



Digital
College

FORMAÇÃO EM

DATA ANALYTICS

MÓDULO 2:

**EXTRAÇÃO, TRANSFORMAÇÃO
E CARGA (ETL)**

UNIDADE 4:

PIPELINE DE DADOS



Sumário

1. Introdução	04
2. Arquitetura de um Pipeline de Dados	05
2.1. Camada de Ingestão de Dados	06
2.2. Camada de Armazenamento	08
2.3. Camada de Processamento	11
2.4. Camada de Análise	13
2.5. Camada de visualização	15
3. Pentaho Data Integration e Pipeline de Dados	16
3.1. JOBS no Pentaho Data Integration	17
3.2. Principais Steps utilizados nos JOBS	18
4. Pentaho Data Integration - JOBS	19
4.1. Fluxo 1	19
4.2. Fluxo 2	26
4.3. Fluxo 3	28
5. Conceitos do Pentaho Server	30
5.1. Arquitetura do Pentaho Server	31
5.2. Instalação e Configuração do Pentaho Server	33
6. Apache Airflow: maestro de pipelines de tarefas agendadas	35
6.1. O que o Airflow é?	35
6.2. O que o Airflow não é	38

1. Introdução

Um pipeline de dados é uma série de processos que são executados em conjunto para mover, processar e transformar dados de uma fonte para um destino específico. Ele é comumente utilizado em ambientes de big data, onde há uma grande quantidade de dados que precisam ser processados e analisados.

O pipeline de dados começa com a coleta de dados brutos de várias fontes, como bancos de dados, arquivos, dispositivos IoT, sensores etc. Em seguida, os dados passam por um processo de limpeza e pré-processamento, que inclui remoção de dados duplicados, correção de erros, preenchimento de valores ausentes e padronização de formatos.

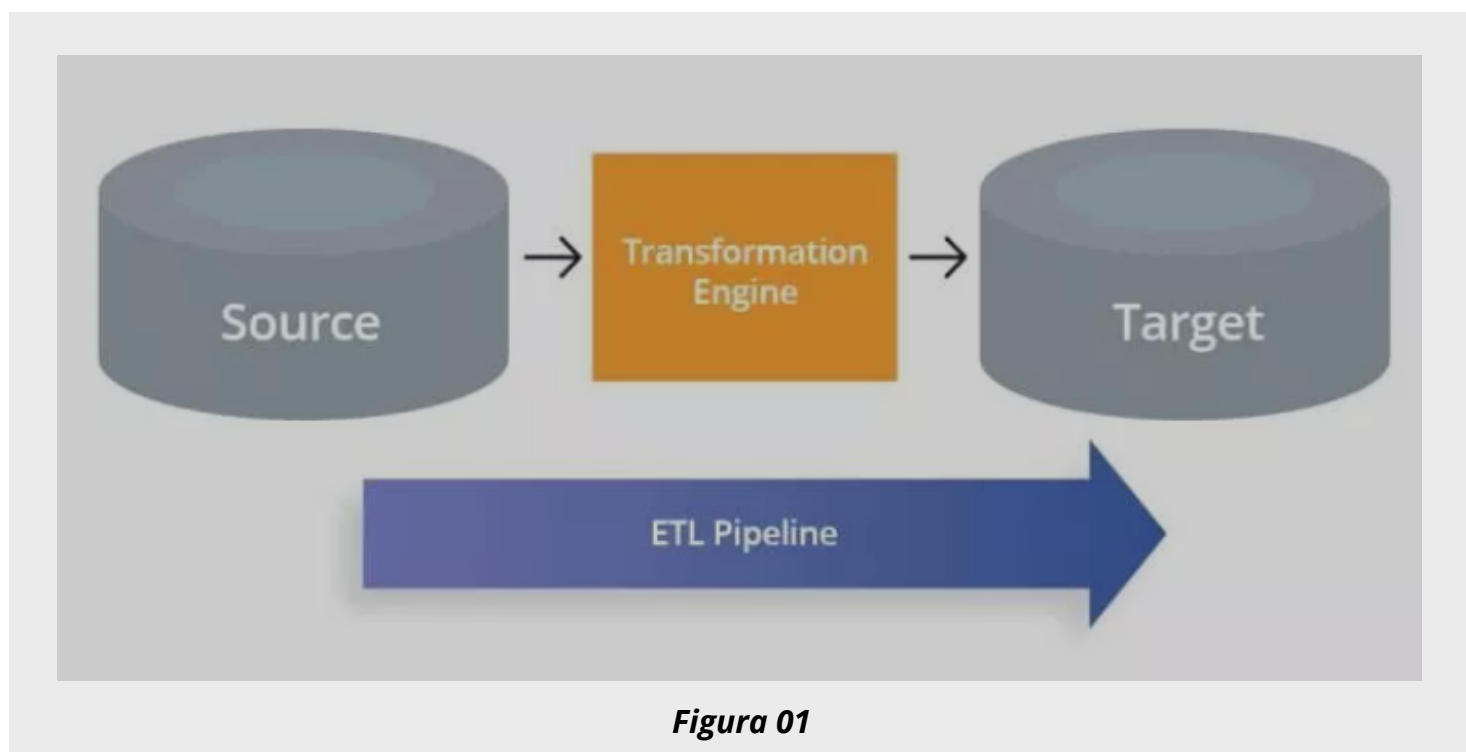
O próximo passo é a transformação dos dados, que envolve aplicação de regras de negócios e técnicas de análise para extrair insights significativos dos dados. A transformação também pode incluir agregação e junção de dados de várias fontes.

Em seguida, os dados são armazenados em um formato acessível para análise futura, que pode incluir bancos de dados, data warehouses ou data lakes.

Finalmente, os dados são disponibilizados para uso, o que pode incluir a criação de visualizações, relatórios ou dashboards para visualização de informações. A pipeline de dados é geralmente automatizada para garantir a consistência e a eficiência do processo.

Em resumo, o pipeline de dados é uma abordagem sistemática para gerenciar dados, desde a coleta até a análise e visualização. Ele é essencial para a tomada de decisões baseada em dados e é usada em uma variedade de setores, incluindo finanças, saúde, varejo e tecnologia.

Nesse material iremos utilizar a ferramenta Pentaho Data Integration para demonstrar a criação de pipeline de dados.



2. Arquitetura de um Pipeline de Dados

Um pipeline de dados é um processo composto por uma série de etapas interconectadas que permitem a coleta, o armazenamento, o processamento e a análise de dados em uma organização. A arquitetura de um pipeline de dados pode variar de acordo com as necessidades específicas da organização, mas geralmente envolve as seguintes etapas:

1. **Coleta de dados:** nesta etapa, os dados são coletados de diversas fontes, como bancos de dados, arquivos, sensores, dispositivos IoT, redes sociais, entre outros;
2. **Armazenamento de dados:** os dados coletados são armazenados em um local centralizado para que possam ser acessados e processados facilmente. O armazenamento de dados pode ser feito em bancos de dados tradicionais, sistemas de arquivos distribuídos ou armazenamento em nuvem;
3. **Processamento de dados:** nesta etapa, os dados são limpos, transformados e agregados para torná-los úteis para análise. As ferramentas de processamento de dados incluem SQL, Spark, Hadoop, entre outros;
4. **Análise de dados:** os dados processados são analisados para obter insights e tomar decisões informadas. As ferramentas de análise de dados incluem estatísticas, aprendizado de máquina, inteligência artificial e visualização de dados;
5. **Visualização de dados:** os dados analisados são apresentados de forma visualmente atraente para que possam ser facilmente compreendidos pelos usuários finais. As ferramentas de visualização de dados incluem gráficos, tabelas, dashboards, entre outros.

A arquitetura de um pipeline de dados pode ser dividida em camadas, com cada camada representando uma etapa específica do processo. Uma arquitetura típica de pipeline de dados inclui as seguintes camadas:

- **Camada de ingestão:** responsável pela coleta de dados de diferentes fontes;
- **Camada de armazenamento:** responsável pelo armazenamento de dados brutos e dados processados;
- **Camada de processamento:** responsável pela transformação de dados brutos em dados processados e prontos para análise;
- **Camada de análise:** responsável pela análise de dados para obter insights;
- **Camada de visualização:** responsável pela apresentação visual dos dados para os usuários finais.

Cada camada pode ser implementada usando diferentes tecnologias, dependendo dos requisitos específicos do pipeline de dados. Por exemplo, a camada de ingestão pode usar tecnologias como Apache Kafka ou Apache NiFi, enquanto a camada de processamento pode usar tecnologias como Apache Spark ou Hadoop. A escolha das tecnologias depende das necessidades da organização em termos de escalabilidade, velocidade, flexibilidade e custo.

2.1. Camada de Ingestão de Dados

A camada de ingestão de dados em um pipeline é responsável pela coleta de dados de várias fontes (como bancos de dados, arquivos, sensores, dispositivos IoT, redes sociais, entre outros), e por sua transferência para a camada de armazenamento de dados. É a primeira etapa do pipeline de dados e, portanto, é crítica para garantir a qualidade dos dados e a integridade destes.

A camada de ingestão de dados pode incluir ferramentas de ETL (Extract, Transform, Load) que permitem a extração de dados de diferentes fontes, transformação dos dados para que se adequem ao formato esperado, e carga dos dados em um local de armazenamento centralizado. As ferramentas de ETL também podem incluir recursos para validação e limpeza dos dados para garantir que os dados coletados sejam precisos, completos e livres de erros.

Além disso, a camada de ingestão de dados pode envolver o uso de tecnologias de streaming para lidar com a ingestão de dados em tempo real, como o Apache Kafka ou Apache NiFi. As tecnologias de streaming permitem a ingestão contínua de dados em um pipeline de dados, garantindo que os dados sejam processados e analisados o mais rápido possível.

A camada de ingestão de dados é crítica para o sucesso de um pipeline de dados e deve ser projetada de forma cuidadosa, levando em consideração os requisitos específicos da organização, como escalabilidade, disponibilidade, segurança e conformidade. A escolha das ferramentas e tecnologias para a camada de ingestão de dados deve ser feita cuidadosamente, para garantir que a coleta de dados seja realizada de maneira eficiente e eficaz.

Existem diversas ferramentas e tecnologias disponíveis para a camada de ingestão de dados em um pipeline de dados. Abaixo, listo alguns exemplos de ferramentas comumente utilizadas para essa camada:

- **Apache Kafka:** é uma plataforma de streaming distribuída que permite a ingestão de dados em tempo real. Ele pode lidar com grandes volumes de dados e suporta integração com diversas fontes de dados. O Kafka é usado em muitas arquiteturas de pipeline de dados para coleta de dados em tempo real;

- **Apache NiFi:** é uma ferramenta de processamento de fluxo de dados que permite a coleta, a agregação e a transformação de dados em tempo real. Ela tem uma interface visual intuitiva e é fácil de usar, tornando-a uma opção popular para a camada de ingestão de dados;
- **AWS Glue:** é um serviço gerenciado de ETL (Extract, Transform, Load) que pode ser usado para a ingestão de dados em um pipeline de dados. Ele pode ser integrado com fontes de dados em nuvem e on-premise, e permite a transformação de dados usando linguagens de programação como Python e Scala;
- **Microsoft Azure Data Factory:** é um serviço gerenciado de ETL na nuvem que permite a ingestão de dados de várias fontes de dados, incluindo dados locais, nuvem e de dispositivos IoT. Ele oferece uma interface visual para criar e gerenciar fluxos de trabalho de ingestão de dados;
- **Google Cloud Dataflow:** é um serviço gerenciado de processamento de dados que permite a ingestão desses em tempo real e em lote. Ele pode ser integrado com várias fontes e permite a transformação de dados usando linguagens de programação como Java e Python.

Esses são apenas alguns exemplos de ferramentas que podem ser usadas na camada de ingestão de dados. A escolha da ferramenta ou tecnologia depende dos requisitos específicos do pipeline de dados e das necessidades da organização em termos de escalabilidade, segurança, custo e disponibilidade.

2.2. Camada de Armazenamento

A camada de armazenamento em um pipeline de dados é responsável por armazenar os dados coletados e processados durante a execução do pipeline. Esses dados são armazenados em um local centralizado para que possam ser acessados e analisados posteriormente.

A camada de armazenamento pode incluir vários tipos de tecnologias de armazenamento de dados, dependendo dos requisitos específicos do pipeline de dados. Algumas das tecnologias de armazenamento de dados mais comuns incluem bancos de dados relacionais, bancos de dados NoSQL, data lakes, armazenamento em nuvem e sistemas de arquivos distribuídos.

Os bancos de dados relacionais são usados quando é necessário manter a consistência dos dados, e geralmente são utilizados em pipelines de dados que envolvem transações financeiras ou outros tipos de processamento transacional. Os bancos de dados NoSQL são usados quando a escalabilidade é um requisito crítico, e podem ser utilizados em pipelines de dados que envolvem grandes volumes de dados.

Os data lakes são usados para armazenar grandes volumes de dados não estruturados ou semiestruturados, permitindo que os dados sejam armazenados em sua forma bruta e posteriormente transformados para fins analíticos. O armazenamento em nuvem permite a escalabilidade, disponibilidade e flexibilidade necessárias para muitos pipelines de dados modernos.

Os sistemas de arquivos distribuídos, como o Hadoop Distributed File System (HDFS), são usados para armazenar grandes volumes de dados em um cluster de computadores distribuídos, permitindo a escalabilidade horizontal e a alta disponibilidade.

A escolha da tecnologia de armazenamento de dados depende dos requisitos específicos do pipeline de dados, incluindo o volume de dados a ser armazenado, o tipo de dados, a frequência de acesso aos dados, a necessidade de escalabilidade e a segurança desses dados.

A camada de armazenamento é uma parte crítica do pipeline de dados, pois permite que os dados sejam armazenados e acessados posteriormente para fins analíticos e de geração de relatórios. Uma camada de armazenamento bem projetada e configurada é essencial para garantir a qualidade e a integridade dos dados armazenados.

Existem várias tecnologias e ferramentas de armazenamento de dados disponíveis para serem usadas em uma camada de armazenamento de um pipeline de dados. Abaixo estão alguns exemplos:

- **Bancos de dados relacionais:** como o MySQL, PostgreSQL e Oracle, são frequentemente usados em pipelines de dados que exigem consistência de dados e suporte a transações, como transações financeiras;
- **Bancos de dados NoSQL:** como o MongoDB, Cassandra e DynamoDB, são usados em pipelines de dados que requerem escalabilidade horizontal, processamento de grandes volumes de dados e gerenciamento de dados não estruturados;
- **Data lakes:** como o Amazon S3, Microsoft Azure Data Lake Storage e Google Cloud Storage, são usados para armazenar grandes volumes de dados não estruturados e semiestruturados, permitindo a análise posterior dos dados;
- **Armazenamento em nuvem:** como o Amazon RDS, Microsoft Azure SQL Database e Google Cloud SQL, oferecem uma maneira escalável e segura de armazenar dados relacionais na nuvem;

- **Sistemas de arquivos distribuídos:** como o Hadoop Distributed File System (HDFS), são usados para armazenar grandes volumes de dados em clusters de computadores distribuídos, permitindo a escalabilidade horizontal e a alta disponibilidade;
- **Armazenamento em memória:** como o Redis e Memcached, são usados para armazenar dados em cache em memória para acesso rápido, frequentemente usado em pipelines de dados em tempo real;
- **Armazenamento de objeto:** como o Amazon S3 e Google Cloud Storage, são usados para armazenar objetos e arquivos em um formato de objeto para posterior análise.

A escolha da tecnologia de armazenamento depende dos requisitos específicos do pipeline de dados, incluindo o volume de dados a ser armazenado, o tipo de dados, a frequência de acesso aos dados, a necessidade de escalabilidade e a segurança dos dados. É importante projetar a camada de armazenamento adequadamente para garantir que os dados sejam armazenados de maneira eficiente e segura para posterior análise.

2.3. Camada de Processamento

A camada de processamento de dados em um pipeline é responsável por realizar a transformação e análise dos dados coletados. Essa camada inclui várias etapas, como limpeza e pré-processamento, agregação e transformação de dados, análise estatística e aprendizado de máquina.

A camada de processamento de dados é crucial para o sucesso de um pipeline, pois é nessa etapa que os dados brutos são transformados em informações úteis e acionáveis. A camada de processamento de dados pode incluir várias ferramentas e tecnologias, dependendo dos requisitos específicos do pipeline de dados e do tipo de análise que será realizada.

Algumas das tecnologias mais comuns usadas na camada de processamento de dados incluem linguagens de programação como Python, R e Java, bem como frameworks de processamento de dados, como Apache Spark e Apache Flink. Além disso, ferramentas de análise, como Tableau, Power BI e Google Data Studio, também podem ser usadas na camada de processamento de dados para criar visualizações e relatórios a partir dos dados.

A camada de processamento de dados é responsável por garantir que os dados sejam transformados em informações úteis e acionáveis, que possam ser usadas para tomar decisões de negócios e para identificar padrões e tendências. É importante que essa camada seja projetada adequadamente para garantir a qualidade dos dados e a precisão das análises realizadas.

A camada de processamento de dados em um pipeline de dados pode incluir várias tecnologias e ferramentas, dependendo dos requisitos específicos do pipeline de dados e do tipo de análise que será realizada. Abaixo estão alguns exemplos comuns de tecnologias e ferramentas usadas na camada de processamento de dados:

- **Linguagens de programação:** Python, R, Java e SQL são frequentemente usados na camada de processamento de dados para realizar operações de limpeza e transformação de dados;
- **Frameworks de processamento de dados:** Apache Spark, Apache Flink e Apache Beam são exemplos de frameworks que permitem o processamento de grandes volumes de dados em tempo real ou em lote;
- **Ferramentas de aprendizado de máquina:** TensorFlow, Scikit-learn e PyTorch são exemplos de bibliotecas de aprendizado de máquina que podem ser usadas para treinar modelos de machine learning em dados;

- **Ferramentas de análise de dados:** Tableau, Power BI, Google Data Studio e Excel são exemplos de ferramentas que podem ser usadas para criar visualizações e relatórios a partir dos dados;
- **Bibliotecas de processamento de linguagem natural (NLP):** NLTK, SpaCy e Gensim são exemplos de bibliotecas que podem ser usadas para processar e analisar texto em linguagem natural;
- **Sistemas de gerenciamento de fluxo de trabalho:** Apache Airflow e Luigi são exemplos de sistemas que podem ser usados para gerenciar fluxos de trabalho complexos de processamento de dados;
- **Ferramentas de integração de dados:** Talend e Apache NiFi são exemplos de ferramentas que podem ser usadas para integrar e transformar dados de várias fontes.

A escolha das tecnologias e ferramentas na camada de processamento de dados dependerá dos requisitos específicos do pipeline de dados e do tipo de análise que será realizada. É importante projetar essa camada adequadamente para garantir que as análises sejam precisas e que os resultados sejam úteis e acionáveis.

2.4. Camada de Análise

A camada de análise de dados é uma parte do processo de análise de dados que envolve a utilização de técnicas e ferramentas para extrair insights úteis e significativos a partir dos dados coletados. Essa camada é crucial para a tomada de decisões baseadas em dados e para a identificação de tendências, padrões e correlações nos dados.



A camada de análise de dados envolve várias etapas, como a limpeza e preparação dos dados, a exploração dos dados para identificar tendências e padrões, a criação de modelos analíticos para prever resultados ou identificar riscos, e a apresentação dos resultados de forma clara e acessível para os usuários finais.

As técnicas utilizadas na camada de análise de dados incluem estatística descritiva, mineração de dados, aprendizado de máquina e análise de séries temporais. As ferramentas utilizadas podem incluir planilhas, bancos de dados, softwares de visualização de dados e plataformas de análise de dados avançadas. Um exemplo de camada de análise de dados pode ser a análise de dados de vendas de uma empresa de comércio eletrônico. Suponha que a empresa coleta dados sobre as vendas diárias, o número de visitantes do site, o tempo de permanência no site, o tipo de produto comprado e o valor total das vendas.

A camada de análise de dados seria responsável por extrair insights úteis a partir desses dados. Por exemplo, a empresa poderia explorar os dados para identificar tendências e padrões, como quais produtos vendem mais em determinados dias da semana ou horários do dia.

A camada de análise de dados também poderia criar modelos analíticos para prever resultados, como prever as vendas futuras com base nas tendências históricas ou prever o valor de vida do cliente com base em seu histórico de compras.

Por fim, a camada de análise de dados seria responsável por apresentar esses insights de forma clara e acessível aos usuários finais, como executivos e gerentes de vendas, para que possam tomar decisões informadas com base nos dados coletados. Isso poderia incluir a criação de relatórios e visualizações de dados interativos para apresentar as informações de maneira clara e concisa.



2.5. Camada de visualização

A camada de visualização é a parte do processo de análise de dados que se concentra em apresentar as informações extraídas da camada de análise de dados de forma clara, concisa e acessível aos usuários finais. Essa camada tem como objetivo fornecer insights acionáveis de maneira intuitiva e visualmente atraente.

A camada de visualização geralmente envolve o uso de gráficos, tabelas, mapas e outras ferramentas visuais para apresentar os dados de maneira clara e eficaz. O objetivo é permitir que os usuários finais possam compreender rapidamente os insights e tomar decisões informadas com base nas informações apresentadas. A camada de visualização também pode envolver a interação com os dados em tempo real, permitindo aos usuários finais filtrar, classificar e explorar os dados de diferentes maneiras para obter insights adicionais.

As ferramentas utilizadas na camada de visualização podem incluir softwares de visualização de dados, dashboards interativos, infográficos e relatórios personalizados. Essas ferramentas são projetadas para apresentar as informações de maneira atraente e fácil de entender, independentemente do nível de experiência em análise de dados do usuário final.

3. Pentaho Data Integration e Pipeline de Dados

O Pentaho Data Integration (também conhecido como Kettle) é uma ferramenta de ETL (Extração, Transformação e Carregamento) que permite extrair, transformar e carregar dados de diferentes fontes em diferentes destinos. O Pentaho Data Integration pode ser usado em um pipeline de dados para automatizar o processo de ETL.

Um pipeline de dados é um conjunto de etapas que são executadas para transformar dados brutos em informações úteis. Essas etapas geralmente incluem a extração de dados de várias fontes, a limpeza e transformação dos dados, a análise dos dados e a carga dos dados em um destino.

O Pentaho Data Integration pode ser usado para criar um pipeline de dados com as seguintes etapas:

- **Extração:** o Pentaho Data Integration pode extrair dados de várias fontes, como bancos de dados relacionais, planilhas, arquivos CSV, arquivos XML, serviços da web, entre outros;
- **Transformação:** o Pentaho Data Integration oferece várias transformações para limpar e transformar dados, incluindo a limpeza de dados inconsistentes, a validação de dados, a combinação de dados de várias fontes e a criação de campos calculados;
- **Análise:** o Pentaho Data Integration também pode ser usado para analisar dados e gerar relatórios, gráficos e dashboards;
- **Carga:** finalmente, o Pentaho Data Integration pode carregar os dados transformados em um destino, como um banco de dados relacional, um arquivo CSV ou um serviço da web.

Em resumo, o Pentaho Data Integration é uma ferramenta poderosa que pode ser usada em um pipeline de dados para automatizar o processo de ETL. Ele oferece várias funcionalidades para extrair, transformar e carregar dados, além de permitir a análise dos dados e a geração de relatórios e gráficos.

3.1. JOBS no Pentaho Data Integration

Os Jobs no Pentaho Data Integration são responsáveis por orquestrar a execução dos pipelines de dados. Um Job é um conjunto de passos, e cada passo pode assumir o papel de pipeline de dados, transformações, execução de scripts ou outras operações.

Em um pipeline de dados, você pode definir a sequência de passos que devem ser executados para realizar uma determinada tarefa. Um exemplo seria a extração de dados de uma fonte, a transformação e validação dos dados e, finalmente, o carregamento em um destino.

Em um Job, você pode executar um ou mais pipelines de dados, em uma ordem específica, com base em determinadas condições, para automatizar processos mais complexos. Por exemplo, você pode criar um Job que executa um pipeline de dados que extrai dados de uma fonte, verifica se há duplicatas, transforma os dados e, em seguida, carrega os dados em um banco de dados. A partir disso, você pode criar um segundo pipeline de dados que executa a validação de dados para garantir que não há erros. O Job pode ser configurado para executar o primeiro pipeline de dados e, se tudo correr bem, poderá também executar o segundo pipeline de dados.

Em resumo, o Pentaho Data Integration oferece tanto a funcionalidade de pipeline de dados como de Jobs para automatizar processos mais complexos. Enquanto os pipelines de dados podem ser usados para executar uma tarefa específica, os Jobs podem ser usados para orquestrar vários pipelines de dados e outras operações para executar processos mais complexos.

3.2. Principais Steps utilizados nos JOBs

Existem vários steps que podem ser utilizados em Jobs no Pentaho Data Integration, cada um com uma finalidade específica. Aqui estão alguns dos principais steps que são frequentemente utilizados em Jobs:

- **Start:** este é o primeiro passo em um Job e é usado para iniciar o processo;
- **Transformation:** este passo é usado para executar um pipeline de dados;
- **Job:** este passo é usado para executar um subJob. Isso é útil quando você tem um Job maior que precisa chamar um Job menor;
- **Simple Evaluation:** este passo é usado para avaliar uma condição simples e, em seguida, executar um dos dois caminhos, dependendo do resultado;
- **File Exists:** este passo é usado para verificar se um arquivo existe em um diretório específico;
- **Table Exists:** este passo é usado para verificar se uma tabela existe em um banco de dados específico;
- **Execute SQL Script:** este passo é usado para executar um script SQL em um banco de dados;
- **Execute a Process:** este passo é usado para executar um processo externo, como um script Python ou um executável;
- **Success:** este passo é usado para indicar que o Job foi executado com sucesso;
- **Failure:** este passo é usado para indicar que o Job falhou.

Esses são apenas alguns dos steps que podem ser utilizados em Jobs no Pentaho Data Integration. Existem muitos outros disponíveis para executar uma ampla variedade de tarefas. É importante escolher o step certo para a tarefa que você está executando para garantir que seu Job seja executado com eficiência.



4. Pentaho Data Integration – JOBs

Neste ponto do conteúdo, serão criados 5 fluxos de dados utilizando JOBs no Pentaho Data Integration, bem como será explicada a teoria dos Steps relacionados a cada fluxo. A parte prática será exposta em sala de aula.

4.1. Fluxo 1

Neste ponto do conteúdo, serão criados 5 fluxos de dados utilizando JOBs no Pentaho Data Integration, bem como será explicada a teoria dos Steps relacionados a cada fluxo. A parte prática será exposta em sala de aula.

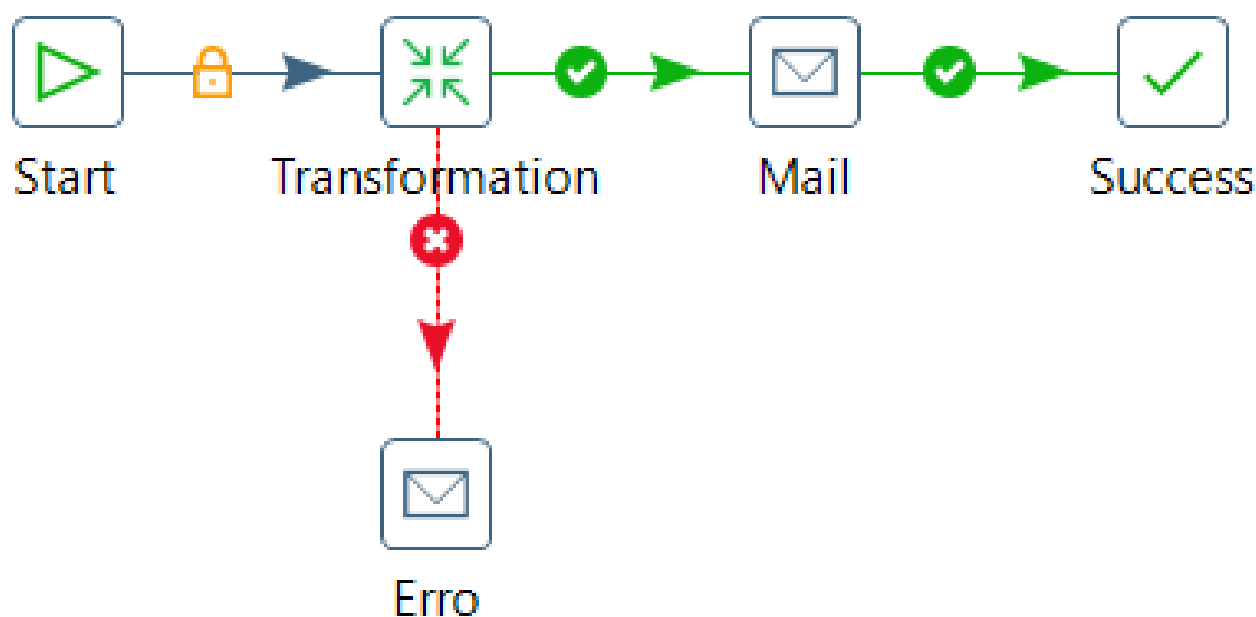


Figura 01

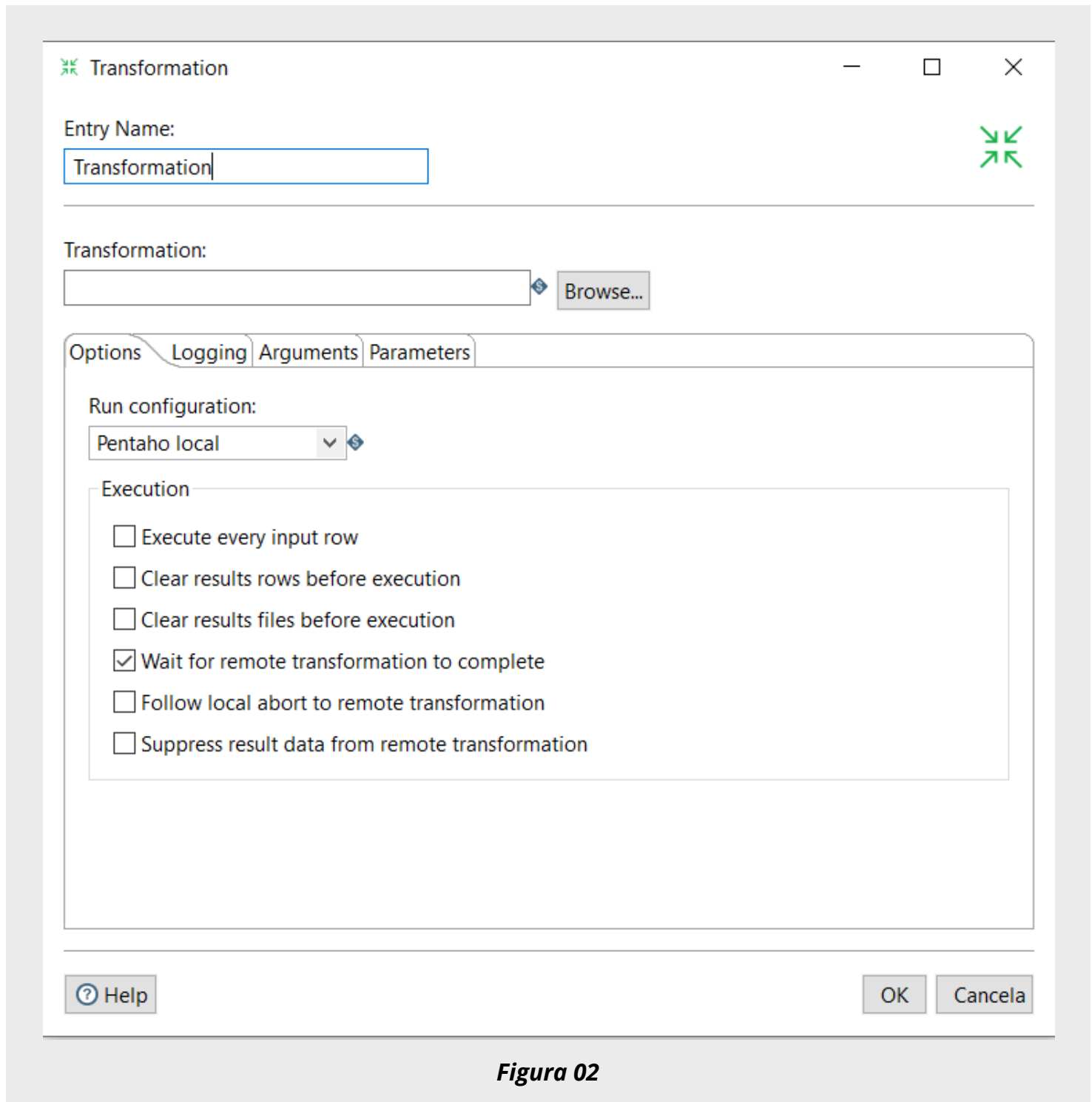
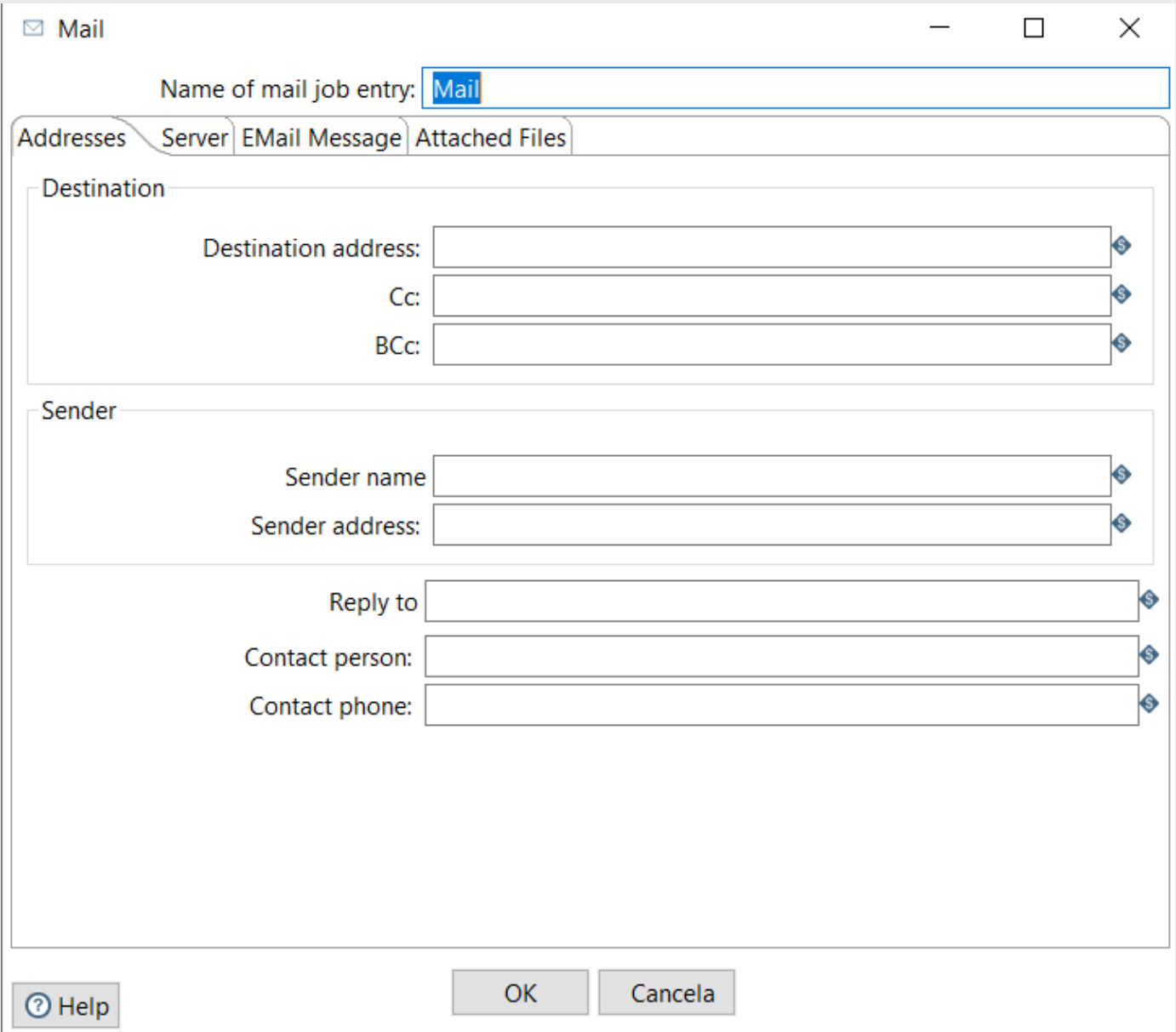


Figura 02

No passo da transformação, basta escolher o caminho e o arquivo KTR.



Mail

Name of mail job entry: Mail

Addresses Server E-Mail Message Attached Files

Destination

Destination address:

Cc:

Bcc:

Sender

Sender name:

Sender address:

Reply to:

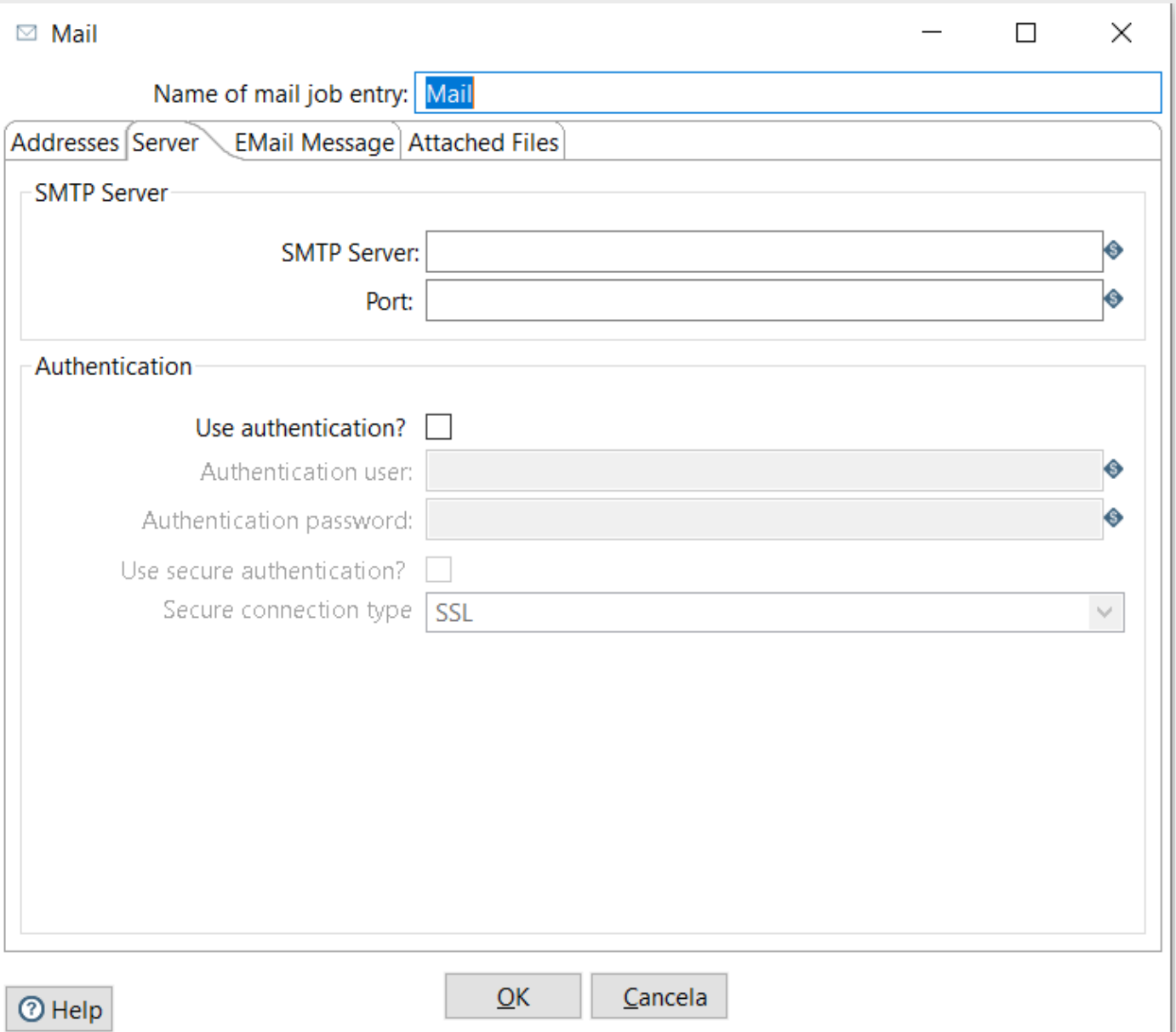
Contact person:

Contact phone:

Help OK Cancela

Figura 03

Nessa primeira guia, colocamos o endereço de e-mail do destinatário.



Mail

Name of mail job entry: Mail

Addresses Server EMail Message Attached Files

SMTP Server

SMTP Server:

Port:

Authentication

Use authentication? ☐

Authentication user:

Authentication password:

Use secure authentication? ☐

Secure connection type SSL

Help OK Cancela

Figura 04

Na guia server é feita a configuração do servidor de envio de e-mail. Em SMTP Server, você deve colocar o IP ou DNS do servidor que vai enviar seus e-mails.

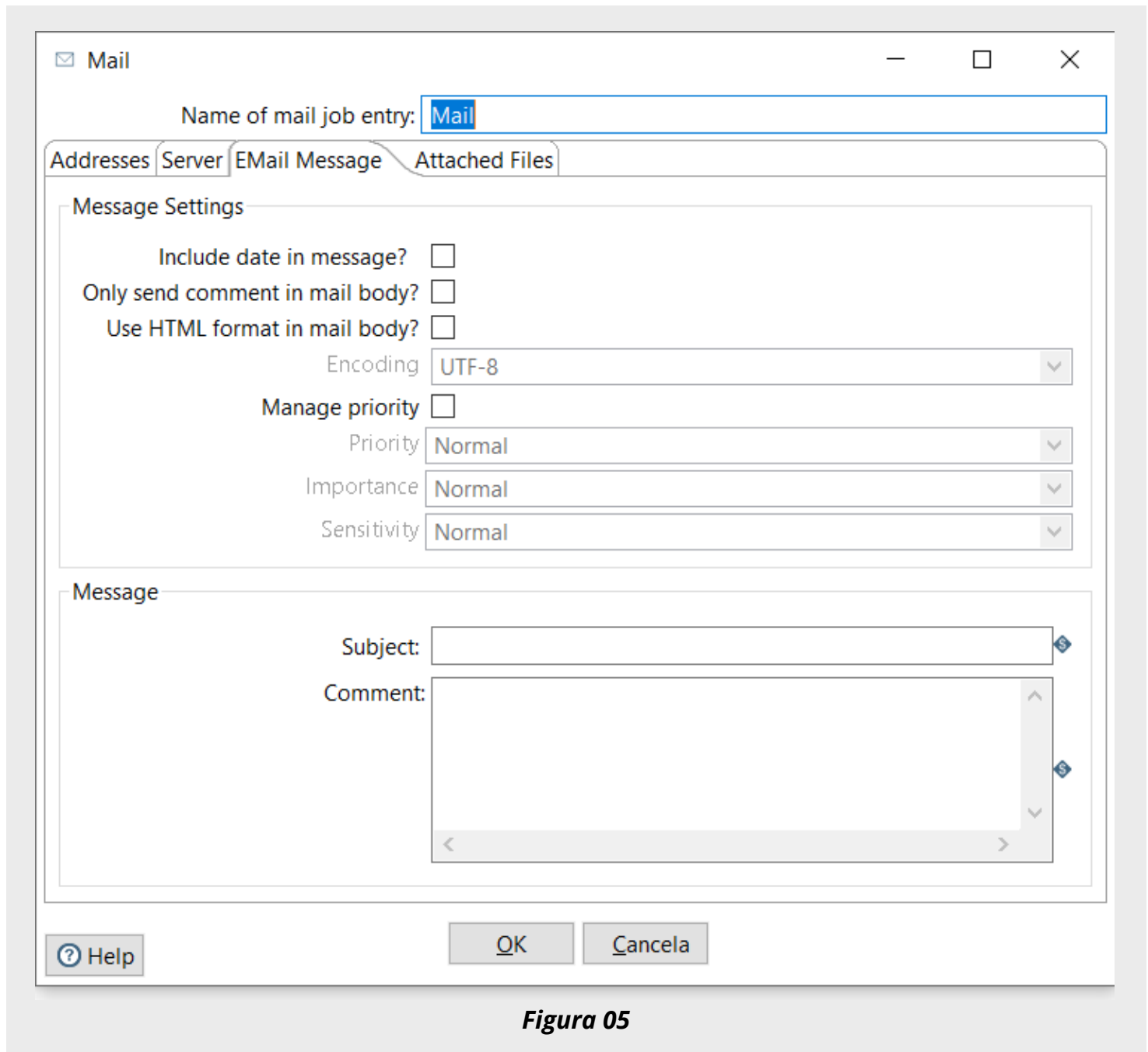
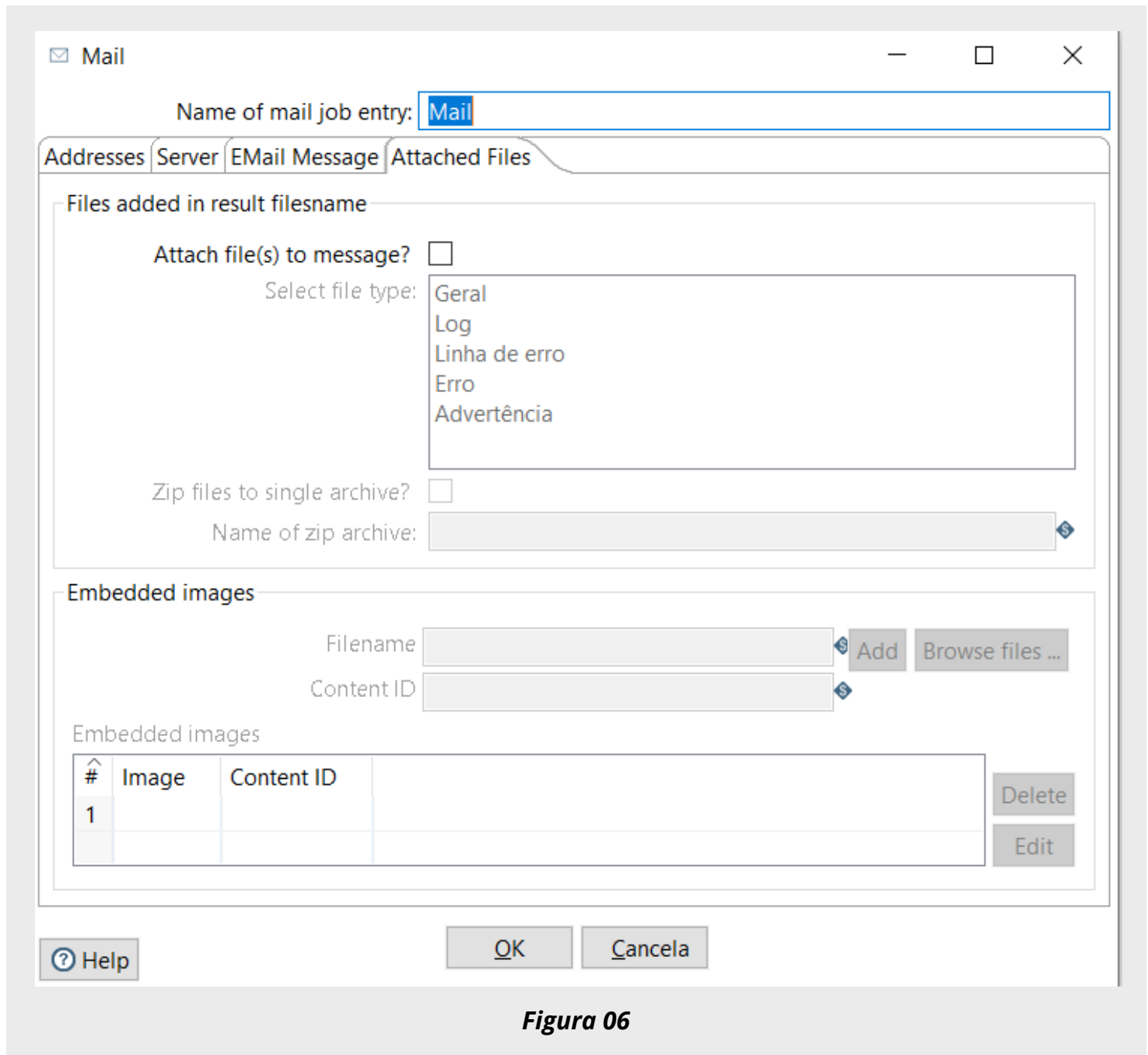


Figura 05

Na guia Email Message, vamos preencher os campos Subject – é o título do e-mail – e o campo Comment – é a mensagem propriamente dita.



Mail

Name of mail job entry: **Mail**

Addresses | **Server** | **EMail Message** | Attached Files

Files added in result filename

Attach file(s) to message? ☐

Select file type: Geral
Log
Linha de erro
Erro
Advertência

Zip files to single archive? ☐

Name of zip archive:

Embedded images

Filename Add Browse files ...

Content ID

Embedded images

#	Image	Content ID
1		

Delete Edit

Help OK Cancela

Figura 06



Por fim, na última guia fica a opção de colocar anexo no e-mail de acordo com as opções do select file type.

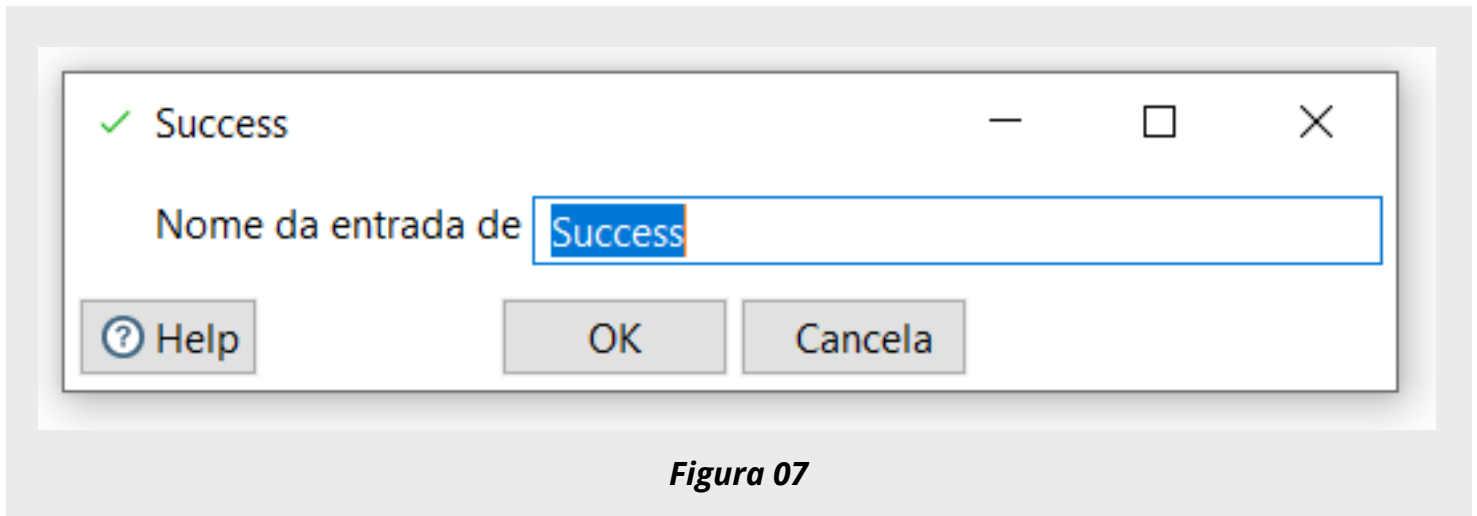
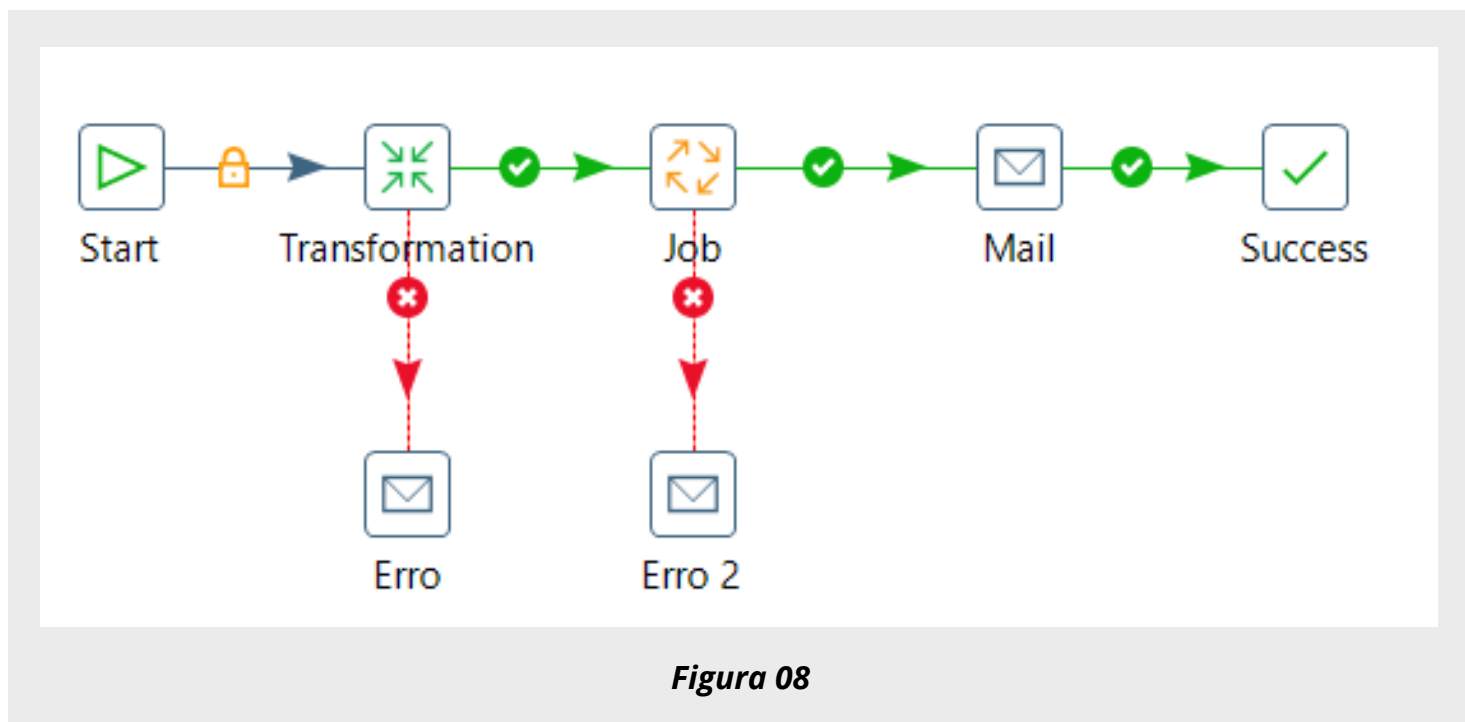


Figura 07

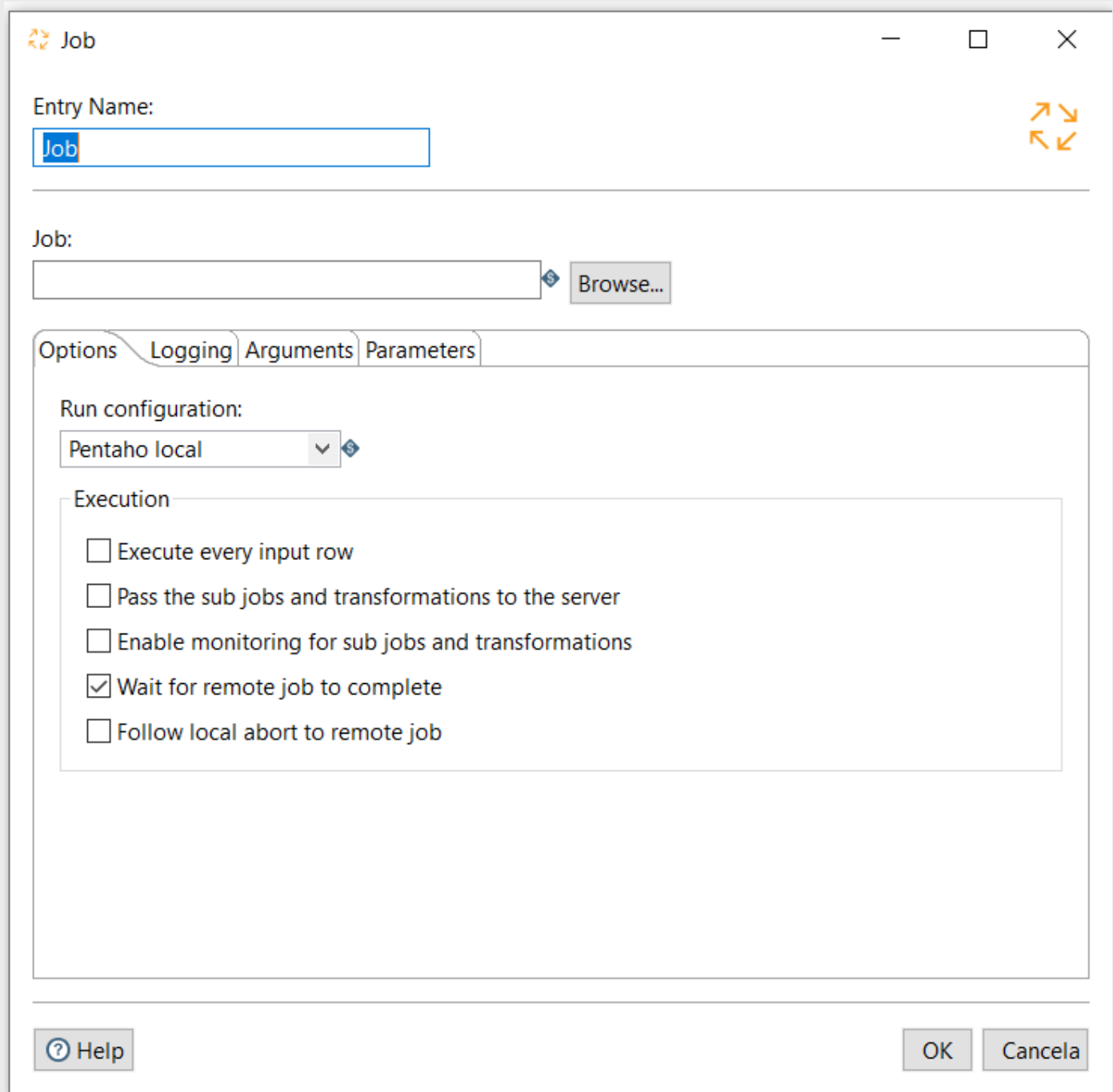
Todo JOB obrigatoriamente deve começar com o step Start, pois este dá início à execução do fluxo de dados. No segundo passo, temos uma Transformation – em suas propriedades temos que configurar o caminho da transformação (KTR) que queremos executar. Logo na sequência temos dois possíveis caminhos, um quando ocorrer erro e outro quando não ocorrer erro. Em ambos os casos serão enviados e-mails informando a situação do nosso fluxo de dados. Por fim, se tudo ocorrer bem, temos o passo final, que é o Success.

4.2. Fluxo 2

Neste ponto do conteúdo, serão criados 5 fluxos de dados utilizando JOBs no Pentaho Data Integration, bem como será explicada a teoria dos Steps relacionados a cada fluxo. A parte prática será exposta em sala de aula.



No JOB da figura, temos o incremento de um step que vai servir para acessar outro JOB. É isso mesmo: um JOB pode chamar outro JOB.



Job

Entry Name:

Job

Job:

Browse...

Options | Logging | Arguments | Parameters

Run configuration:

Pentaho local

Execution

- ☐ Execute every input row
- ☐ Pass the sub jobs and transformations to the server
- ☐ Enable monitoring for sub jobs and transformations
- ☒ Wait for remote job to complete
- ☐ Follow local abort to remote job

Help OK Cancela

Figura 09

Similarmente ao step do fluxo anterior, onde chamarmos uma transformação nesse step, chamaremos outro JOB.

4.3. Fluxo 3

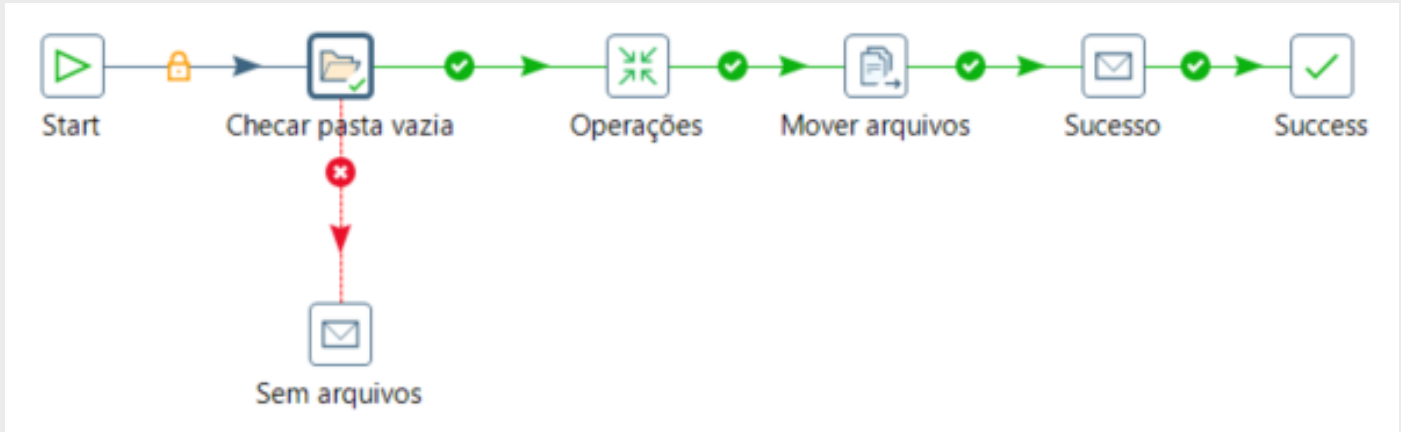


Figura 10

O terceiro fluxo realiza a verificação se uma determinada pasta está vazia. Caso esteja com conteúdo, o fluxo passará para o passo das operações; caso não, o fluxo enviará um e-mail alertando que a pasta está vazia.

No passo das operações, o fluxo irá chamar um KTR como exposto nos fluxos anteriores. Na sequência, ele moverá o conteúdo dessa pasta para outra pasta e, por fim, finalizará o processo enviando um e-mail de sucesso.

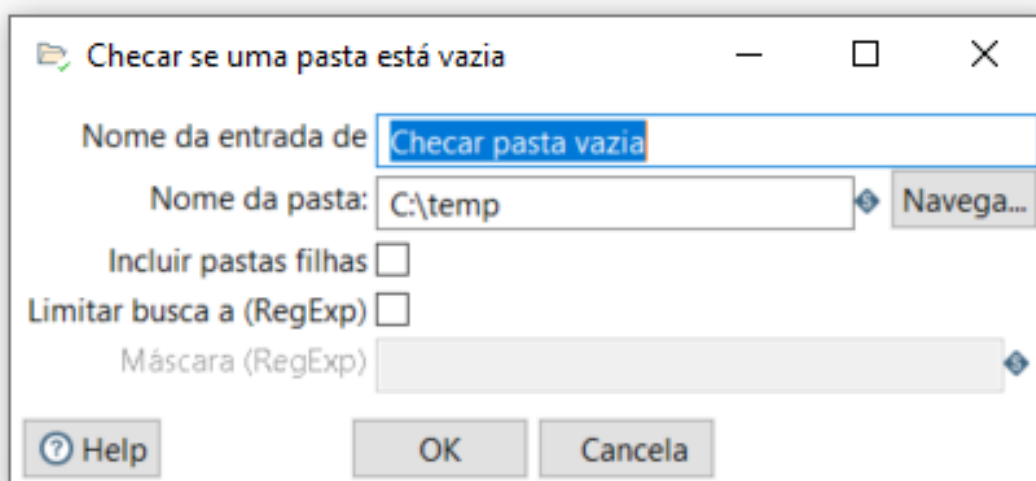


Figura 11



Nesse step vamos clicar no botão “navegar” e escolher a pasta que queremos verificar. Também temos a opção de incluir as pastas filhas.

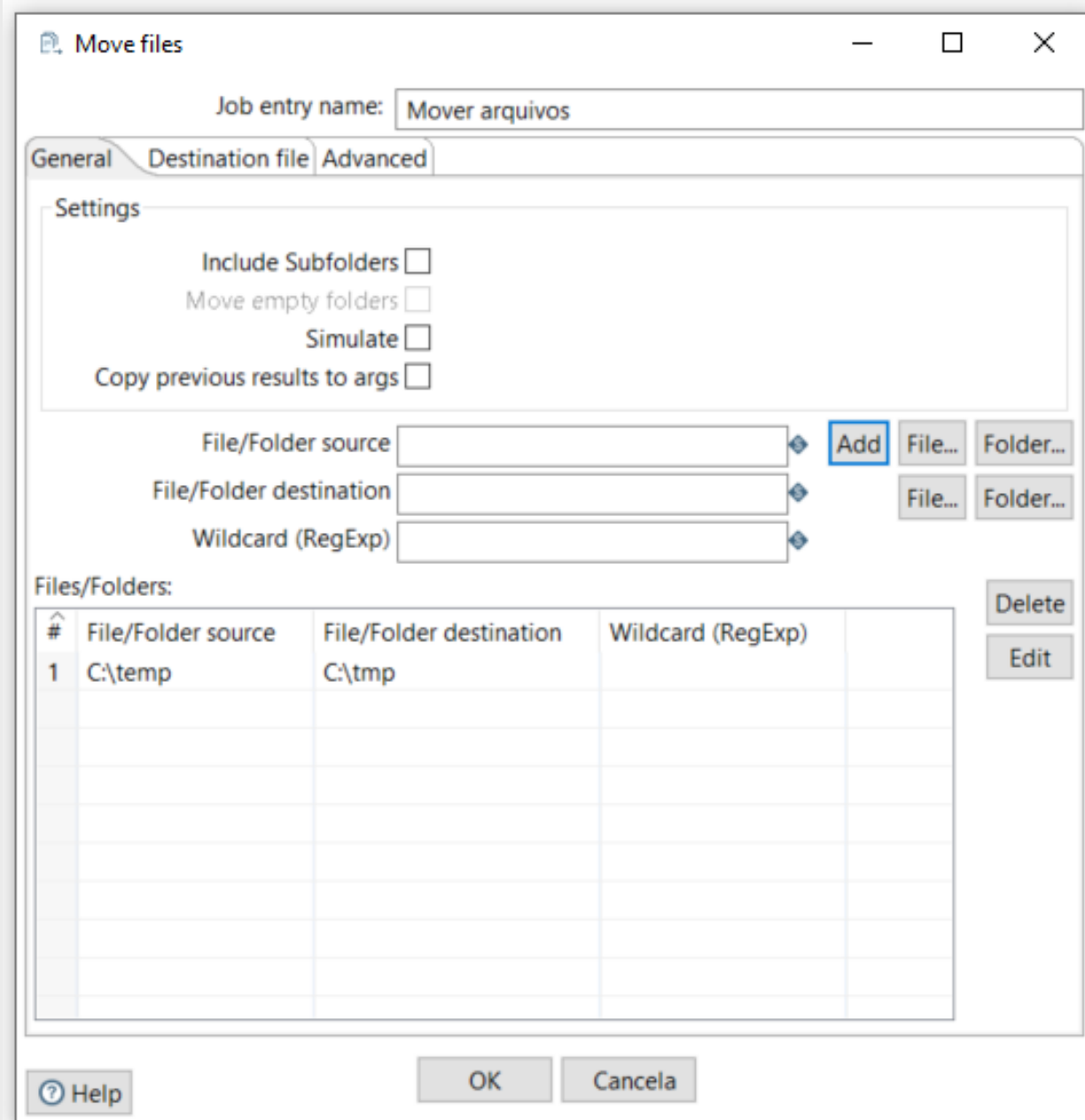


Figura 12

Na figura 12, configuramos o step que move os arquivos. Nesse passo vamos escolher duas pastas, uma de origem, de onde iremos recortar os arquivos, e outra de destino, onde iremos colar os arquivos. Após essas escolhas, basta clicar no botão add.



5. Conceitos do Pentaho Server

Neste ponto do conteúdo, serão criados 5 fluxos de dados utilizando JOBS no Pentaho Data Integration, bem como será explicada a teoria dos Steps relacionados a cada fluxo. A parte prática será exposta em sala de aula.

Pentaho Business Analytics

Figura 13

O Pentaho Server é uma plataforma de BI (Business Intelligence) que oferece recursos para gerenciamento de dados, geração de relatórios, análise e visualização de dados. Alguns dos principais conceitos do Pentaho Server incluem:

1. **Repositório:** o repositório é onde todos os dados, relatórios e dashboards são armazenados. Ele é usado para gerenciar o acesso aos dados e garantir a integridade dos dados;
2. **Data Source:** um conjunto de dados que é usado para criar relatórios e análises. O Pentaho Server suporta várias fontes de dados, incluindo bancos de dados relacionais, arquivos CSV e Excel, dentre outros;
3. **ETL (Extract, Transform, Load):** o Pentaho Server inclui o Pentaho Data Integration, uma ferramenta de ETL que permite extrair, transformar e carregar dados em diferentes fontes de dados;
4. **Relatórios:** o Pentaho Server permite criar relatórios a partir de várias fontes de dados. Os relatórios podem ser personalizados para incluir tabelas, gráficos e outros elementos visuais;

5. **Dashboards:** o Pentaho Server inclui recursos para criar dashboards personalizados que permitem visualizar informações importantes de várias fontes de dados em um único lugar;
6. **Segurança:** o Pentaho Server oferece recursos de segurança para proteger o acesso aos dados. Os usuários podem ser autenticados e autorizados com base em suas permissões;
7. **Agendamento:** o Pentaho Server permite agendar a execução de relatórios e outros processos em um horário específico. Isso permite automatizar tarefas e garantir que os relatórios sejam gerados regularmente.

Esses são apenas alguns dos principais conceitos do Pentaho Server. Ele é uma plataforma rica em recursos e oferece muitas outras funcionalidades para gerenciamento de dados, geração de relatórios e análise de dados.

5.1. Arquitetura do Pentaho Server

A arquitetura do Pentaho Server é composta por vários componentes que trabalham juntos para fornecer recursos de BI (Business Intelligence) e gerenciamento de dados. Aqui estão os principais componentes da arquitetura do Pentaho Server:

- **Pentaho User Console (PUC):** o PUC é a interface web do Pentaho Server. É aqui que os usuários podem acessar e interagir com relatórios, dashboards e outros recursos;
- **Pentaho Metadata Editor (PME):** o PME é usado para criar e gerenciar metadados, que descrevem os dados em diferentes fontes de dados. Os metadados permitem que os usuários acessem e manipulem dados de várias fontes de dados sem ter que entender as complexidades subjacentes dos dados;

- **Pentaho Data Integration (PDI):** o PDI é uma ferramenta de ETL (Extract, Transform, Load) que permite extrair, transformar e carregar dados em diferentes fontes de dados. Ele é usado para criar pipelines de dados que movem dados de uma fonte para outra;
- **Pentaho Reporting:** o Pentaho Reporting é usado para criar e gerar relatórios a partir de várias fontes de dados. Ele oferece recursos para criar relatórios personalizados com tabelas, gráficos e outros elementos visuais;
- **Pentaho Analysis Services (Mondrian):** o Mondrian é uma ferramenta de OLAP (Online Analytical Processing) que permite aos usuários analisar dados multidimensionais. Ele oferece recursos para criar cubos de dados e visualizá-los em diferentes perspectivas;
- **Pentaho Dashboards:** o Pentaho Dashboards permite criar e compartilhar dashboards personalizados que permitem visualizar informações importantes de várias fontes de dados em um único lugar;
- **Pentaho Security:** o Pentaho Security permite controlar o acesso aos recursos do Pentaho Server. Ele oferece recursos para autenticação de usuários, gerenciamento de permissões e outras funcionalidades de segurança.

Esses são apenas alguns dos principais componentes da arquitetura do Pentaho Server. A plataforma é altamente modular e escalável, permitindo que os usuários adicionem e personalizem recursos de acordo com suas necessidades específicas.

5.2. Instalação e Configuração do Pentaho Server

A instalação e a configuração do Pentaho Server envolvem as seguintes etapas:

- **Requisitos de sistema:** antes de começar a instalação, certifique-se de que seu sistema atende aos requisitos mínimos de hardware e software. Verifique se a versão do Java instalada é compatível com o Pentaho Server;
- **Baixe o arquivo de instalação:** faça o download do arquivo de instalação do Pentaho Server a partir do site oficial da Pentaho. Escolha a versão adequada para o seu sistema operacional;
- **Descompacte o arquivo:** depois de baixar o arquivo, descompacte-o em um diretório no seu sistema;
- **Inicie o Pentaho Server:** para iniciar o Pentaho Server, navegue até o diretório onde você descompactou o arquivo de instalação e execute o arquivo `start-pentaho.bat` ou `start-pentaho.sh`, dependendo do sistema operacional;
- **Acesse o Pentaho User Console:** depois que o Pentaho Server for iniciado, você poderá acessar o Pentaho User Console (PUC) em um navegador da web. Digite o endereço `http://localhost:8080/pentaho` no seu navegador para acessar o PUC;
- **Configurações do servidor:** a partir do PUC, você pode acessar as configurações do servidor e personalizar as opções de segurança, idioma e outros recursos. É recomendável revisar e configurar essas opções de acordo com suas necessidades;



- **Configurações do usuário:** crie usuários e configure as permissões para acessar os recursos do Pentaho Server. Você pode criar usuários locais ou integrar o Pentaho Server com seu diretório de autenticação existente, como Active Directory ou LDAP;
- **Configurações de fonte de dados:** configure as fontes de dados que o Pentaho Server usará para acessar os dados. Você pode configurar fontes de dados para bancos de dados relacionais, arquivos CSV, Excel e outras fontes de dados;
- **Configurações de segurança:** configure as opções de segurança para proteger o acesso aos recursos do Pentaho Server. Você pode definir permissões para usuários individuais ou grupos de usuários e limitar o acesso a recursos específicos.

Essas são as etapas básicas para instalar e configurar o Pentaho Server. Lembre-se de seguir as instruções específicas para a versão que você está instalando e revise a documentação oficial para obter mais informações e suporte.



6. Apache Airflow: maestro de pipelines de tarefas agendadas

Apesar de ser incrivelmente eficaz, o Airflow pode se tornar relativamente confuso para iniciantes (e algumas vezes até mesmo para usuários avançados) devido ao seu grande número de configurações e ampla versatilidade, que, conseqüentemente, acabam trazendo uma carga de complexidade a mais para o projeto.

6.1. O que o Airflow é?

O Apache Airflow é uma ferramenta de código aberto, escrita em python e desenvolvida pela Apache Foundation, seu objetivo é orquestrar pipelines de tarefas agendadas por meio de arquivos python com instruções de sequenciamento definidas, chamados DAGs. Pense nele como um versátil maestro, capaz de orquestrar diferentes músicas, de diversos tempos e com diferentes instrumentos de maneira igualmente ótima.

Para seu funcionamento, o Airflow conta com alguns elementos chave que permitem a existência da sinergia necessária entre tarefas, eventos, estados e filas, todos funcionando de maneira sincronizada e de acordo com as configurações definidas pelo usuário.

A Figura abaixo representa, de maneira relativamente simplificada e em uma mesma máquina (visto que é possível configurar o Airflow de maneira escalonável) a estrutura de uma instância da ferramenta:

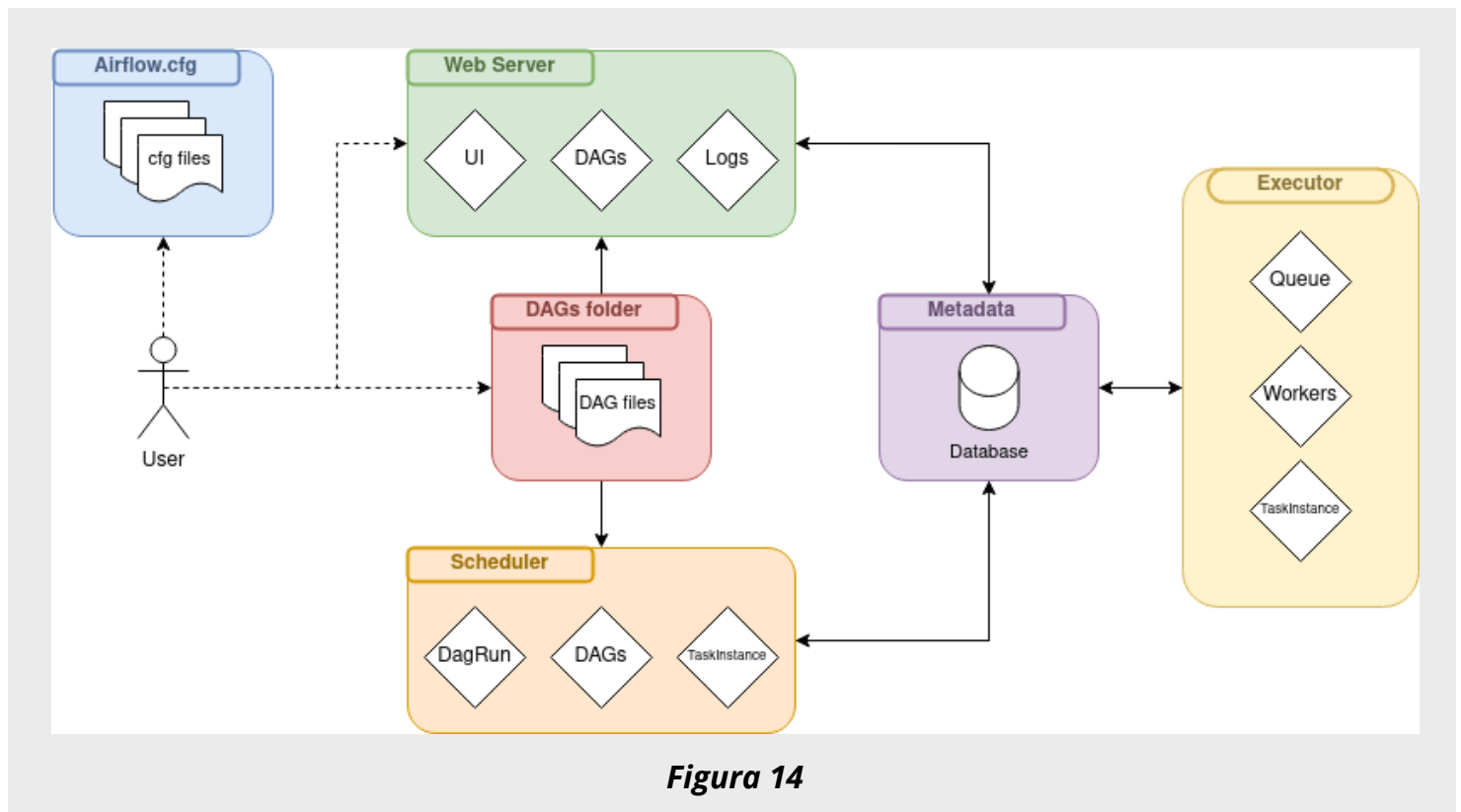


Figura 14

Sendo assim, descrevendo de maneira também simplificada a funcionalidade e comportamento de cada elemento apresentado na Figura anterior:

- **Airflow.cfg:** arquivo de configurações que descreve, principalmente, as conexões utilizadas para comunicação com o banco de dados de metadados da ferramenta, os intervalos de verificação de novos arquivos DAGs e a frequência de atualização dos estados correntes de cada tarefa;
- **Web Server:** subsistema responsável pela integração e execução de uma interface visual para o usuário. Aqui é apresentada graficamente a maior parte dos elementos que podem ser utilizados pelo usuário, como DAGs, logs, alertas, avisos e todo tipo de monitoramento do sistema;

- **Scheduler:** este componente pode ser entendido como o “coração” do Airflow. No mundo musical, é possível comparar o Scheduler a um metrônomo, ou ao compasso que dá o tempo à música. Aqui, ele é responsável pela temporização do sistema, resultando na execução programada de DAGs, no agendamento de execução de tarefas individuais das DAGs e também da distribuição destas para diferentes Executors. Resumidamente, garantir o bom funcionamento deste componente faz parte de um grande diferencial para garantir o bom funcionamento do Airflow como ferramenta, já que faz a integração de quase todos os outros componentes/subsistemas;
- **Metadata:** se o Scheduler pode ser considerado o “coração” do Airflow, então o banco de dados de metadados seria o “cérebro”. É neste elemento que são armazenadas todas as variáveis utilizadas por todos os outros componentes da ferramenta, desde usuários até retornos de tarefas. Faz uso de um banco de dados relacional que permite, inclusive, a troca de informação entre tarefas, principal causa de problemas de má utilização do Airflow, que serão discutidos mais adiante no conteúdo;
- **Executors:** subsistema responsável pela execução das tarefas programadas pelo Scheduler, que estão localizadas na fila (queue) deste subsistema (na atual representação). O Airflow permite a execução de diferentes tipos de tarefas por meio de operadores de diferentes naturezas, como o PythonOperator, para execução de scripts Python, o DockerOperator, para trabalho com containers do Docker ou até mesmo o BashOperator, para a execução de comandos bash. Essa versatilidade permite que cada tarefa possa ser executada isoladamente dentro do ambiente específico definido como Worker. Todas essas funções são definidas (para a arquitetura local apresentada) e executadas dentro de um Executor, que ao fim se comunica com o banco de metadados para informar o retorno das ações.

Com a integração de todos esses componentes, o usuário é capaz, então, de escrever e programar a execução de diferentes conjuntos de tarefas acíclicas com uma imensa variedade de possibilidades para a execução de cada tarefa, que vão desde interpretadores Python, containers Docker e até mesmo comandos bash.

6.2. O que o Airflow não é

Após as definições dos componentes que integram o Airflow, é possível que o leitor esteja imaginando diferentes tipos de fluxo de trabalho, com inúmeras aplicações e a possibilidade de processar dados quase que de forma irrestrita. Este conteúdo se faz ainda mais importante para este caso, visto que uma limitação extremamente importante foi apresentada indiretamente no conteúdo anterior mas que pode ter passado despercebida: o Airflow não é um processador de dados e não pode ser utilizado para sequências indefinidamente cíclicas de tarefas. Trazendo ao leitor novamente a analogia musical, seu maestro até pode ser um grande multi-instrumentista, mas acaba sendo impossível orquestrar a música e efetivamente tocar seus instrumentos simultaneamente.

O problema de processamento de dados nesta ferramenta pode ser mais facilmente compreendido através da explicação mais minuciosa do funcionamento da transferência de dados entre tarefas. A transferência de dados entre tarefas é feita através de um componente chamado Xcom, que nada mais é do que a abstração de acesso, leitura e escrita de dados no banco de dados do Airflow. Ou seja, para cada leitura/escrita desse banco de dados, é necessário fazer uma conexão e executar uma nova operação no banco, que, para grandes quantidades de dados, pode acabar resultando em problemas de consulta para outras DAGs ou tarefas que estejam sendo executadas simultaneamente. Por esse motivo, é indicado que o processamento de dados seja feito externamente ao Airflow, utilizando Xcoms apenas para a troca de pequenas informações entre tarefas, como metadados.

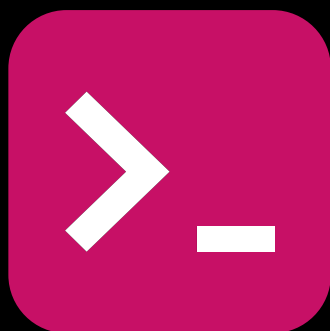


Já o problema de execução de tarefas indefinidamente cíclicas se dá pela maneira que os diferentes componentes e subsistemas dentro do Airflow se comunicam. É esperado que um bloco de tarefas (DAG) tenha um início bem definido e programado temporalmente, que em seguida irá executar suas tarefas de forma sequencial ou paralela até que chegue a um fim determinado, atualizando, assim, o estado da execução desse conjunto de tarefas com a condição final dele (sucesso ou falha). Por esse motivo, e para evitar que usuários tentem executar tarefas indefinidamente, o Airflow utiliza e deixa tão claro o conceito de grafos acíclicos dirigidos.



Referências

- ALVES, William Pereira. **Banco de Dados [BV:MB]**. 1ª ed. São Paulo: Érica, 2014.
- BALIEIRO, R. **Banco de Dados [BV:RE]**. 1ª ed. Rio de Janeiro: SESES, 2015.
- ELMASRI, R.; NAVATHE. **S. Sistemas de Banco de Dados [BV:PE]**. 7 ed. São Paulo: Pearson, 2018.
- FONSECA, Cleber Costa da. **Implementação de banco de dados. Banco de Dados [BV:RE]**. 1ª ed. Rio de Janeiro: SESES, 2016.
- HEUSER, C. **Projeto de Banco de Dados. [BV:MB]**. 6 ed. Porto Alegre: ARTMED, 2009.
- MACHADO, Felipe N. R. **Banco de Dados - Projeto e Implementação [BV:MB]**. 3 ed. São Paulo: Érica, 2014.
- NETO, Geraldo H. **MODELAGEM DE DADOS. [BV:RE]**. 1ª ed. Rio de Janeiro: SESES, 2015.
- PUGA, Sandra; FRANÇA, Edson; GOYA, Milton. **Banco de Dados: implementação em SQL, PL/SQL e Oracle 11g [BV:PE]**. 1ª ed. São Paulo: Pearson, 2013.
- RAMAKRISHNAN, R. **Sistemas de gerenciamento de banco de dados [BE:MB]**. 3 ed. Porto Alegre: McGraw-Hill, 2008.



Digital College

ENSINO DE HABILIDADES DIGITAIS



@digitalcollegebr



/school/digitalcollegebr



/digitalcollegebr



digitalcollege.com.br

