
Apache Hop

Fernando Anselmo

<http://fernandoanselmo.orgfree.com/wordpress/>

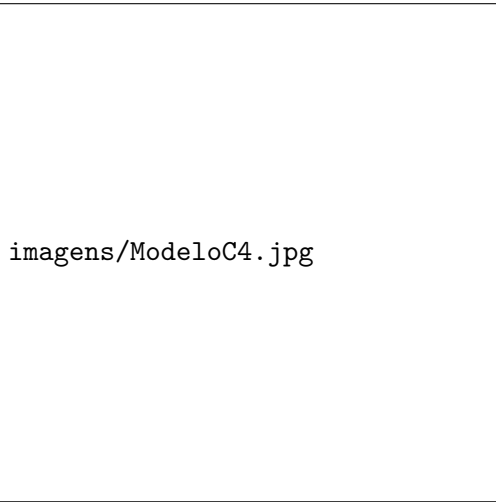
Versão 1.0 em 5 de junho de 2025

Resumo

Apache Hop (*Hop Orchestration Platform*) é uma plataforma moderna de código aberto para engenharia e orquestração de dados, projetada para tornar os processos de integração de dados mais acessíveis, flexíveis e reutilizáveis. Desenvolvido com foco em produtividade e usabilidade, o **Apache Hop** permite criar pipelines e *workflows* de dados de forma visual e intuitiva, sem depender necessariamente de programação.

1 CMS - Liferay DXP

Por muitos tempo, arquitetura significou projetar e construir estruturas físicas, como casas e edifícios. Mas, nas últimas décadas, os termos arquitetura e arquiteto passaram a serem utilizados também no mundo da tecnologia e do desenvolvimento de software.



imagens/ModeloC4.jpg

Figura 1: *Exemplo do Modelo C4*

Uma das grandes vantagens do **Apache Hop** é sua capacidade de projetar *pipelines* uma única vez e executá-los em diferentes ambientes — locais, em nuvem, ou em frameworks como **Apache Spark**, **Apache Flink** ou **Google Dataflow** - por meio das chamadas configurações de tempo de execução, um tipo especial de metadado que abstrai o ambiente de execução.

Além disso, o **Apache Hop** centraliza os processos em uma plataforma única e gerenciável, oferece recursos avançados de controle de qualidade, persistência e rastreabilidade dos dados, contribuindo para a confiabilidade e a governança da informação.

A plataforma é desenvolvida por uma comunidade aberta, colaborativa e acolhedora, sob a governança da **Apache Software Foundation**. Todos são convidados a participar: seja tirar dúvidas, relatar problemas, propor novos recursos, contribuir com código ou documentação, auxiliar nos testes de versões ou melhorar o site oficial.

Apache Hop é uma solução robusta, extensível e preparada para os desafios modernos da engenharia de dados, ideal para organizações que buscam eficiência, automação e escalabilidade em seus fluxos de dados.

2 Apache Hop e Pentaho

Apache Hop é uma poderosa ferramenta de integração e orquestração de dados de código aberto, criada como um fork evoluído do **Pentaho Data Integration (PDI)**, também conhecido como Kettle. Embora compartilhe raízes com o PDI, **Apache Hop** foi totalmente reestruturado e modernizado para atender às necessidades atuais de engenharia de dados com mais desempenho, modularidade e escalabilidade.

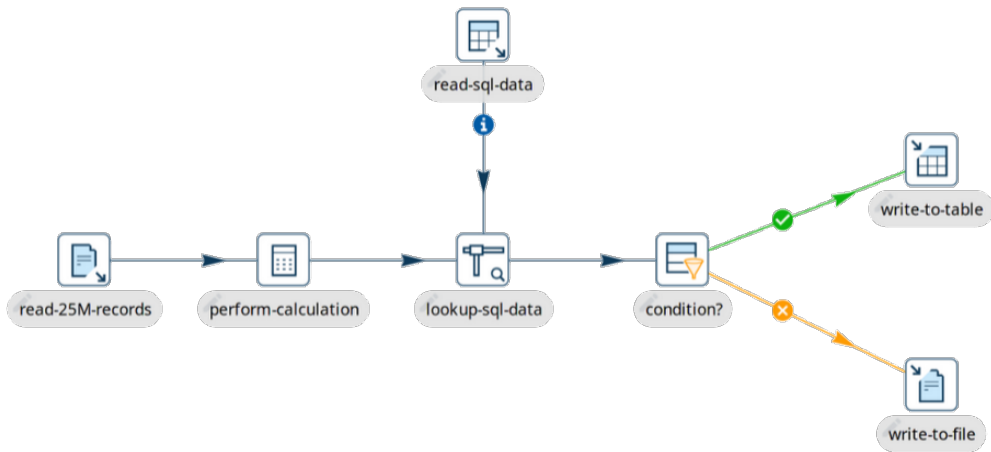


Figura 2: Exemplo de uma Pipeline no Apache Hop

Com uma interface de desenvolvimento visual, **Apache Hop** permite que Engenheiros e Arquitetos de Dados construam *pipelines* e *workflows* complexos de forma intuitiva, sem a necessidade de escrever código, embora isso continue sendo uma opção para os usuários mais avançados. Essa abordagem visual acelera o desenvolvimento, reduz erros e facilita a colaboração entre equipes técnicas e de negócio.

Além disso, **Apache Hop** introduz conceitos modernos, como metadados reutilizáveis, configurações de tempo de execução e suporte a múltiplos motores, tornando-se uma solução versátil para projetos locais, em nuvem ou em ambientes distribuídos.

Comparativo: Apache Hop vs Pentaho Data Integration (PDI)

Característica	Apache Hop	Pentaho Data Integration (PDI)
----------------	------------	--------------------------------

Origem	Fork moderno e reescrito do PDI/Kettle	Projeto original, mantido pela Hitachi Vantara
Licença	Apache License 2.0 (open source completo)	Community Edition: LGPL Enterprise Edition: Proprietária
Governo do Projeto	Apache Software Foundation (comunidade aberta)	Hitachi Vantara (foco comercial)
Interface de Desenvolvimento	Visual (Hop GUI)	Visual (Spoon)
Modularidade	Altamente modular, orientado a plugins	Arquitetura mais monolítica
Abordagem Baseada em Metadados	Sim – pipelines, workflows, variáveis, ambientes	Parcialmente, com menos flexibilidade
Configurações de Execução	Suporte a múltiplos ambientes com "run configurations"	Limitado, dependente de configurações locais
Execução Distribuída	Suporte nativo a Spark, Flink, Beam via plugins	Requer customizações adicionais
Linha de Comando	Ferramentas modernas: <code>hop-run</code> , <code>hop-gui</code> , <code>hop-server</code>	Ferramentas legadas: <code>pan</code> , <code>kitchen</code>
Comunidade	Ativa, aberta, com crescimento contínuo	Reduzida na versão open source
Atualizações e Roadmap	Frequentes e transparentes	Lentamente atualizada na versão gratuita
Documentação	Completa e mantida pela comunidade	Limitada na versão open source

As principais vantagens do **Apache Hop** são:

Integração nativa com GIT - Não é necessário usar clientes GIT de terceiros para tornar o ambiente DevOps e DataOps mais amigável e produtivo. Existe uma interface visual no **Apache Hop** que permite ver tudo que foi alterado, inclusive mostrando graficamente o *pipeline* ou *workflow* editado. Com certeza este é um grande avanço se comparado a todos os tipos de repositórios de artefatos (transformations e jobs) do *Pentaho Community Edition*.

Velocidade quanto a atualizações - **Pentaho** e **Apache Hop** atualmente tomam rumos diferentes, possuem objetivos diferentes e portanto têm suas atualizações seguindo por caminhos diferentes. quando há uma necessidade da comunidade sobre a atualização de uma *transform* do **Apache Hop** (equivalente ao *step* do **Pentaho**), isso acontece em uma maior velocidade.

Projeto Top Level da Apache Software Foundation - *Apache Software Foundation* é uma organização sem fins lucrativos criada para suportar os projetos de código aberto. Ser um projeto da Apache requer que o Software preencha uma série de requisitos, o que dá grande credibilidade e robustez ao projeto.

3 Componentes do Apache Hop

Apache Hop possui três componentes principais, são eles:

Hop GUI é a interface gráfica principal do **Apache Hop**, projetada para facilitar o desenvolvimento de pipelines (antigas transformações no PDI) e *workflows* (antigos jobs). Com uma abordagem visual e intuitiva. Elimina a necessidade de codificação ao permitir que criemos fluxos complexos de ETL (Extração, Transformação e Carga) por meio de elementos de arrastar e soltar (*drag-and-drop*). Cada *pipeline* representa uma sequência lógica de transformações de dados, enquanto *workflows* permitem orquestrar múltiplas tarefas e *pipelines* em uma ordem específica, com controle de fluxo, paralelismo e dependências. O ambiente também oferece recursos de debug, execução local, parametrização, controle de versão, validação e reutilização de metadados, o que torna o desenvolvimento altamente produtivo e sustentável.

Hop Run é uma ferramenta de linha de comando (CLI) autônoma usada para executar pipelines e *workflows* fora do ambiente gráfico. Ideal para integrações com *scripts*, automações de DevOps, servidores CI/CD ou agendamentos via **cron** (agendamento). Permite a execução headless (sem interface gráfica) com suporte completo a parâmetros, ambientes e variáveis definidos no projeto. Garante que pipelines criados visualmente possam ser facilmente executados em produção, ambientes de teste ou *containers*, promovendo flexibilidade e consistência no ciclo de vida das soluções de dados.

Hop Server é um servidor leve baseado em web, capaz de executar remotamente *pipelines* e *workflows* em ambientes distribuídos. Pode ser implantado em um ou mais nós, permite a execução paralela, balanceamento de carga e alta disponibilidade. Expõe uma API RESTful completa, possibilitando que outros sistemas ou aplicações interajam com os *pipelines* de forma programática — ideal para automações, integrações com plataformas externas e arquiteturas orientadas a eventos. Através dele, é possível orquestrar fluxos de dados complexos a partir de múltiplos servidores, promover escalabilidade horizontal e melhorar o gerenciamento das cargas de trabalho.

4 E isso tudo em contêineres do Docker

Docker, como plataforma de contêineres, oferece uma maneira rápida e flexível de empacotar, distribuir e executar aplicações em ambientes isolados. Ao utilizar Docker para orquestrar a integração entre Liferay DXP e Alfresco, as empresas podem garantir uma configuração mais ágil, consistência nos ambientes de desenvolvimento, testes e produção, além de facilitar o escalonamento das aplicações conforme necessário.

Uma das principais vantagens de usar Docker nessa integração é a simplificação do processo de deployment. Com o Docker, tanto o Liferay DXP quanto o Alfresco podem ser configurados e executados em contêineres isolados, o que facilita o gerenciamento de dependências e versões. Isso elimina conflitos de ambiente que poderiam ocorrer ao rodar as aplicações em servidores tradicionais. Além disso, a utilização de contêineres permite que cada ferramenta seja executada com a sua própria configuração e requisitos de infraestrutura, enquanto ainda mantém a comunicação entre elas de forma eficaz e segura. Isso reduz o risco de erros e aumenta a previsibilidade no desenvolvimento.

Outro benefício é a flexibilidade de escalabilidade. Quando uma das ferramentas, como o Alfresco, precisar de mais recursos de processamento ou armazenamento, a solução pode ser escalada facilmente sem impactar o funcionamento do Liferay DXP, graças à natureza isolada dos contêineres. Docker também facilita o gerenciamento de clusters e a replicação de ambientes em diferentes servidores ou

nuvens, permite uma melhor utilização dos recursos e garante a tão sonhada "alta disponibilidade".

4.1 Instalar o Liferay no Docker

Instalar o Liferay a partir do Docker é um dos trabalhos mais fáceis que podem ser realizados por um desenvolvedor, talvez trocar a lâmpada seja mais fácil. Com o simples comando:

```
$ docker run -it -d -m 4g -p 8081:8080 --name=meu-liferay -e JAVA_VERSION=zulu21 -v /home/[usuario]/liferaymnt:/mnt/liferay liferay/portal:latest
```

Basicamente, definimos um contêiner que utiliza 4 gigabytes de memória, que será executado na porta **8081** (pois a porta 8080 será do Alfresco), que executará o Java 21 e se chamará meu-liferay.

Para interromper o contêiner:

```
$ docker stop meu-liferay
```

Para reiniciar o contêiner:

```
$ docker start meu-liferay
```

Testemos para ver se funciona corretamente, no navegador acessar o endereço: <http://localhost:8081>, o usuário padrão é **test@liferay.com**, e a senha: **test**.

ATENÇÃO: Na primeira vez que se logar será forçado para mudar a senha, troque-a para **admin** (toda em minúsculas), uma vez que entrou corretamente com a nova senha, no perfil em *User Profile* devemos modificar o Screen Name de **test** para **admin** (também toda em minúsculas), salvar e confirmar a mudança com a senha.

4.2 Instalar o Alfresco no Docker

Subir um Alfresco não é tão simples como o Liferay pois são diversos componentes incluindo o Banco Postgres, Apache Solr e o servidor de mensagens ActiveMQ. Assim o melhor modo é colocá-lo sobre o Docker Compose de modo a gerenciar toda essa informação e trabalhar de modo unido.

Faça um clone do repositório oficial da comunidade no Git:

```
$ git clone https://github.com/Alfresco/acs-deployment.git
```

Entre no diretório do Docker Compose:

```
$ cd acs-deployment/docker-compose
```

Crie e suba os contêineres necessários:

```
$ docker compose -f community-docker-compose.yaml up
```

Para interromper os contêineres:

```
$ docker compose -f community-docker-compose.yaml stop
```

Para reiniciá-los:

```
$ docker compose -f community-docker-compose.yaml start
```

Testemos para ver se funciona corretamente, no navegador acessar o endereço: <http://localhost:8080/alfresco>, o usuário padrão é **admin**, e a senha: **admin**.

Outros endereços úteis são:

- Control Center `http://localhost:8080/admin`
- Share `http://localhost:8080/share`
- Alfresco Content App `http://localhost:8080/content-app`
- Search Services administration `http://localhost:8083/solr`

Lembre-se sempre de estar, obrigatoriamente, neste diretório para realizar essas operações.

Outro passo extremamente importante é colocar o Liferay na mesma rede que o Alfresco. Primeiro vamos verificar as redes:

```
$ docker network ls
```

Provavelmente deve existir uma rede chamada `docker-compose_default`, vamos associar o contêiner do Liferay a esta:

```
$ docker network docker-compose_default meu-liferay
```

5 Configurar a autenticação do Liferay

O primeiro passo que devemos proceder para que tudo funcione corretamente é que o usuário administrador do **Liferay** deve ser o mesmo do **Alfresco**, pois dessa forma não realizamos autenticações separadamente, por isso ao término da instalação do **Liferay** garantimos que o usuário e sua senha são **admin** (que corresponde ao padrão contidos no **Alfresco**). Agora devemos mudar a forma como o usuário do **Liferay** se autentica que é por **e-mail**, enquanto que o **Alfresco** realiza esse processo por **nome de usuário**.

Aqui temos um "pulo do gato", o contêiner tanto do **Liferay** quanto do **Alfresco** não possuem nenhum editor instalado, ou mesmo, a possibilidade em se instalar um, então os arquivos devem ser criados na máquina local e em seguida copiados para o contêiner nos locais indicados.

Devemos criar um arquivo local com o seguinte conteúdo:

```
1 layout.show.portlet.access.denied=true
2 session.store.password=true
3 company.security.auth.type=screenName
4 web.server.host=<IP-DA-MAQUINA>
```

Na propriedade **web.server.host** devemos colocar o endereço IP da máquina onde o contêiner do **Liferay** está executando.

Este arquivo deve ser salvo com o nome **portal-ext.properties**. Próximo passo é copiá-lo para o contêiner do Liferay:

```
$ docker cp portal-ext.properties meu-liferay:/opt/liferay/tomcat/webapps/ROOT/WEB-INF/classes
```

Este comando deve retornar a mensagem:

```
Successfully copied 2.05kB to meu-liferay:/opt/liferay/tomcat/webapps/ROOT/WEB-INF/classes
```

Por fim, devemos reiniciar o contêiner do Liferay:

```
$ docker restart meu-liferay
```

E uma vez ativo, devemos conseguir logar com o *UserName* **admin** e mesma senha. E se tudo está funcionando corretamente o Liferay deve ser acessível tanto pelo endereço **http://localhost:8081** quanto por **http://[IP-MAQUINA]:8081**, isso é importante quando desejamos usar duas máquinas na mesma rede.

6 Preparar o Alfresco

A comunicação pode ser realizada de duas formas, a primeira é por **Web Scripts** isso significa escrevê-los e disponibilizá-los como funcionalidades, por exemplo, desejamos ter uma simples listagem dos documentos disponíveis, ou uma pesquisa personalizada, veremos essa forma em breve. A segunda é por meio de um repositório de interligação através da comunicação **CMIS**.

Podemos dizer que esse é um modelo de integração total, onde o usuário do **Liferay** teria permissão de realizar qualquer processo dentro do **Alfresco**.

Devemos criar um arquivo com o seguinte conteúdo:

```
1 cmis.enabled=true
```

Este arquivo deve ser salvo com o nome **alfresco-global.properties**. Próximo passo é copiá-lo para o contêiner do **Alfresco**:

```
$ docker cp alfresco-global docker-compose-alfresco-1:/usr/local/tomcat/shared  
/classes
```

7 Conclusão

Em diferentes cenários, *Digital Experience Platforms* como o **Liferay DXP** (algum tempo atrás conhecido como *Enterprise Portals*), e um *Content Services Platforms* como o **Alfresco** convivendo juntos e gerenciando partes do ciclo da vida de documentos. A integração via portlet providencia uma solução para o publicador de conteúdo via **CMIS API**.

Ao utilizar **Docker** para integrar **Liferay** e **Alfresco**, as organizações podem não apenas melhorar a eficiência operacional, mas também garantir uma arquitetura mais resiliente, fácil de manter e expandir à medida que as necessidades do negócio evoluem.

Sou um entusiasta do mundo **Open Source** e novas tecnologias. Qual a diferença entre Livre e Open Source? Livre significa que esta apostila é gratuita e pode ser compartilhada a vontade. Open Source além de livre todos os arquivos que permitem a geração desta (chamados de arquivos fontes) devem ser disponibilizados para que qualquer pessoa possa modificar ao seu prazer, gerar novas, complementar ou fazer o que quiser. Os fontes da apostila (que foi produzida com o LaTeX) está disponibilizado no GitHub [5]. Veja ainda outros artigos que publico sobre tecnologia através do meu Blog Oficial [3].

Referências

- [1] Site oficial do Liferay
<https://www.liferay.com/>
- [2] Site oficial do Alfresco Community Edition
<https://docs.alfresco.com/content-services/community/>
- [3] Fernando Anselmo - Blog Oficial de Tecnologia
<http://www.fernandoanselmo.blogspot.com.br/>
- [4] Encontre essa e outras publicações em
<https://cetrex.academia.edu/FernandoAnselmo>
- [5] Repositório para os fontes da apostila
<https://github.com/fernandoans/publicacoes>