CAPÍTULO 3

O Framework ETL4NoSQL

Neste capítulo serão apresentados os conceitos do Framework ETL4NoSQL. Este consiste numa plataforma de software para desenvolvimento de sistemas ETL, cujo os dados de origem provêm de bases de dados não relacionais. Mais especificamente, bases de dados NoSQL que pertencem a um dos quatro paradigmas de NoSQL: Orientada a documentos, Família de Colunas, Chave-Valor e Baseada em Grafos.

O Framework oferece um ambiente integrado para modelar processos de ETL e implementar funcionalidades utilizando uma linguagem de programação independente de uma GUI (*Graphical User Interface* - Interface Gráfica do Usuário).

Para a especificação do Framework foram definidas as estruturas dos dados dos ambientes de origem, destino e área de processamento de dados e suas respectivas linguagens de manipulação, e também, as principais funcionalidades dos sistemas de ETL, chamados mecanismos de ETL. Para realizar os processos de ETL foi definido um controlador de operações que é capaz de se comunicar com os ambientes e os mecanismos de ETL.

Nas seções deste capítulo serão detalhados os requisitos, a arquitetura, os fluxos de dados e diagramas utilizados no desenvolvimento do Framework.

3.1 Requisitos do ETL4NoSQL

Para que seja possível a extração, transformação e carga dos dados armazenados em bancos de dados que utilizam um dos paradigmas de NoSQL é preciso que seja definido o esquema no qual os dados necessários estão armazenados. Dessa forma, algumas questões importantes são abordadas:

- Quanto os BDs NoSQL diferem dos BDs suportados pelas ferramentas de ETL?
- Como é possível oferecer suporte para ETL em BDs NoSQL?

As diferenças dos BDs NoSQL já foram abordadas nos capítulos de fundamentação teórica. Assim, fica explícito o problema de como é possível converter os modelos de dados NoSQL em modelos relacionais que possam ser lidos por qualquer ferramenta de ETL. [Nas12] define uma tabela como uma representação de uma coleção de instâncias de entidades comparáveis. Então, possibilitar suporte para armazenamentos de dados NoSQL é permitir extração e importação de instâncias de entidades comparáveis em tabelas relacionais. Porém, os problemas a serem resolvidos abordados por [Nas12] para atingir isso incluem:

1. Como permitir o usuário especificar as entidades comparáveis e seus atributos?
2. Como, e de onde, extrair exatamente?

A solução sugerida pelo autor é um esquema ser deduzido por meio de uma amostra do banco de dados. Este esquema pode, então, ser apresentado ao usuário que procede selecionando quais atributos importar do que foi apresentado - exatamente como no caso do RDBMS. É claro que também deverá ser possível ao usuário editar os esquemas apresentados tendo em vista que a amostra pode não ser perfeita. Essa abordagem geral é referida como dedução de esquema - *schema inference*.

Construir uma amostra com todo o banco de dados pode ser muito custoso, por isso é preferível que faça uma amostra com pequenas estruturas, como por exemplo, uma entidade inteira. É claro que muitas vezes não há esse tipo de estrutura, então é possível que a amostra seja colocada em clusters. Múltiplas entidades podem ser divididas cada uma em um cluster, e assim, é possível ter um esquema de entidade por cluster.

Dessa forma, dado o que extrair, consultar e recuperar dados de um RDBMS é direto por causa do SQL. Sistemas NoSQL geralmente tem diferentes interfaces que suportam diferentes tipos de consultas. Então, não há nenhuma sugestão de solução geral para isso. Ao invés disso, uma investigação de cada interface particular dos sistemas deve ser conduzida para resolver o problema [Nas12].

Para suprir o problema a respeito das várias interfaces a serem lidadas na extração de dados de bases NoSQL, foi sugerido criar um ambiente programável que oferecesse interfaces previamente selecionadas e permitisse a inserção de novas interfaces por meio de linguagem de programação.

3.1.1 Descrição Geral do Sistema

O ETL4NoSQL consiste em um framework programável, flexível e integrado para modelagem e execução de processos ETL em BDs NoSQL.

Separamos os requisitos em duas etapas, os requisitos para as funcionalidades de ETL e os requisitos para a usabilidade de BDs NoSQL. Baseado nesses requisitos foi possível identificar os padrões de análise do framework, e assim, definir sua arquitetura.

3.2 Padrões de Análise de NoSQL

Segundo a pesquisa de Nasholm (2012), os requisitos necessários para lidar com dados que são armazenados em banco de dados não relacionais devem suportar buscas sob demanda. Devido à característica de bancos de dados NoSQL lidarem com um grande volume de dados é importante que as buscas sejam feitas conforme a necessidade desses dados. Dessa forma, os requisitos para BDs NoSQL são baseados nessa premissa, os padrões definidos pelos requisitos levantados foram os padrões de importação e padrões de mapeamento.

3.2.1 Padrões de importação

Padrões de importação são requisitos para possibilitar a importação dos dados de bases NoSQL, e assim, permitir a execução de processos ETL.

3.2.1.1 Definição de ambiente e estrutura de dados

a) Contexto: É preciso definir quais as estruturas da base de dados NoSQL a ser importada, bem como a linguagem de manipulação dos dados da base para que seja possível a importação.

b) Problema: É necessário oferecer a linguagem de manipulação da base de dados a ser importada, os tipos de dados que serão utilizados e local onde os dados se encontram.

c) Estrutura: Define as configurações de importação, oferecendo a informação de onde o dado é localizado, que dados incluir no resultado, como importá-lo e outras fontes de parâmetros específicos.

d) Participantes: Apresenta os elementos para a importação.

- Ambiente: fonte de onde o dado é localizado.
- Consulta: define quais dados serão necessários.
- Repositório: Define os atributos do repositório de dados, bem como a forma de conexão com o ambiente, forma de manipulação dos dados e outras configurações que permitam a leitura e escrita.
- Amostra: Define quais colunas e tipos de dados que serão utilizados para cada ambiente.

e) Próximos padrões: Concluindo as definições de ambiente e estrutura de dados, a próxima etapa é estipular os mecanismos de importação.

3.2.1.2 Mecanismos de importação

a) Contexto: É necessário definir mecanismos para a importação dos dados da base NoSQL de origem para permitir o uso desses dados em ferramentas de ETL.

b) Problema: É preciso determinar quais mecanismos são fundamentais para a importação dos dados.

c) Estrutura: O mecanismo pode ser definido como busca Sob Demanda ou busca Exaustivo. Mecanismo de busca Sob Demanda significa que o dado da fonte de dados só é buscado quando explicitamente demandado, e que não é buscado nenhum outro dado que não foi demandado. O mecanismo de busca Sob Demanda é muito importante quando se trata de análises em Big Data, pois as análises podem ser conduzidas de cima para baixo onde o usuário começa com uma visão agregada de um conjunto de dados e pode explorar detalhes se desejado. A visão inicial do agregado pode originar vários processos de ETL (Nasholm, 2012). Mecanismo de busca Exaustiva a busca é feita em uma única vez. O único requisito necessário é a configuração de importação.

d) Participantes:

Tabela 3.1 Mecanismos básicos para Importação

Fase de Importação	Mecanismo Padrão
Definição da estrutura de dados	Conexão ao Banco de dados Definição da linguagem de manipulação de dados
Busca	Sob Demanda Exaustiva

e) Próximos padrões: Após a definição dos mecanismos de importação, o próximo passo é executar o mapeamento para a estrutura que será utilizada pelos processos de ETL.

3.2.2 Padrões de Mapeamento

Padrões de mapeamento são requisitos para executar o mapeamento dos dados importados das bases NoSQL, modelando-os de forma que possibilite a leitura e execução dos processos de ETL.

3.2.2.1 Execução do processo de mapeamento

a) Contexto: Para cada paradigma NoSQL, é necessário definir regras de mapeamento que transforme o modelo não relacional em uma estrutura que mapeie as relações permitindo a leitura e execução de processos de ETL.

b) Problema: Deve-se identificar a estrutura da base de dados importada e possibilitar o mapeamento dessa base para um esquema que permita a leitura e execução de processos de ETL.

c) Estrutura: O mapeamento é definido de acordo com os quatro paradigmas NoSQL: chave-valor, família de colunas, orientado a documentos e baseado em grafos.

d) Participantes:

Tabela 3.2 Regras de Mapeamento

Paradigma NoSQL	Regras
Chave-Valor	Um campo chave pode ser mapeada como tabela relacional Uma coluna da chave correspondente pode ser mapeada como coluna da tabela da chave O valor correspondente de uma coluna é mapeada como linha da tabela
Família de Colunas	Uma família pode ser mapeada como tabela Uma coluna da família de colunas pode ser mapeada para uma coluna da tabela
Orientado a Documentos	Um documento corresponde a uma linha da tabela Cada campo do documento pode ser coluna da tabela relacional
Baseado em Grafos	O vértice ou aresta pode ser mapeado para uma tabela relacional Cada atributo do vértice ou aresta pode ser mapeado para colunas

e) Próximos padrões: Com as regras de mapeamento definidas, o passo seguinte é definir metadados e os mecanismos de ETL padrões para os processos de ETL.

3.2.3 Padrões de Análise de ETL

Segundo Silva (2012), não há um consenso a respeito das funcionalidades básicas dos sistemas de ETL, porém é possível definir, baseado nos conceitos disponíveis na literatura, requisitos de operações e metadados.

Requisitos de metadados definem as estruturas que cada processo de ETL necessita, onde podem ser consideramos pontos de flexibilidade. Já os requisitos de operações não mudam para qualquer processo de ETL, sendo assim, pontos de estabilidade.

Os principais requisitos de metadados e operações baseadas nas definições de Silva (2012) e consideradas fundamentais para o desenvolvimento do framework deste trabalho são descritas a seguir.

3.2.4 Padrões de Metadados

Padrões de metadados são requisitos de preparação do ambiente e definição de variáveis para a execução de operações de ETL.

3.2.4.1 Definição dos ambientes e das estruturas de dados

Este padrão consiste na preparação para a execução de processos de ETL.

a) Contexto: É necessário ao menos três ambientes para a execução de processos de ETL: fonte, área de processamento e destino.

b) Problema: É preciso definir que tipo de ambiente e estrutura de dados estarão disponíveis e permitirão acesso pelo framework.

c) Estrutura: Expõe sugestões para resolver o problema de definição dos ambientes, de acordo com cada requisito da estrutura dos dados. As definições de ambiente do framework proposto, para a parte de ETL, define a estrutura para os principais repositórios não relacionais e relacionais na literatura.

d) Participantes: Apresenta os elementos do ambiente e estrutura de dados.

   1. Ambiente: Define se o ambiente é fonte, destino ou área de processamento.
   2. Repositório de dados: Define os atributos do repositório de dados, bem como a forma de conexão com o ambiente, forma de manipulação dos dados e outras configurações que permitam a leitura e escrita.
   3. Esquema de dados: Define as tabelas, campos, relacionamentos, visões, índices, pacotes, procedimentos, funções, filas, gatilhos, tipos, sequências, visões materializadas, sinônimos, enlaces de banco de dados, diretórios, esquemas XML e outros elementos dos ambientes.
   4. Amostra: Define quais colunas e tipos de dados que serão utilizados para cada ambiente.

e) Próximos padrões: Após as definições de ambientes e estrutura de dados, o passo seguinte é definir os mecanismos padrões para o ETL.

3.2.4.2 Mecanismos de ETL

a) Contexto: É necessário definir mecanismos básicos para a execução de processos de ETL. Os mecanismos mais comuns encontrados na literatura (Silva, 2012) estão expostos na tabela 3.1.

Figura 3.1 Diagrama de Classe da Definição dos Ambientes e Estrutura de dados

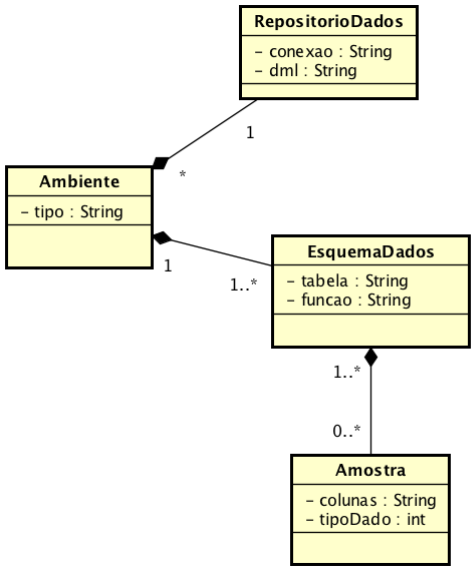


Tabela 3.3 Mecanismos básicos para o padrão de análise Mecanismos de ETL

Fase de ETL	Mecanismo Padrão
Extração	Extrair dados
Transformação	Filtrar dados Unir dados Agregar dados Juntar dados Gerar Chaves Pesquisar chaves Converter dados
Carga	Carregar dados

b) Problema: É necessário determinar quais os mecanismos fundamentais para o funcionamento dos recursos padrões de ETL. Dessa forma, determinar quais processos deverão estar disponíveis, e como utilizá-los.

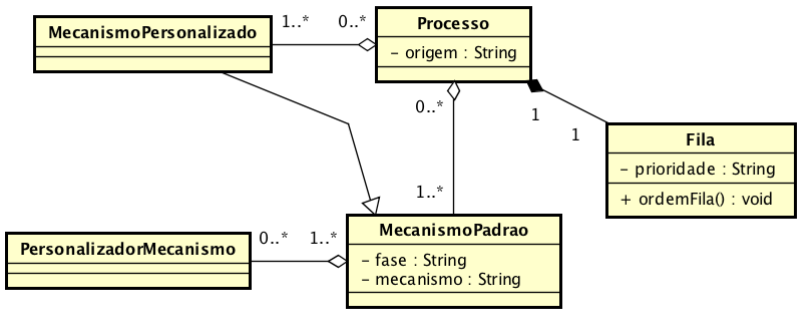
c) Estrutura: Define os mecanismos padrões utilizados pelos processos de ETL. Porém, esse conjunto de mecanismos pode não atender a necessidade de todas as aplicações, é necessário permitir a criação de mecanismos personalizados para um determinado domínio de aplicação. Além disso, deve ser possível personalizar e reutilizar os mecanismos existentes.

d) Participantes: Apresenta os elementos dos mecanismos comuns e personalizados.

1. Fila: conjunto de processos de um mecanismo em ordem de execução.
2. Processo: área de processamento, origem do processo, atributos do processo.

3. Mecanismo Padrão: mecanismos básicos mais relevantes para a execução de processos de ETL.
4. Mecanismo Personalizado: mecanismos personalizados para atender domínios específicos.
5. Personalizador de mecanismo: tem a função de permitir ao programador alterar comportamentos dos mecanismos padrões.

Figura 3.2 Diagrama de Classe Mecanismos de ETL



e) Próximos padrões: O passo seguinte à modelagem dos mecanismos padrões de ETL é a definição dos padrões de operação.

3.2.5 Padrões de Operação

Padrões de Operação são requisitos para a execução dos processos de ETL.

3.2.5.1 Leitura dos ambientes e das estrutura de dados

a) Contexto: É preciso executar a leitura dos ambientes e de estrutura de dados que farão parte dos processos de ETL. Para a execução dos mecanismos padrões é necessário que haja a definição e posteriormente a leitura dos dados.

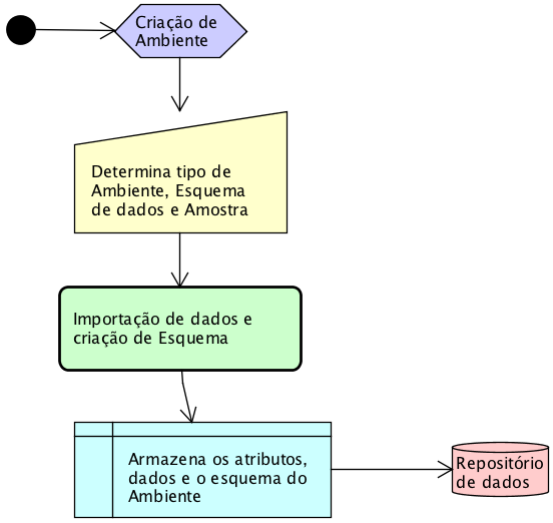
b) Problema: A leitura dos dados deve ser feita de forma que possibilite a execução dos processos de ETL por meio dos mecanismos padrões ou personalizados.

c) Estrutura: A operação de leitura é feita por meio da instanciação dos elementos:

1. Ambiente: determina o método de criação do ambiente, bem como a instanciação dos seus atributos.
2. Repositório de dados: determina o método de criação de repositório de dados, onde é possível definir quais tipos de dados estão presente no determinado ambiente.
3. Esquema de dados: determina o método de leitura do esquema dos dados do ambiente, essa leitura é feita por meio da linguagem de manipulação de dados.

4. Amostra: determina a leitura das colunas e tipos de dados de um determinado esquema de dados.

Figura 3.3 Fluxograma da criação de Ambientes



d) Participantes: apresentação os principais métodos para leitura dos ambientes e estrutura de dados.

Tabela 3.4 Métodos de Leitura de Metadados

Descrição	Método
Conexão ao Banco de dados	connectDB
Criação de Esquema de dados	createSchema
Armazena os dados	setData
Consulta os dados	getData
Criação de Amostra	createSample

e) Próximos padrões: a próxima etapa a ser definida é o padrão da execução dos processos de ETL.

3.2.5.2 Execução dos processos de ETL

a) Contexto: Por meio da definição dos mecanismos básicos dos processos de ETL é possível realizar a sua execução.

b) Problema: A execução dos processos deverá ser feita de acordo com a seu grau de prioridade definida pelo usuário.

c) Estrutura: A execução dos processos é feita por meio da definição do seu mecanismo.

d) Participantes: Mecanismos dos processos de ETL.

e) Próximos padrões: Após a execução dos processos de ETL, o passo seguinte é a definição dos padrões de análise de NoSQL para permitir o uso de banco de dados do paradigma NoSQL pelo framework.

Tabela 3.5 Padrões de Análise do ETL4NoSQL

Subsistema	Padrão de Análise	Etapa
ETL	Metadados	- Definição de Ambiente e Estrutura de dados - Mecanismos de ETL
ETL	Operação	- Leitura de Ambiente e Estrutura de dados - Execução dos mecanismos de ETL
NoSQL	Importação	- Definição de Ambiente e Estrutura de dados Mecanismos de NoSQL
NoSQL	Mapeamento	- Leitura de Ambiente e Estrutura de dados - Execução dos mecanismos NoSQL

Com isso, por meio das definições de requisitos foi possível definir os padrões de análise que estão apresentados na tabela 3.5.

3.3 Arquitetura do ETL4NoSQL

A arquitetura do Framework será apresentada nesta seção, ela foi definida baseada em componentes. Segundo [Sam97], componentes são uma parte do sistema de software que podem ser identificados e reutilizados, onde descrevem ou executam funções específicas e possuem interfaces claras, documentação apropriada e a possibilidade de reuso bem definida. Ainda de acordo com o autor, um componente deve ser autocontido, identificável, funcional, possuir uma interface, ser documentado e ter uma condição de reuso. Dessa forma, definimos os componentes do ETL4NoSQL de acordo com essas características e serão apresentados na seção 3.3.1.

3.3.1 Componentes do ETL4NoSQL

A seguir são apresentados os modelos de componentes do Framework ETL4NoSQL de acordo com os padrões de análise e as convenções estabelecidas por [Hei01].

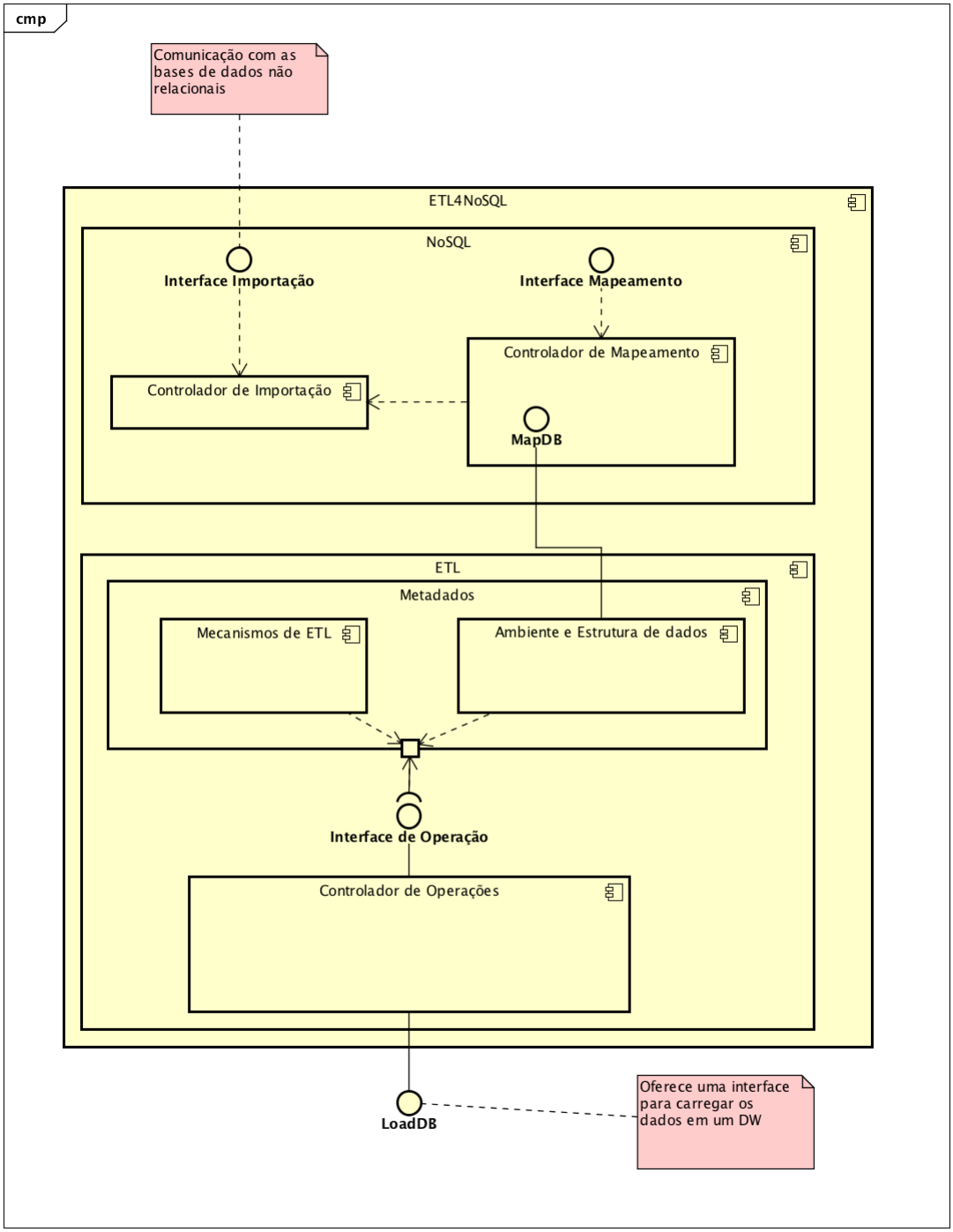


Figura 3.4 Arquitetura do Framework baseada em Componentes

3.3.1.1 Componente de Importação

a) Interface: Componente responsável pela importação dos dados da base de origem

b) Nomeação: Componente de Importação

c) Metadados: Este componente será composto com as informações da base de origem e da busca dos dados, apresentará uma interface para o usuário disponibilizar as informações e fornecerá os dados para o componente de mapeamento.

d) Interoperabilidade: Deve possibilitar a comunicação entre outros componentes

e) Composição:

Tabela 3.6 Métodos do Componente Importação

Método	Descrição	Comportamento
ConnImportDB	Conecta com o banco de dados onde os dados serão importados	Estabelece a conexão com a base de dados de importação
setDMLImportDB	Define a linguagem de manipulação de dados da base de importação	Armazena os parâmetros de manipulação de dados da base de dados de importação
queryImportDB	Define a consulta de onde os dados serão importados	Retorna os dados que serão importados mantendo suas relações, a importação pode ser Sob Demanda ou Exaustiva
getDMLImportDB	Retorna a configurações da linguagem de manipulação de dados da base de importação que foram estabelecidas	Retorna as configurações da DML da base de importação

f) Customização: Este componente pode ser customizado em sua interface com o usuário, possibilitando outras meios de fornecer as informações para os seus métodos

g) Suporte a evolução: Deve possibilitar o suporte aos métodos de acordo com as mudanças de conexões e manipulações de bases de dados futuras

h) Empacotamento e utilização: Os métodos deverão ser encapsulados e poderão ser utilizados pela importação de sua classe e a interface com o usuário será por meio de linha de comando

3.3.1.2 Componente de Mapeamento

a) Interface: Componente responsável por gerar o mapeamento dos dados oferecidos pelo componente de importação para um esquema relacional

b) Nomeação: Componente de Mapeamento

c) Metadados: Este componente deverá estabelecer comunicação com o componente de importação, apresentará uma interface para o usuário definir a regra a ser utilizada no mapeamento e fornecerá os dados para o componente de leitura e escrita de metadados

d) Interoperabilidade: Deve possibilitar a comunicação entre outros componentes

e) Composição:

Tabela 3.7 Métodos do Componente Mapeamento

Método	Descrição	Comportamento
getImportData	Busca os dados do componente de importação	Estabelece a comunicação com o componente de importação
setMapRules	Define as regras de mapeamento	Define um conjunto de regras de mapeamento para um esquema relacional
getMapRules	Retorna uma regra de mapeamento	Retorna uma regra de mapeamento do conjuntos de regras
runMap	Gera o mapeamento de acordo com a regra estabelecida	Retorna os dados no esquema relacional de acordo com a regra de mapeamento escolhida

f) Customização: É possível customizar as regras de mapeamento para outros esquemas não relacionais

g) Suporte a evolução: Deve possibilitar o suporte aos métodos de acordo com a necessidade de alterar os esquemas dos dados

h) Empacotamento e utilização: Os métodos deverão ser encapsulados e poderão ser utilizados pela importação de sua classe e a interface com o usuário será por meio de linha de comando

3.3.1.3 Componente de Leitura e Escrita de Metadados

a) Interface: Componente responsável pela criação de ambientes para a execução de processos de ETL

b) Nomeação: Componente de Leitura e Escrita de Metadados

c) Metadados: Este componente deverá estabelecer comunicação com o componente de importação caso o ambiente seja fonte de dados, deve criar um esquema para o ambiente de processamento de dados, um esquema para o ambiente de destino e uma amostra para os dados importados

d) Interoperabilidade: Deve possibilitar a comunicação entre outros componentes

e) Composição:

Tabela 3.8 Métodos do Componente de Leitura e Escrita de Metadados

Método	Descrição	Comportamento
connectDB	Estabelece conexão com o banco de dados do ambiente	Permite a conexão com as bases de dados fonte, destino e área de processamento de dados
createSchema	Cria a esquema de dados do ambiente	Permite a criação de um esquema relacional para cada tipo de ambiente
setData	Armazena os dados num determinado ambiente	Permite armazenar os dados de um determinado ambiente
getDataMap	Estabelece a comunicação com os dados mapeados	Retorna os dados que foram mapeados pelo componente de mapeamento
createSample	Cria uma amostra dos dados mapeados pelo componente de mapeamento	Retorna os dados da amostra

f) Customização: É possível customizar os ambientes de acordo com a necessidade do usuário

g) Suporte a evolução: Deve possibilitar o suporte aos métodos de acordo com a necessidade de alterar os esquemas dos dados

h) Empacotamento e utilização: Os métodos deverão ser encapsulados e poderão ser utilizados pela importação de sua classe e a interface com o usuário será por meio de linha de comando

3.3.1.4 Componente de Mecanismos de ETL

a) Interface: Componente responsável por estabelecer os mecanismos que estarão disponíveis para a execução de processos de ETL

b) Nomeação: Componente Mecanismos de ETL

c) Metadados: Este componente deverá estabelecer comunicação com o componente de Leitura e Escrita de Metadados, apresentará uma interface para o usuário definir quais mecanismos a serem utilizados, bem como manterá os mecanismos que estarão disponíveis para o uso

d) Interoperabilidade: Deve possibilitar a comunicação entre outros componentes

e) Composição:

f) Customização: É possível customizar e criar mecanismos de acordo com a necessidade de cada processo de ETL

Tabela 3.9 Métodos do Componente Mecanismos de ETL

Método	Descrição	Comportamento
mechanismType	Define o tipo de mecanismo	Estabelece a descrição de um mecanismo
mechanismPhase	Define a fase a qual o mecanismo pertence	Estabelece a descrição da fase a qual o mecanismo pertence
setParams	Determina os parâmetros utilizados pelo mecanismo	Estabelece conexão com o componente de metadados e define os parâmetros que serão executados pelo mecanismo
exec	Executa o mecanismo	Executa o comportamento do mecanismo de acordo com o parâmetros estabelecidos
outPutSchema	Retorna o esquema criado pela a execução do mecanismo	Retorna o esquema resultado da execução do mecanismo

g) Suporte a evolução: Deve possibilitar o suporte aos métodos de acordo com a necessidade de alterar os esquemas dos dados

h) Empacotamento e utilização: Os métodos deverão ser encapsulados e poderão ser utilizados pela importação de sua classe e a interface com o usuário será por meio de linha de comando

3.3.1.5 Componente de Operações

a) Interface: Componente responsável por criar e executar processos de ETL

b) Nomeação: Componente de Operação

c) Metadados: Este componente deverá possibilitar a comunicação com o componente de metadados e o componente de mecanismos de ETL e deverá criar e executar processos de ETL

d) Interoperabilidade: Deve possibilitar a comunicação entre outros componentes

e) Composição:

f) Customização: É possível customizar os processos de ETL criados

g) Suporte a evolução: Deve possibilitar o suporte aos métodos de acordo com a necessidade de alterar os processos

h) Empacotamento e utilização: Os métodos deverão ser encapsulados e poderão ser utilizados pela importação de sua classe e a interface com o usuário será por meio de linha de comando

Tabela 3.10 Métodos do Componente Operações

Método	Descrição	Comportamento
createProcess	Cria um processo de ETL	Estabelece comunicação entre o componente de metadados e o componente de mecanismos de ETL
createEnviroment	Cria um ambiente para executar o processo de ETL	Cria um ambiente fonte, APD ou Destino
execMechanism	Executa um mecanismo	Executa um mecanismo do componente mecanismos de ETL de acordo com os parâmetros estabelecidos
execProcess	Executa o processo de ETL criado	Retorna o resultado do processo de ETL
loadDB	Carrega o resultado do processo de ETL para um ambiente destino	Armazena os dados do resultado do processo no banco de dados destino

3.3.2 Implementação dos componentes do ETL4NoSQL

3.4 Considerações finais

CAPÍTULO 4

Estudo de Caso

Para a aplicação da representação da modelagem do processo de ETL observar Trujillo and Mora (2003) e Lujan-Mora and Trujillo (2003) referenciado na dissertação de Mario.

4.1 Descrição

4.2 Considerações Finais

CAPÍTULO 5

Conclusão

5.1 Principais Contribuições

5.2 Discussão

5.3 Resultados

5.4 Trabalhos Futuros

Referências Bibliográficas

[dS12] Mário Sergio da Silva. Um framework para desenvolvimento de sistemas etl. Master's thesis, Universidade Federal de Pernambuco, September 2012.

[Hei01] George T. Heineman. *Component-Based Software Engineering: putting pieces together*. Addison-Wesley, 2001.

[KR02] Ralph Kimball and Margy Ross. *The Data Warehouse Toolkit*. Robert Ipsen, 2002. Second Edition.

[LTP13] Xiufeng Liu, Christian Thomsen, and Torben Bach Pedersen. Cloudetl: Scalable dimensional etl for hive. *DB Tech Reports*, July 2013.

[Nas12] Petter Nasholm. *Extracting Data From NoSQL Databases*. PhD thesis, Chalmers University of Technology, SE-412 96 Goteborg Sweeden, January 2012.

[Sam97] Johannes Sametinger. *Software Engineering with Reusable Componets*. Springer, 1997.