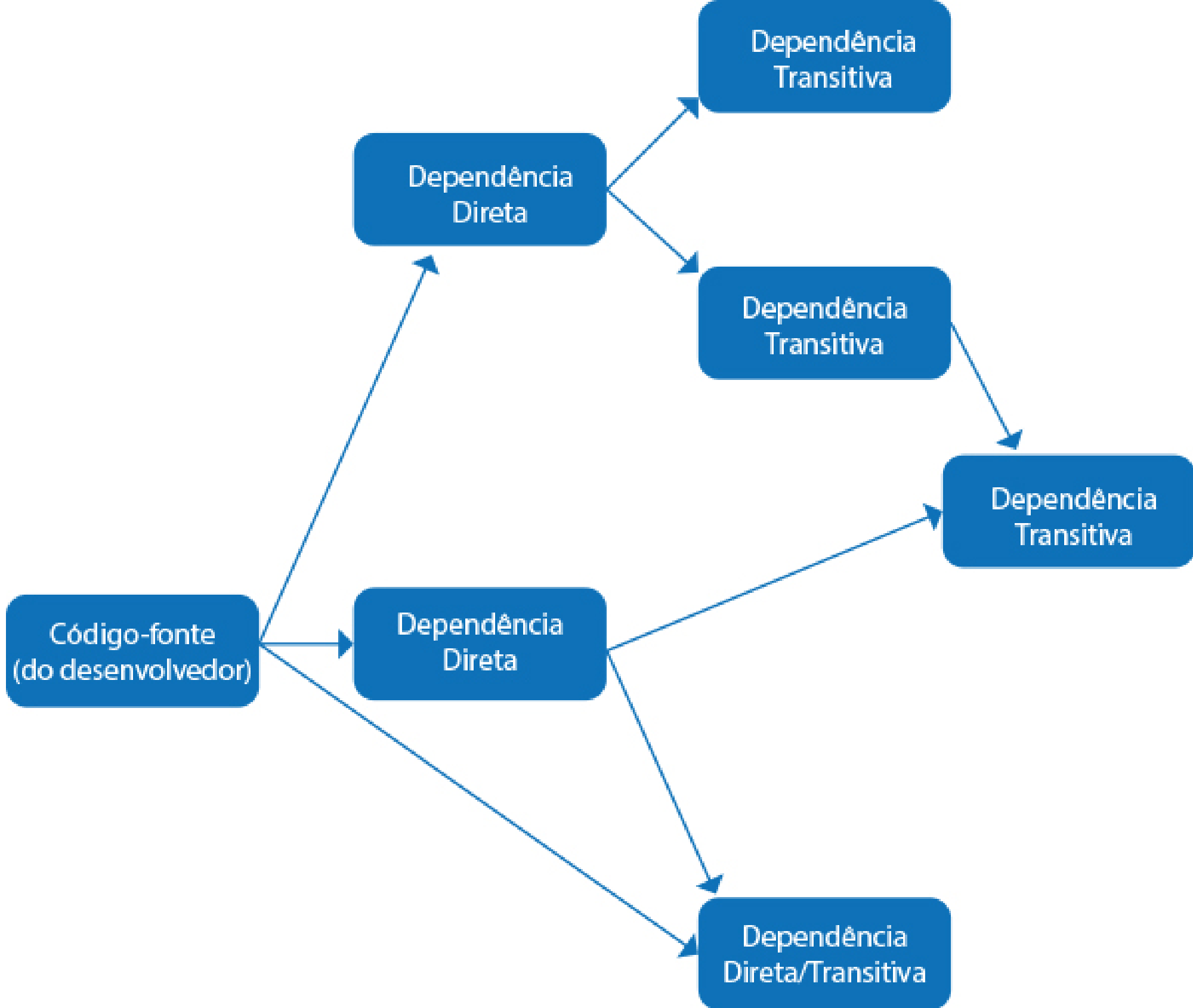
- Código-fonte;
- Arquivos individuais;
- Grupos de arquivos ou pacotes de arquivos;
- Bibliotecas internas à organização;
- Bibliotecas externas à organização (publicadas na *web*, por exemplo);
- Projetos inteiros (*jars*, por exemplo);
- Outros itens de configuração.

Constatamos aqui a importância da Gerência de Dependências como prática isolada, bem como em relação ao Gerenciamento de Construção. Projetos envolvendo dezenas, centenas ou até milhares de dependências cairiam em uma verdadeira espiral negativa caso essas dependências precisassem ser geridas manualmente.

O Gerenciamento de Dependências automatiza boa parte desse processo, permitindo visualização, navegação e análise das dependências. Isso é ainda mais importante dado o uso cada vez maior de código *open source*.

Dependências são classificadas em dois tipos em particular:





📷 A diferença entre dependências diretas e transitivas. Fonte: Adaptado de Dataverse Project.

Em termos de benefícios tangíveis do Gerenciamento de Dependências, podemos citar:

[Melhor segurança](#)



Componentes ou bibliotecas podem possuir vulnerabilidades. O Gerenciamento de Dependências ajuda a manter esses componentes atualizados, reduzindo as chances de a aplicação ser exposta e propagar o risco associado a quem a usa.

[Melhorias de performance](#)



Se as dependências estão desatualizadas, é provável que não se tenha acesso às últimas melhorias implementadas. A Gerência de Dependências ajuda diretamente a garantir acesso às últimas atualizações, o que possivelmente afetará a performance do aplicativo de maneira positiva.

[Garantia da qualidade](#)



A gestão de dependências ajuda na prevenção de referências circulares ou conflitantes.


[Melhor aderência a políticas de licenciamento](#)



Não é somente software proprietário que envolve termos de licenciamento. Software open source também pode trazer consigo termos semelhantes, e o Gerenciamento de Dependências ajuda a manter em cheque problemas relacionados à propriedade intelectual.

Atenção! Aqui existe uma videoaula, acesso pelo conteúdo online

 Boas Práticas na Gestão de Construção

 Clique no botão acima.

O Gerenciamento de Construção, embora específico e trazendo consigo méritos próprios, é comumente menosprezado, pois toda organização acaba por realizá-lo em algum nível, quer isto seja reconhecido como uma atividade, um processo ou qualquer outra coisa. Considerando que o foco aqui é a profissionalização desse esforço, comecemos a rever as boas práticas em relação à Gestão de Construção, com especial atenção a duas práticas específicas.

A primeira delas diz respeito à própria adoção da Gerência de Construção, o que passa por um bom mapeamento do que precisa ser feito / atingido. Em um primeiro momento, isso pode minimamente incluir coisas como:

- Quais subtarefas precisam estar envolvidas nas construções?
- Quais tipos de construção serão necessários?
- Quais ferramentas de construção são adequadas?

Conforme o tema se aprofunde, e a organização e os profissionais adquiram visibilidade sobre a Gerência de Construção, novas considerações precisam ser feitas, como por exemplo:

- Quem irá realizar a construção?
- O que será construído?
- Quando ocorrerá a construção?
- Onde ocorrerá a construção?
- Como a construção será completada?
- Quem é responsável pelo sucesso final da construção?
- Quem é a pessoa principal a ser contatada em caso de erros?
- Quais são as dependências de cada construção?
- Quando as construções serão implantadas?
- Em que ambientes cada construção será implantada?

Construção (Build)



Resultado de

Configuração

- grupo definido de componentes com versões específicas
- componentes = unidades funcionais
- componentes e suas versões são escolhidas para entregar objetivos específicos ou tarefas

Objetivos Gerais



Funcionalidade



Performance

Um destes pontos para o qual devemos chamar mais atenção é a questão de quem realizará as construções. Uma das abordagens adotadas por organizações é delegar essa função aos analistas de configuração ou mesmo aos gerentes de configuração. Em geral, isso dependerá da carga de trabalho desses profissionais, independentemente de eles cuidarem da execução das construções; o que é importante ressaltar aqui é a necessidade de segregação de funções.

Isso significa que, qualquer que seja o papel que realize a codificação, ele deve ser separado do que irá realizar a construção. No mínimo, deve haver separação para geração dos *builds* que serão implantados em ambientes de teste e produção.

Isso representa uma salvaguarda para limitar os riscos no código sendo alterado e chegando aos ambientes sem passar pelo processo normal de *check-in*, o que poderia resultar em danos ao ambiente e em efeitos negativos para os usuários e clientes.

É óbvio que em caso de extrema necessidade, como por exemplo uma mudança ou liberação emergenciais, esse racional poderá ser relaxado até certo ponto; no entanto, a gestão de construção deve retornar ao seu estado normal após a situação de crise ter sido contornada, até para garantir que nenhuma área seja afetada pela construção / mudança / liberação emergenciais.

O segundo ponto a ser levado em consideração é a necessidade de sempre se garantir que seja possível criar construções de maneira manual. Isso pode soar contraintuitivo, mas não é, afinal sempre existe a possibilidade de algo dar errado com uma construção automatizada. Um exemplo é o *turnover*, quando os automatizadores de alguma construção em específico deixam a organização.

Outras razões para disrupção na automação de construções podem ser mudanças em sistemas operacionais, um *patch* ou atualização que quebram o processo de construção automatizada. Essas situações não são tão incomuns, e quando ocorrem podem levar dias ou semanas para serem reparadas. Em geral, é inaceitável que se perca a capacidade de realizar construções por tão longo período; logo, a manutenção da habilidade em se realizar construções manuais é altamente justificável.

Tendo sido cobertas essas duas boas práticas de maneira mais aprofundada, vejamos algumas outras boas práticas aplicáveis ao Gerenciamento de Construção:

- Realize a construção de produtos que estão sendo alterados ao menos uma vez por semana (se possível diariamente);
- Procure otimizar a árvore do código fonte, mais especificamente os ramos da árvore, de forma que para cada componente do produto se tenha no máximo um ramo;
- Comece construções durante o dia, não durante a noite;
- Publique o cronograma de construções oficial em algum lugar, se possível em um website amplamente acessível a todos os envolvidos;
- Adote um padrão claro e consistente para o nome das construções;
- Caso as construções em questão se destinem a uso internacional, se certifique de ter suporte multilíngue;
- Institua check-ins atômicos dos repositórios privados para o principal, ou seja, ao se realizar check-ins, todos os artefatos / itens de configuração devem fazer parte dele. Não se deve deixar nenhum fora.

Identificar e conhecer a ferramenta Maven

A ferramenta Maven é uma popular ferramenta, idealizada em 2002, para ser empregada em Gestão de Projetos. Apesar disso, o Maven realmente se tornou popular por suas capacidades de automação de construção, sendo usado principalmente em projetos de software que fazem uso da linguagem de programação Java. O Maven, no entanto, é capaz de suportar múltiplas linguagens de programação, entre elas Ruby on Rails, C, C# e outras.

A ferramenta Maven se estrutura ao redor do conceito de Project Object Model (POM), que é implementado por um arquivo XML, descritivo do projeto de *software* que está sendo construído. Essa descrição, embutida no arquivo POM, inclui coisas como:

Projeto sendo construído;

Dependências;

Ordem de construção;

Diretórios;

Plug-ins.

O pom.xml pode prover a configuração para um projeto inteiro, mas é comum que, à medida que os projetos cresçam em tamanho e complexidade, eles sejam divididos em módulos, cada um com seu próprio POM se reportando a um POM geral-raiz.

Também é importante dizer que é possível manter relacionamentos pai-filhos entre os POMs, ou seja, o POM filho herda todos os elementos de configuração do pai. O POM raiz também pode ser o super POM, que é geral, fornecido pelo próprio Maven, e que se sobrepõe a outros POMs.

Comentário

A estrutura POM, ao mesmo tempo em que traz simplicidades e facilidades, como, por exemplo, a centralização da configuração, também é apontada como fonte de problemas, pois, à medida em que os projetos crescem, a estrutura XML passa a ficar cada vez mais desorganizada.

Outro ponto é que o tempo de construção tende a apresentar menos previsibilidade, apresentando cada vez maiores taxas de variação.

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
    http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
```

```
  <groupId>com.example.maven</groupId>
```

```
  <artifactId>BookStore</artifactId>
```

```
  <packaging>pom</packaging>
```

```
  <version>1.0-SNAPSHOT</version>
```

```
  <modules...>
```

```
  <profiles>
```

```
    <profile...>
```

```
    <profile>
```

```
      <id>productionServer</id>
```

```
      <properties>
```

```
        <database.url>
```

```
          jdbc:postgresql://host/database
```

```
        </database.url>
```

```
      </properties>
```

```
      <dependencies>
```

```
        <dependency>
```

```
          <groupId>org.postgresql</groupId>
```

```
          <artifactId>postgresql</artifactId>
```

```
          <version>9.4-1206-jdbc4</version>
```

```
        </dependency>
```

```
      </dependencies>
```

```
    </profile>
```

```
  </profiles>
```

```
  <dependencies...>
```

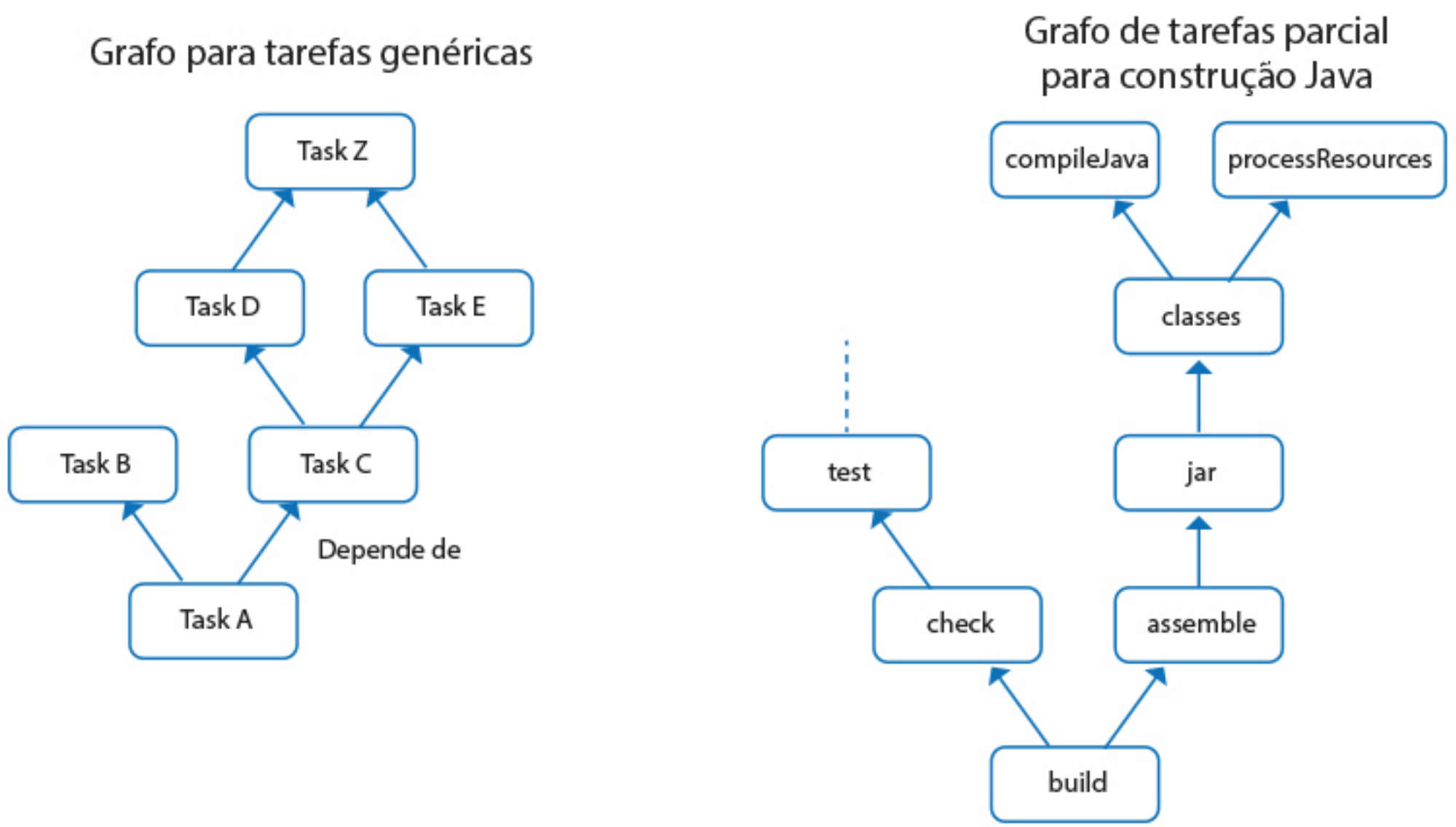
```
</project>
```

📄 Visão interna de um arquivo pom.xml. Adaptado de JetBrains.

Identificar e conhecer a ferramenta Gradle

O Gradle é outra ferramenta voltada para o suporte da automação em construção. Famosa por ser empregada no Android Studio, onde constrói apps móveis para a plataforma Android, é flexível o bastante para ser usada em conjunto com outras plataformas e/ou linguagens de programação, tais como Python, por exemplo. Essa flexibilidade se dá em parte graças ao fato de o Gradle rodar sobre uma Java Virtual Machine.

Outra parte da flexibilidade (e agilidade) do Gradle vem de sua concepção baseada no emprego de Grados Diretos Acíclicos para as tarefas. Este grafo é baseado nas dependências apresentadas entre as tarefas de construção, e após gerado torna mais fácil para a ferramenta determinar a melhor ordem para execução das tarefas, o que ocorre após essa determinação. Essa é uma maneira bem agnóstica de representar dependências e as sequências envolvidas na construção.



Fonte: Adaptado de docs.gradle.org

Em geral, as maiores reclamações a respeito do Gradle giram ao redor da necessidade em se ter maiores habilidades técnicas para se conseguir operá-lo. Outro ponto é que a comunidade que suporta o Maven é maior do que a que suporta o Gradle. Assim, desenvolvedores usando Gradle tendem a levar mais tempo para conseguirem resolução de dúvidas e problemas.

Atenção! Aqui existe uma videoaula, acesso pelo conteúdo online

Atividade

1. A Gestão de Construção não inclui:
- a) Coleta de itens de configuração.
 - b) Compilação.
 - c) União de ativos.
 - d) Orquestração.
 - e) Implantação.

2. Não é uma vantagem advinda do uso de dependências:

- a) Melhor performance.
 - b) Reuso.
 - c) Criação de frameworks.
 - d) Maior segurança.
 - e) Redução de defeitos.
-

3. A automação proporcionada pela Gestão da Construção:

- a) Não tem a capacidade de substituir plenamente a construção manual.
 - b) Tem capacidade de substituir plenamente a construção manual.
 - c) Deve mandatoriamente substituir plenamente a construção manual.
 - d) Não deve jamais substituir plenamente a construção manual.
 - e) Não tem ligação com a construção manual.
-

4. A ferramenta Gradle funciona:

- a) Com base no Linux.
 - b) Rodando sobre uma Sandbox.
 - c) Rodando sobre uma JVM.
 - d) Com base em Python.
 - e) Com base em C#.
-

Notas

Mudança emergencial¹

As mudanças emergenciais são basicamente o oposto das mudanças padrão: Geralmente, mais do que se caracterizarem por trazerem riscos médios a altos, exigem implementação o mais rapidamente possível, de forma que também possam ser operacionalizadas com grande rapidez. São, portanto, mudanças para “situações de crise”, e que requerem procedimentos à altura.

Referências

AIELLO, B. **Configuration Management Best Practices**. 1.ed. Pearson, 2013.

BOURQUE, P.; FAIRLEY, R. **Software Engineering Body of Knowledge (SWEBOK)**. 3.ed. IEEE Computer Society, 2017.

HAAS, J. **Configuration Management Principles and Practice**. 1.ed. Addison Wesley, 2003.

KLEIN, H. **Gradle Dependency Management**. 1.ed. PACKT Publishing, 2015.

SIRIWARDENA, P. **Maven Essentials Get Started With The Essentials of Apache Maven and you're your build your automation system up and running quickly**. 1.ed. PACKT Publishing, 2015. Acesso em 14 nov. 2020.

Dataverse Project. Disponível em <http://guides.dataverse.org/en/latest/developers/dependencies.html>. Acesso em 14 nov.

2020.

Próxima aula

- Conceituação e tipos de liberação;
- Boas práticas da Gestão de Liberação;
- Software / Ferramentas de Gerenciamento de Liberação.

Explore mais

Visite o site DevMedia e leia sobre a automação do processo de construção.