Pós-Graduação em Ciência da Computação

ETL4NOSQL: UM FRAMEWORK DE ETL PARA BDS NOSQL

Por

CARINE CALIXTO AGUENA

Dissertação de Mestrado

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
www.cin.ufpe.br/~posgraduacao

RECIFE/2017

Carine Calixto Aguena

ETL4NOSQL: UM FRAMEWORK DE ETL PARA BDS NOSQL

Trabalho apresentado ao Programa de Pós-graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Doutor em Ciência da Computação.

Orientador: Valéria Cesário Times

RECIFE
2017

Carine Calixto Aguena
ETL4NoSQL: Um framework de ETL para BDs NoSQL/ Carine Calixto Aguena. – RECIFE, 2017-
54 p. : il. (algumas color.) ; 30 cm.

Orientador Valéria Cesário Times

Dissertação de Mestrado – Universidade Federal de Pernambuco, 2017.

1. Palavra-chave1. 2. Palavra-chave2. I. Orientador. II. Universidade xxx. III. Faculdade de xxx. IV. Título

CDU 02:141:005.7

Dissertação de Mestrado apresentada por **Carine Calixto Aguena** ao programa de Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, sob o título **ETL4NoSQL: Um framework de ETL para BDs NoSQL**, orientada pelo **Prof. Valéria Cesário Times** e aprovada pela banca examinadora formada pelos professores:

_____

Prof.
Centro de Informática/UFPE

_____

Prof.
Centro de Informática/UFPE

_____

Prof.
Departamento de Ciência da Computação/UFBA

_____

Prof.
Instituto de Computação/UFF

_____

Prof.
Departamento de Estatística e Informática/UFRPE

RECIFE
2017

I dedicate this thesis to all my family, friends and professors
who gave me the necessary support to get here.

Agradecimentos

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

I have yet to see any problem, however complicated, which, when looked at in the right way, did not become still more complicated.

—POUL ANDERSON

Resumo

O gerenciamento eficiente de solicitações de mudança (SM) é fundamental para o sucesso das atividades de manutenção e evolução de software. Entretanto, a atribuição de SMs a desenvolvedores de software é um aspecto custoso desse gerenciamento, pois demanda tempo e requer conhecimento apropriado do projeto de software. Com o propósito de diminuir esse custo, várias pesquisas já propuseram métodos de atribuição automática de SMs. Embora representem avanços na área, existem vários fatores inerentes a atribuição de SMs que não são considerados nessas pesquisas e são essenciais para a automação.

Como demonstrado nesse trabalho, a atribuição automática deve, por exemplo, considerar a carga de trabalho, a experiência e o conhecimento dos desenvolvedores, a prioridade e a severidade das SMs, a afinidade dos desenvolvedores com os problemas descritos nas SMs, e até mesmo os relacionamentos interpessoais. Para tornar esse cenário ainda mais complexo, esses fatos podem variar de acordo com o projeto de software que está sendo desenvolvido. Assim, uma solução para o problema de atribuição de SMs depende de informações contextuais.

Assim, esse trabalho propõe, implementa e valida uma solução arquitetural sensível ao contexto para atribuição automática de SMs. Dado o aspecto contextual da solução, a arquitetura enfatiza a necessidade de considerar as diversas fontes de informações presentes na organização, assim como a necessidade de se desenvolver algorítimos que implementem diferentes estratégias de atribuição. A proposta e implementação dessa solução é embasada em resultados de duas pesquisas quantitativas: um estudo de mapeamento sistemático da literatura, e uma pesquisa de questionário com desenvolvedores de software. Esse último forneceu um conjunto de requisitos que a solução automatizada deve satisfazer para que as estratégias de atribuição sejam atendidas, enquanto o mapeamento da literatura identificou técnicas, algoritmos, e outros requisitos necessários a automação.

A implementação da arquitetura segue uma estratégia de automação, também elaborada nesse trabalho, que possui dois componentes principais: um sistema especialista baseado em regras (SEBR); e um modelo de recuperação de informação (MRI) com técnicas de aprendizagem. Em nossa estratégia, esses dois componentes são executados alternadamente em momentos diferentes a fim de atribuir uma SM automaticamente. O SEBR processa regras simples e complexas, considerando informações contextuais do projeto de software e da organização que o desenvolve. O MRI é utilizado para fazer o casamento entre SMs e desenvolvedores de acordo com o histórico de atribuições.

Palavras-chave: Engenharia de Software, Manutenção e Evolução de Software, Gerenciamento de Solicitações de Mudança, Atribuição Automática de Solicitações de Mudança

Abstract

The efficient management of Change Requests (CRs) is fundamental for successful software maintenance; however the assignment of CRs to developers is an expensive aspects in this regard, due to the time and expertise demanded. To overcome this, researchers have proposed automated approaches for CR assignment. Although these proposals present advances to this topic, they do not consider many factors inherent to the assignments. Indeed, different complex factors may have influence on CR assignment, and most of them vary from one organization to another. For instance, developers' workload, CRs severity, interpersonal relationships, or developers know-how must be considered in the assignments. Actually, as we demonstrate in this work, CR assignment is a complex activity and automated approaches cannot rely on simplistic solutions. Ideally, it is necessary to consider and reason over contextual information in order to provide an effective automation.

In this regarding, this work proposes, implements, and validates a context-aware architecture to automate CR assignment. The architecture emphasizes the need for considering the different information available at the organization to provide a more context-aware solution to automated CR assignment. The development of such architecture is supported by evidence synthesized from two empirical studies: a survey with practitioners and a systematic mapping study. The survey provided us with a set of requirements that automated approaches should satisfy. In the mapping study, in turn, we figured out how state-of-the-art approaches are implemented in regarding to these requirements, concluding that many of them are not satisfied. In addition, new requirements were identified in this mapping study.

For the implementation of the proposed architecture, we developed a strategy to automate CR assignments which is based on two main components: a Rule-Based Expert System (RBES) and an Information Retrieval (IR) model. The strategy coordinately applies these two components in different steps to find the potential developer to a CR. The RBES takes care of the simple and complex rules necessary to consider contextual information in the assignments, e.g., to prevent assigning a CR to a busy or unavailable developer. Since these rules vary from one organization/project to another, the RBES facilitates their modification for different contexts. On the other hand, the IR model is useful to make use of the historical information of CR assignments to match CRs and developers.

Keywords: Software Engineering, Software Maintenance and Evolution, Change Request Management, Automatic Change Request Assignment

Lista de Figuras

Lista de Quadros

Lista de Acrônimos

Sumário

1

Introdução

Este capítulo contextualiza os principais assuntos abordados neste trabalho de dissertação, apresenta as motivações que levaram à escolha do tema, os objetivos gerais e específicos da proposta desta pesquisa, bem como a justificativa para conduzir uma investigação no assunto debatido e suas principais contribuições.

1.1 Contextualização

Desde a década de 1970, com a criação do modelo relacional por Edgar Frank Codd, a estrutura de armazenamento adotada por muitos desenvolvedores de sistemas da área de tecnologia da informação tem se baseado no conceito de entidade e relação proposto por Codd. A maioria dos sistemas gerenciadores de banco de dados que possui aceitação no mercado fazem uso desse modelo, por exemplo o MySQL, Oracle e Microsoft SQL Server. Porém, os requisitos para o desenvolvimento de ferramentas de software modernas têm mudado significativamente, especialmente com o aumento das aplicações Web [NASHOLM (2012)]. Este segmento de aplicações exige requisitos com alta escalabilidade e vazão, onde sistemas que utilizam um armazenamento com esquema relacional não conseguem atender satisfatoriamente. Em resposta a isso, novas abordagens de armazenamentos de dados utilizando o termo de NoSQL tornaram-se popular [SILVA (2016)].

O termo NoSQL é constantemente interpretado como "*Not Only SQL*", cujo SQL refere-se a linguagem de manipulação de dados dos gerenciadores de armazenamento de dados relacionais (RDBMS - Relational Database Management System) - Structure Query Language [NASHOLM (2012)]. O grande propósito das abordagens NoSQL é oferecer alternativas onde os esquemas relacionais não apresentam um bom desempenho. Esse termo abrange diferentes tipos de sistemas. Em geral, banco de dados NoSQL usam modelo de dados não-relacionais, com poucas definições de esquema, são executados em clusters e aplicados a alguns bancos de dados recentes como o Cassandra, o Mongo, o Neo4J e o Riak [FOWLER; SADALAGE (2013)].

Muitas empresas coletam e armazenam milhares de gigabytes de dados por dia, no qual a análise desses dados torna-se uma vantagem competitiva no mercado. Por isso, há uma grande necessidade de uma nova arquitetura para o gerenciamento de suporte à decisão que possa alcançar melhor escalabilidade e eficiência [LIU; THOMSEN; PEDERSEN (2013)]. Para auxiliar no processo de gerenciamento de suporte à decisão uma das formas mais utilizadas é a criação de um ambiente data warehousing, que é responsável por providenciar informações estratégicas e esquematizadas a respeito do negócio [CHAUDHURI; DAYAL (1997)].

Segundo a definição de KIMBALL; ROSS (2002), data warehouse (DW) é uma coleção de dados para o processo de gerenciamento de suporte à decisão orientado a assunto, integrado, variante no tempo e não volátil. Os dados de diferentes fontes de sistemas são processados em um data warehouse central através da Extração, Transformação e Carga (ETL) de maneira periódica. Os processos de ETL consistem em um conjunto de técnicas e ferramentas para transformar dados de múltiplas fontes de dados para fins de análise de negócio [SILVA (2016)]. Ferramentas de ETL são sistemas de software responsáveis por extrair dados de diversas fontes, transformar e customizar os dados e inseri-los no data warehouse. Comumente, esses processos são executados periodicamente, onde a otimização do seu tempo de execução torna-se importante [VASSILIADIS et al. (2005), SILVA (2016)].

O projeto de ETL consome cerca de 70% dos recursos de implantação de um DW, pois

desenvolver esse projeto é crítico e custoso, tendo em vista que gerar dados incorretos pode acarretar em más decisões. Porém, por algum tempo pouca importância foi dada ao processo de ETL pelo fato de ser visto somente como uma atividade de suporte aos projetos de DW. Apenas a partir do ano 2000, a comunidade acadêmica passou a dar mais importância ao tema[(SILVA (2012)].

Tradicionalmente, o DW é implementado em uma base de dados relacional, onde o dado é armazenado nas tabelas fato e tabelas dimensões, na qual forma um esquema em estrela [KIMBALL; ROSS (2002)]. Por isso, é comum que as ferramentas de ETL utilizadas no mercado atualmente demonstrem mais importância aos esquemas relacionais. Para oferecer suporte aos sistemas que necessitem utilizar um esquema não relacional de BDs NoSQL em DW, a proposta desse trabalho é especificar um framework programável, flexível e integrado para modelagem e execução de processos ETL em BDs NoSQL.

1.2 Motivação

A integração de dados e os processos de ETL são procedimentos cruciais para a criação de data warehouses e sistemas BI (*business intelligence*). Porém, os sistemas para ETL e integração de dados são tradicionalmente desenvolvidos para dados estruturados em modelos relacionais que representam apenas uma pequena parte dos dados mantidos por muitas empresas [DARMONT et al. (2005), Russom 2007, Pedersen 2009]. Dessa forma, existe uma demanda crescente para integrar os dados não estruturados e semi estruturados em um repositório unificado. Devido a complexidade desses dados, novos desafios estão surgindo quando lidamos com dados heterogêneos e distribuídos no ambiente de integração [Salem, 2012].

Além disso, muitas empresas encontram dificuldades para lidar com as ferramentas ETL disponíveis no mercado. Aprender a lidar com essas ferramentas pode ser muito custoso em termos financeiros e de tempo, e por isso, acabam optando desenvolver os seus processos por meio de uma linguagem de programação de propósito geral [Awad et al., 2011; Muñoz et al., 2009].

Portanto, este trabalho propõe um framework programável para desenvolvimento de sistemas de ETL que possibilita a integração de dados estruturados, não estruturados e semi estruturados armazenados em bases relacionais ou NoSQL. O framework possui um ambiente integrado para a importação e mapeamento dos dados, além da modelagem e customização dos processos de ETL. Os processos de importação e mapeamento do framework integram dados estruturados, não estruturados e semi estruturados. Esses processos possibilitam a leitura e manipulação de dados de bases NoSQL, e também o armazenamento desses dados em bases deste tipo, oferecendo uma alternativa não relacional para a construção de DWs.

Uma alternativa para organizar e manipular grandes volumes de dados sem utilizar um modelo relacional e ainda processá-los e armazená-los de maneira distribuída é fazer o uso de BDs NoSQL [CARVALHO SCABORA (2016)]. Com isso, surge a necessidade de se promover

meios para o uso desses BDs em DWs.

As pesquisas presentes na literatura sobre extração de dados em BDs NoSQL mostram que não há uma ferramenta que seja integrada para o uso de BDs NoSQL, as ferramentas existentes no mercado apenas oferecem a possibilidade para alguns SGBDs NoSQL, ficando a cargo da equipe de implantação do projeto de DW todo o trabalho de modelagem e programação ao se utilizar BDs NoSQL [SILVA (2016), CHEVALIER et al. (2015), LIU; THOMSEN; PEDERSEN (2013)].

SILVA (2012) aponta em sua pesquisa que muitas empresas evitam ferramentas de ETL disponíveis no mercado, e adotam o desenvolvimento dos processos a partir de uma linguagem de programação de propósito geral, pelo fato dessas ferramentas terem uma longa curva de aprendizagem e grande complexidade no seu uso.

O aumento do uso de banco de dados com esquemas não relacionais baseados no paradigma NoSQL e a falta de uma ferramenta programável, flexível e integrada, independente de plataforma que dê suporte à extração, transformação e carga em data warehouses para esses esquemas é a grande motivação deste trabalho.

Dessa forma, encontrar uma solução que seja programável, flexível e integrada para extração, transformação e carga dos dados em BDs NoSQL é a proposta deste trabalho.

1.3 Objetivos

O objetivo principal desta pesquisa é especificar um framework programável, flexível e integrado para modelagem e execução de processos ETL de banco de dados estruturados, não estruturados e semi estruturados sob os modelos relacionais e NoSQL. Os objetivos específicos são detalhados a seguir.

1.3.1 Objetivo Específico

Este trabalho de dissertação tem como um dos objetivos específicos apresentar os componentes do framework ETL4NoSQL, bem como suas funcionalidades. Outro objetivo deste trabalho é realizar um estudo experimental de software a fim de caracterizar as principais funcionalidades das ferramentas de ETL na manipulação de dados estruturados, semi estruturados e não estruturados. O estudo experimental poderá comparar o framework proposto, suas vantagens e desvantagens, em relação às ferramentas de ETL encontradas na literatura.

1.4 Contribuições

Uma das contribuições deste trabalho é fornecer um framework programável, flexível e integrado que auxilia na modelagem e execução dos processos de ETL em bases de dados estruturadas, semi estruturadas e não estruturadas, denominado ETL4NoSQL. Assim, é possível extrair, integrar e carregar dados que estão armazenados em diversas estruturas como é o caso

dos bancos de dados NoSQL, ou até mesmo, repositórios de dados textuais e banco de dados relacionais em um único repositório. O ETL4NoSQL é um recurso valioso, principalmente para os desenvolvedores responsáveis pela fase de ETL, onde muitos encontram dificuldades para lidar com as ferramentas ETL disponíveis no mercado.

Outra contribuição desta pesquisa é apresentar, por meio de um estudo experimental, as principais características, de acordo com algumas ferramentas de ETL presentes na literatura, bem como possíveis melhorias, vantagens e desvantagens, em suas funcionalidades.

1.5 Organização do Trabalho

Este trabalho está organizado de acordo com a seguinte estrutura:

- **Fundamentação Teórica:** apresenta uma revisão da literatura dos principais assuntos abordados neste trabalho. Serão tratados temas a respeito de ETL, banco de dados NoSQL, Frameworks e estudo experimental de software.
- **Trabalhos Correlatos:** descreve os trabalhos correlatos encontrados na literatura a respeito de ferramentas de ETL. Este capítulo visa um comparativo das diversas ferramentas existentes com a proposta deste trabalho.
- **O Framework ETL4NoSQL:** descreve os requisitos, arquitetura e componentes do framework exposto neste trabalho.
- **Estudo Experimental de Software:** expõe o roteiro da experimentação de software para ferramentas de ETL. Define o objetivo, planejamento, operação e resultado do estudo.
- **Considerações Finais:** expressa as limitações e ameaças à validade do trabalho, considerações finais e sugere de trabalhos futuros.

2

Fundamentação Teórica

Neste capítulo são apresentados os conceitos relacionados ao desenvolvimento desta pesquisa, bem como o embasamento teórico necessário para o entendimento do estudo. Os temas abordados são: ETL, Banco de Dados NoSQL, Frameworks, Estudo Experimental de Software e trabalhos correlatos ao tema deste trabalho.

2.1 Conceitos Básicos

Conceitos de ETL, banco de dados NoSQL, *frameworks* e estudo experimental de software são essenciais para esta pesquisa. Dessa forma, os princípios básicos desses conceitos são apresentados nesta seção. Ainda que estudo experimental de software não seja o tema principal deste trabalho, seus conceitos são essenciais para avaliar e medir nossa proposta, sendo assim é fundamental entender seus conceitos.

2.1.1 ETL

ETL sigla para *Extraction, Transform and Load* (Extração, Limpeza/Transformação e Carga) é conhecido na literatura por definir processos que permitem a integração de dados, centralizando-os numa base destino facilitando o gerenciamento e análise dos dados [KIMBALL; CASERTA (2004), RUD (2009)]. O fluxo do processo de ETL inicia-se com extração dos dados a partir de uma fonte, que podem ser arquivos textuais, banco de dados relacionais ou banco de dados NoSQL. Os dados são propagados para uma Área de Processamento de Dados onde são executadas a limpeza e transformação por meio de mecanismos de ETL definidos como agregação, junção, filtro, união, entre outros. Finalmente, os dados são carregados em estruturas que podem ser data warehouses ou repositórios analíticos [SILVA (2016)].

KIMBALL; CASERTA (2004) definem ETL em quatro macroprocessos, com 34 subsistemas. Os quadros 2.1, 2.2, 2.3 e 2.4 mostram os subsistemas do processo de ETL. Os quatro macroprocessos são:

- Extração: Recolhe os dados dos sistemas de origem e grava na área de processamento de dados antes de qualquer reestruturação significativa. Esta etapa possui 3 subsistemas.
- Limpeza e Transformação: Envia os dados de origem, por meio de várias etapas de processamento no sistema ETL. Melhora a qualidade dos dados recebidos da fonte, mescla dados de duas ou mais fontes para criar e aplica dimensões e métricas. Esta etapa possui 5 subsistemas.
- Entrega ou Carga: Estrutura fisicamente e carrega os dados conforme desejado em DWs ou repositórios analíticos. Esta etapa possui 13 subsistemas.
- Gerenciamento: Gerencia os sistemas e processos relacionados ao ambiente ETL de forma coerente. Esta etapa possui 13 subsistemas.

Quadro 2.1 Subsistemas do macroprocesso de extração do processo de ETL

Macroprocesso	subsistema
Extração	**Data Profiling:** Explora uma origem de dados para determinar seu ajuste para inclusão como uma fonte associado à limpeza e ajuste de requisitos. **Change Data Capture:** Isola as mudanças ocorridas nos sistemas de origem, de forma a reduzir os processos de ETL - Carga Incremental. **Sistema de Extração:** Extração e movimentação dos dados de origem para dentro do DW, para processamento futuro.

Quadro 2.2 Subsistemas do macroprocesso de limpeza e extração do processo de ETL

Macroprocesso	subsistema
Limpeza e Transformação	**Data Cleasing System - Sistema de Limpeza de Dados:** Implementa processos de qualidade de dados para identificar violações de qualidade. **Error Event Tracking - Acompanhamento de erro:** Captura todos os 'eventos de erro', que serão as entradas vitais para a melhoria da qualidade dos dados. **Criação de Dimensão de auditoria:** Junta Metadados para cada Tabela Fato, como uma dimensão. Este Metadados estará disponível para a geração de aplicações de BI que visualizem a qualidade dos dados. **Deduplication - Tirar a duplicidade de dados:** Elimina dados redundantes de dimensões, como clientes ou produtos. Pode requerer integração cruzada entre multiplas origens e a aplicação de regras para identificar qual a versão mais correta de uma linha duplicada. **Data Conformance - conformidade de dados:** Força o uso de atributos comuns entre as principais Conformed Dimensions versus as métricas comuns nas Tabelas Fato relacionadas.

Quadro 2.3 Subsistemas do macroprocesso de entrega ou carga do processo de ETL

Macroprocesso	subsistema
Entrega ou Carga	**Slowly Changing Dimension (SCD) Manager:** Implementa a lógica para os atributos SCD. **Surrogate Key Generator:** Cria as chaves substitutas (chaves de negócio) - surrogate keys independentes para cada dimensão. **Hierarchy Manager:** Entrega multipla e simultânea de estruturas hierarquicas na dimensão. **Special Dimensions Manager:** Cria locais - placeholders na estrutura de ETL para sustentar os processos repetitivos específicos da organização, no desenho de dimensões específicas coma as Junk Dimensions, Mini Dimensions e indicadores de comportamento. **Fact Table Builders:** Construção dos três tipos básicos de tabela fato: Transacional, Periódico e Cumulativo (transaction grain, periodic snapshot e accumulating snapshot). **Surrogate Key Pipeline:** Substitui, nas dimensões, a chave natural operacional das tabelas de origem pelas chaves substitutas (Surrogate Key) que serão utilizadas para o relacionamento com as tabelas fato. **Multi-Valued Bridge Table Builder:** Construção e Manutenção das tabelas ponte (bridge tables) para suportar os relacionamentos multi-valorados. **Late Arriving Data Handler:** Aplica modificações especiais nas procedures do processo padrão para lidar com tabelas fato recém definidas (late-arriving) e dimensões. **Dimension Manager:** Centraliza a autoridade para preparar e divulgar as dimensões conforme (conformed dimensions) para a comunidade do Data Warehouse. **Fact Table Provider:** Detém a administração de uma ou mais tabelas fato, e a responsabilidade de criação, manutenção e uso. **Aggregate Builder:** Construção e manutenção de agregações que serão usadas de forma continua com tecnologias de navegação agregada para melhorar a performance das consultas. **OLAP Cube Builder:** Seleciona os dados do esquema dimensional para popular os cubos OLAP. **Data Propagation Manager:** Prepara dados conformados e integrados no servidor de apresentação do Data Warehouse, para entrega em outros ambientes, para propósitos especiais.

Quadro 2.4 Subsistemas do macroprocesso de gerenciamento do processo de ETL

Macroprocesso	subsistema
Gerenciamento	**Job Scheduler:** A estratégia de gerenciamento da execução dos ETLs deve ser confiável, incluindo os relacionamentos e dependências entre os ETLs. **Backup System:** Mantem cópia do ambiente de ETL para propósito de recuperação, restart e arquivamento. **Recovery and Restart:** Processos para recuperação do ambiente de ETL ou processo de reinicio, em caso de eventuais falhas. **Version Control:** Mantem arquivadas versões dos ETLs, para eventual recuperação das lógicas e metadados do 'ETL pipeline'. **Version Migration:** Migração de uma versão completa do 'ETL pipeline' a partir do ambiente de desenvolvimento para um ambiente de testes e, finalmente, para o ambiente de produção. **Workflow Monitor:** Garante que os processos de ETL estão sendo eficientemente executados e que as cargas iniciem precisamente nas janelas de tempo estipuladas. **Sorting:** Garante a fundamental alta performance nos grupos de processos de ETLs. **Lineage and Dependency:** Identifica a origem dos dados, as localizações intermediarias, as transformações e o dado final, permitindo acompanhar de forma estruturada, a trajetória dos dados até a sua carga no Data Warehouse. **Problem Escalation:** Estrutura de suporte que encaminha os problemas encontrados nos processos de ETLs (erros) para o nível de solução apropriado. **Paralleling and Pipelining:** Habilita ao sistema de ETL a potencializar automaticamente o uso de recursos como múltiplos processadores ou computação em grade (grid computing) para entregas dentro dos prazos restritos. **Security:** Garante o acesso autorizado aos ETLs e Metadados, de forma individual ou em grupos, mantendo um histórico dos acessos. **Compliance Manager:** Suporta os requerimentos organizacionais de conformidade, através, tipicamente, da manutenção da custódia da cadeia de dados e do acompanhamento dos acessos aos dados (quem teve o acesso autorizado ao dado). **Metadata Repository:** Captura os metadados do ETL, incluindo os metadados de processo, metadados técnicos e metadados do negócio que significam todos os metadados do ambiente de DW/BI.

2.1.2 Bancos de Dados NoSQL

Consistem em bancos de dados não relacionais projetados para gerenciar grandes volumes de dados e que disponibilizam estruturas e interfaces de acesso simples (Lima; Mello, 2015). Cada SGBD (Sistema Gerenciador de Banco de Dados) NoSQL possui um esquema de modelagem diferente, nos quais são divididas pela literatura em quatro categorias amplamente usadas: Chave-Valor, Orientado a Documentos, Famílias de Colunas e Baseado em Grafos [FOWLER; SADALAGE (2013), KAUR (2013)].

As principais características dos banco de dados NoSQL são: distribuído, escalabilidade horizontal, construído para grande volume de dados, BASE ao invés de ACID, modelo de dados não relacional, não suporta SQL [FOWLER; SADALAGE (2013), NASHOLM (2012)].

2.1.2.1 Banco de dados Orientados à Documentos

Banco de dados orientados a documentos são capazes de armazenar documentos como dado. Esses documentos podem ser em qualquer formato como XML (eXtensible Markup Language), YAML (Yet Another Markup Language), JSON (JavaScript Object Notation), entre outros. Os documentos são agrupados na forma de coleções. Comparando com banco de dados relacional, as coleções são como tabelas e os documentos como os registros. Porém, a diferença entre eles é que cada registro na tabela do banco relacional tem o mesmo número de campos, enquanto que na coleção do banco de dados orientado a documentos, podem ter campos completamente diferentes [KAUR (2013)].

Existem mais de 15 banco de dados orientados a documentos disponíveis e os mais utilizados são MongoDB, CouchDB e o RavenDB [KAUR (2013)].

2.1.2.2 Banco de dados Famílias de Colunas

Banco de dados baseados em Famílias de Colunas são desenvolvidos para abranger três áreas: número enorme de colunas, a natureza esparsa dos dados e frequentes mudanças no esquema. Os dados em Famílias de colunas são armazenados em colunas de forma contínua, enquanto que em bancos de dados relacionais as linhas é que são contínuas. Essa mudança faz com que operações como agregação, suporte para ad-hoc e consultas dinâmicas se tornem mais eficientes [KAUR (2013)].

A maioria dos bancos de dados baseados em Famílias de Colunas são também compatíveis com o framework MapReduce, no qual acelera o processamento de enorme volume de dados pela distribuição do problema em um grande número de sistemas. Os bancos de dados de Família de Colunas open-source mais populares são Hypertable, HBase e Cassandra [KAUR (2013)].

2.1.2.3 Banco de dados Baseado em Grafos

Bancos de dados baseado em Grafos são como uma estrutura de rede contendo nós e arestas, onde as arestas interligam os nós representando a relação entre eles. Comparando com o modelo Entidade-Relacionamento, o nó corresponde à entidade, a propriedade do nó à um atributo, a relação entre as entidades ao relacionamento entre os nós. Nos bancos de dados relacionais as consultas requerem atributos de mais de uma tabela resultando numa operação de junção, por outro lado, bancos de dados baseado em Grafos são desenvolvidos para encontrar relações dentro de uma enorme quantidade de dados rapidamente, tendo em vista que não é preciso fazer junções, ao invés disso, ele fornece indexação livre de adjacência KAUR (2013).

2.1.2.4 Banco de dados Chave-Valor

Em Bancos de dados Chave-Valor os dados são organizados como uma associação de vetores de entrada consistindo em pares de chave-valor. Cada chave é única e é usada para recuperar os valores associados a ele. Esses bancos de dados podem ser visualizados como um banco de dados relacional contendo múltiplas linhas e apenas duas colunas: chave e valor. Buscas baseadas em chaves resultam num baixo tempo de execução, além disso, os valores podem ser qualquer coisa como objetos, hashes, entre outros [KAUR (2013)].

Os bancos de dados Chave-Valor mais populares são Riak, Voldemort e Redis [KAUR (2013)].

2.1.3 Frameworks

Frameworks podem ser considerados aglomerados de softwares, onde estes são capazes de serem estendidos e adaptados para utilidades específicas [TALIGENT (1994)]. PREE; SIKORA (1997), consideram que *frameworks* são aplicações semi-completas e que podem ser reutilizadas para especializar produtos de software customizados. SOMMERVILLE (2013), ressalta que *framework* é uma estrutura genérica estendida com o intuito de criar uma aplicação mais específica e SCHIMIDT; GOKHALE; NATARAJAN (2004) define como sendo um conjunto de artefatos de software (como classes, objetos e componentes) que colaboram para fornecer uma arquitetura reusável.

Os *frameworks* possibilitam a reusabilidade de projeto, bem como ao reúso de classes específicas, pois fornecem uma arquitetura de esqueleto para a aplicação, que é definida por classes de objetos e suas interações. As classes são reusadas diretamente e podem ser estendidas usando-se recursos, como a herança SOMMERVILLE (2013).

E.; C. (1997), separam os *frameworks* em três principais classes: de infraestrutura de sistema, de integração de *middleware* e de aplicações corporativas. *Frameworks* de infraestrutura de sistema apoiam o desenvolvimento de infraestruturas, como comunicações, interfaces de usuários e compiladores. Já os *frameworks* de integração de *middleware* são um conjunto de normas e classes de objetos associados que suportam componentes de comunicação e troca de

informações. E finalmente, os *frameworks* de aplicações corporativas estão relacionados com domínios de aplicação específicos, como sistemas financeiros. Eles incorporam conhecimentos sobre o domínios de aplicações e apoiam o desenvolvimento para o usuário final.

Muitas vezes, os *frameworks* são implementações de padrões de projeto, como por exemplo o *framework* MVC (Model-View-Control). A natureza geral dos padrões e o uso de classes abstratas e concretas permitem a extensibilidade SOMMERVILLE (2013).

Para estender um *framework* não é necessário alterar o seu código, apenas é preciso adicionar classes concretas que herdam operações de classes abstratas. Ademais, há a possibilidade de definir *callbacks*, que são métodos chamados em resposta a eventos reconhecidos pelo *framework*.Esses métodos são reconhecido como 'inversão de controle' (SCHIMIDT; GOKHALE; NATARAJAN (2004)). A figura 2.1 expressa o funcionalidade da inversão de controle. Os responsáveis pelo controle no sistema são os objetos do *framework*, ao invés de serem objetos específicos de aplicação. E em resposta aos eventos de interface do usuário, banco de dados, entre outros, esses objetos do *framework* invocam 'métodos hook' que, em seguida, são vinculados à funcionalidade fornecida ao usuário. A funcionalidade específica de aplicação responde ao evento de forma adequada. Por exemplo, um *framework* terá um método que lida com um toque em uma tecla a partir do ambiente. Esse método chama o método *hook*, que deve ser configurado para chamar os métodos de aplicação adequada para tratar o toque na tecla (SOMMERVILLE (2013)).

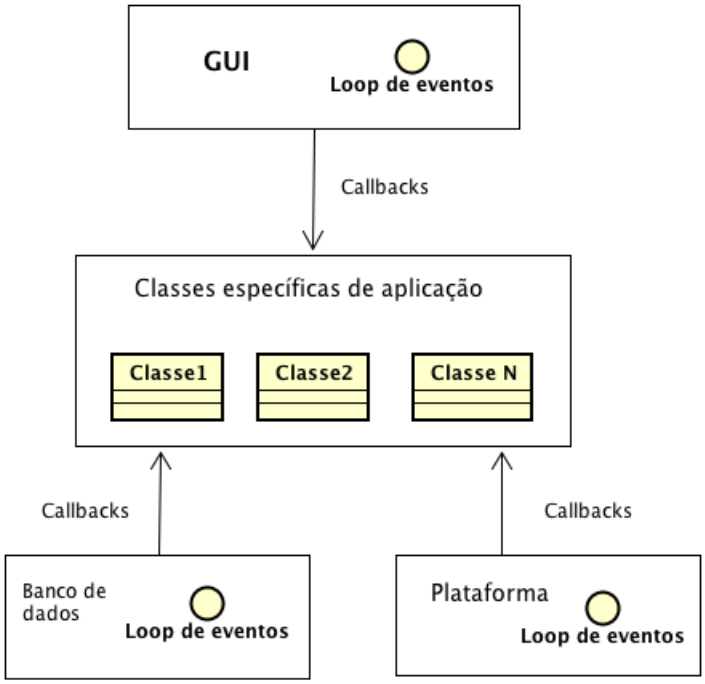

Figura 2.1 Inversão de controle em *framework*. (Adaptado de SOMMERVILLE (2013))

O *framework* ETL4NoSQL encaixa-se na categoria de aplicações corporativas, pois serve

como base para aplicações de ETL, incorporando conhecimentos sobre a área de domínio para apoiar o desenvolvimento ao usuário final.

2.1.4 Estudo Experimental de Software

Esta pesquisa de dissertação considera a execução do estudo experimental de software para caracterizar, avaliar e propor melhorias ao framework ETL4NoSQL. O objetivo principal da aplicação do experimento é definir se o *framework* proposto é uma ferramenta adequada para auxiliar no desenvolvimento de processos de ETL em dados estruturados, semi estruturados e não estruturados. Os participantes escolhidos foram as principais ferramentas de ETL encontradas na literatura. Os questionários utilizados para a coleta de dados são baseadas nos requisitos mínimos considerados pela literatura para ferramentas de ETL.

Segundo Travassos (2002), a experimentação é o centro do processo científico, por meio dos experimentos que é possível verificar teorias, explorar fatores críticos e formular novas teorias. O autor reforça ainda a necessidade de avaliar novas invenções e sugestões em comparação com as existentes.

Para Wohlin00, existem quatro métodos relevantes para experimentação em Engenharia de Software: científico, de engenharia, experimental e analítico.

O paradigma indutivo, ou método científico, observa o mundo, pode ser utilizado quando se quer entender o processo, produto de software, ambiente. Ele mede e analisa, verifica as hipóteses do modelo ou teoria. Já o método de engenharia observa as soluções existentes, é uma abordagem baseada na melhoria evolutiva, modifica modelos de processos ou produtos de softwares existentes com propósito de melhorar os objetos de estudo. O método experimental é uma abordagem baseada na melhoria revolucionária. Ela sugere um modelo, não necessariamente baseado em um existente, aplica o método qualitativo e/ou quantitativo, faz a experimentação, analisa e repete o processo. Por fim, o método analítico sugere uma teoria formal, é um método dedutivo que oferece uma base analítica para o desenvolvimento de modelos (Travassos, 2002).

Travassos (2002) sugere que a abordagem mais apropriada para a experimentação na área de Engenharia de Software seja o método experimental, pois considera a proposição e avaliação do modelo com os estudos experimentais.

Os principais objetivos relacionados à execução de um estudo experimental de software são: caracterização, avaliação, previsão, controle e melhoria a respeito de produtos, processos, recursos, modelos e teorias.

Os elementos principais do experimento são: as variáveis, os objetos, os participantes, o contexto do experimento, hipóteses e o tipo de projeto do experimento.

2.2 Trabalhos Correlatos

Esta seção apresenta os principais *frameworks* correlatos a este trabalho encontrados na literatura, bem como os descreve demonstrando suas características, seus pontos positivos e negativos.

2.2.1 ARKTOS II

O principal objetivo do ARKTOS II é facilitar a modelagem dos processos de ETL, de forma que o usuário define a fonte dos dados e o destino, os participantes e o fluxo de dados do processo. Como ilustrado na figura 2.2, o usuário pode desenhar atributos e parâmetros, conectá-los ao seu esquema de dados, criar relacionamentos e desenhar arestas de um nó para outro de acordo com a arquitetura do grafo (VASSILIADIS et al. (2005)).

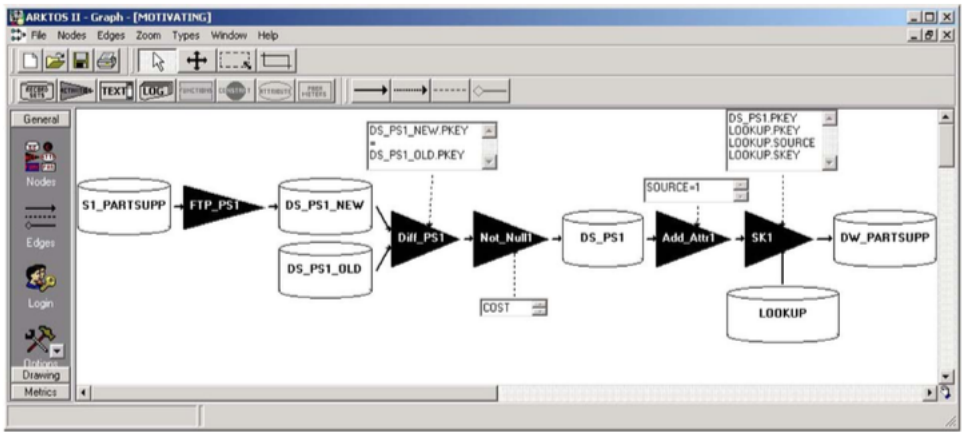

Figura 2.2 Exemplo da Ferramenta ARKTOS II em uso (Adaptado de VASSILIADIS et al. (2005))

A customização no ARKTOS II é oferecida pela reusabilidade de seus *templates*. Os processos são armazenados em um repositório implementado em um banco de dados relacional. Os autores do ARKTOS II ainda pretendem melhorar a ferramenta permitindo mais formatos de dados como XML e orientado a objetos.

2.2.2 PygramETL

PygramETL é um *framework* programável para desenvolvedores de ETL. Ele oferece a funcionalidade para desenvolver ETL demonstrando como deve-se iniciar um projeto. O propósito da ferramenta é facilitar a carga dos dados no DW gerenciado por banco de dados relacionais (RDBMS). Focando nos RDBMS como destino torna o desenvolvimento simples, não considerando outros tipos de estrutura de dados e a integração deles. Dessa forma, o PygramETL oferece suporte apenas a bancos de dados relacionais (THOMSEN; PEDERSEN (2009)).

2.2.3 ETLMR

ETLMR é um framework de ETL que utiliza *MapReduce* para atingir escalabilidade. Ele suporta esquemas de DW como o esquema estrela, o *snowflake*, e o *slowly changing dimensions* LIU; THOMSEN; PEDERSEN (2011).

A figura 2.3 ilustra o fluxo de dados usando o ETLMR usando MapReduce. O processamento da dimensão é feito em uma tarefa do MapReduce, e o processamento do fato é feito por

outra tarefa MapReduce. A tarefa MapReduce gera um número de tarefas map/reduce paralelas para processar a dimensão ou o fato. Cada tarefa consiste em inúmeros passos, incluindo a leitura dos dados no sistema de arquivos distribuído (DFS - distributed file system), execução da função de mapeamento, particionamento, combinação do mapeamento de saída, execução da função reduce e escrita dos resultados(LIU; THOMSEN; PEDERSEN (2011)).

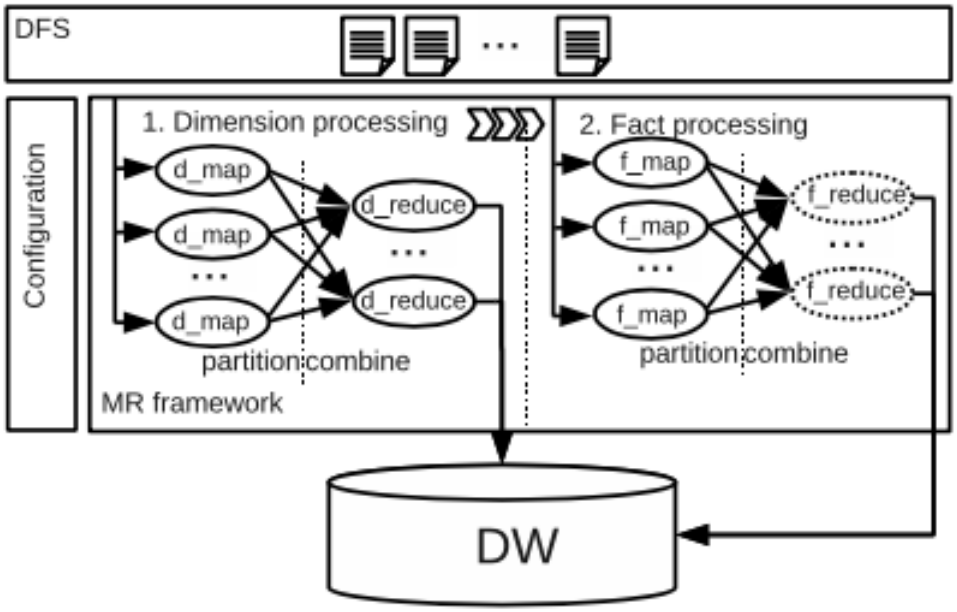

Figura 2.3 Fluxo de dados ETL no framework MapReduce (Adaptado de LIU; THOMSEN; PEDERSEN (2011))

O ETLMR possui inúmeras contribuições, ele permite construir dimensões de ETL em alto nível processando os esquemas estrela, snowflake, SCDs e dimensões de dados intensivos. Pelo fato dele utilizar MapReduce, ele pode automaticamente processar mais de um nó enquanto ao mesmo tempo fornece a sincronização dos dados através dos nós. Além da escalabilidade, ele oferece alta tolerância à falhas, possui código aberto, e é fácil de usar com um único arquivo de configuração executando todos os parâmetros.

O principal objetivo do ETLMR é otimizar o tempo de processamento dos processos de ETL por meio do framework MapReduce. Porém, não há nada a respeito de como deve ser feita a modelagem dos processos de ETL.

2.2.4 CloudETL

O framework CloudETL é uma solução para processos de ETL que usa *Hadoop* para paralelizar fluxos de ETL e *Hive* para processar os dados de forma distribuída. Para o CloudETL o *Hadoop* é a plataforma de execução dos processos de ETL e o *Hive* é o sistema de armazenamento. Conforme a figura 2.4, os componentes do CloudETL são as APIs (interfaces de programação de

aplicação), um conjunto de elementos para efetuar as transformações nos dados identificados como ETL *transformers*, e um gerenciador de tarefas que controla a execução das tarefas submetidas ao *Hadoop*.

CloudETL fornece suporte de alto nível em ETL para construção de diferentes esquemas de DW, como esquema estrela, *snowflake* e SCD (*slowly changing dimensions*). Ele facilita a implementação de processos de ETL em paralelo e aumenta a produtividade do programador significativamente. Esta abordagem facilita as atualizações de SCDs em um ambiente distribuído LIU; THOMSEN; PEDERSEN (2013).

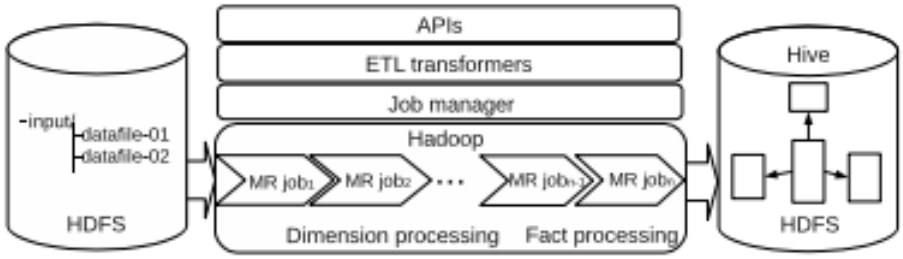

Figura 2.4 Arquitetura do CloudETL (Adaptado de LIU; THOMSEN; PEDERSEN (2013))

O CloudETL é uma alternativa quando o problema é o processamento de um grande volume de dados por possuir a propriedade de processamento distribuído, porém não oferece nenhum suporte para modelagem de processos de ETL ficando a cargo do programador ou da equipe responsável pelo projeto de DW.

2.2.5 P-ETL

P-ETL (Parallel - ETL) foi desenvolvido utilizando o *framework* Hadoop com o paradigma MapReduce. Ele oferece duas maneiras de ser configurado: por meio de uma GUI (*graphical user interface*) ou um arquivo de configuração XML. A figura 2.5 mostra a interface gráfica de configuração do P-ETL, ela é organizada em três abas: *Extract, Transform, Load* e uma parte para parâmetros avançados.

O processo de ETL do *framework* inicia-se na aba *Extract*, as configurações fornecidas pelas outras abas dependem desta primeira, principalmente o formato dos dados da fonte e sua estrutura. O primeiro passo da fase de extração é localizar a fonte de dados. O arquivo base do P-ETL é no formato "csv"(*Comma Separated Values*). Ele converte a fonte para o formato "csv"permitindo a entrada dos dados em vários formatos. Para acelerar a carga dos dados da fonte no HDFS (formato utilizado pelo *Hadoop*), o P-ETL permite o usuário comprimi-los. A respeito da partição, o usuário pode escolher o tipo de partição (*single, Round Robin, Round Robin by block*) e o número de dados por partição, além disso, ele pode configurar a extração pela quantidade de tuplas (por linhas ou blocos). A aba *Transform* permite o usuário escolher um lista de funções para transformação, cada função deve ser especificamente configurada (condições, expressões, entradas, etc.), assim, as funções são executadas na ordem que foram inseridas. E

finalmente, a aba *Loading* permite configurar as tarefas de carga e incluir o destino dos dados (*data warehouse, datamart, etc.*), os dados são comprimidos antes de serem carregados no HDFS e o separados no formato "csv"(BALA (2014)).

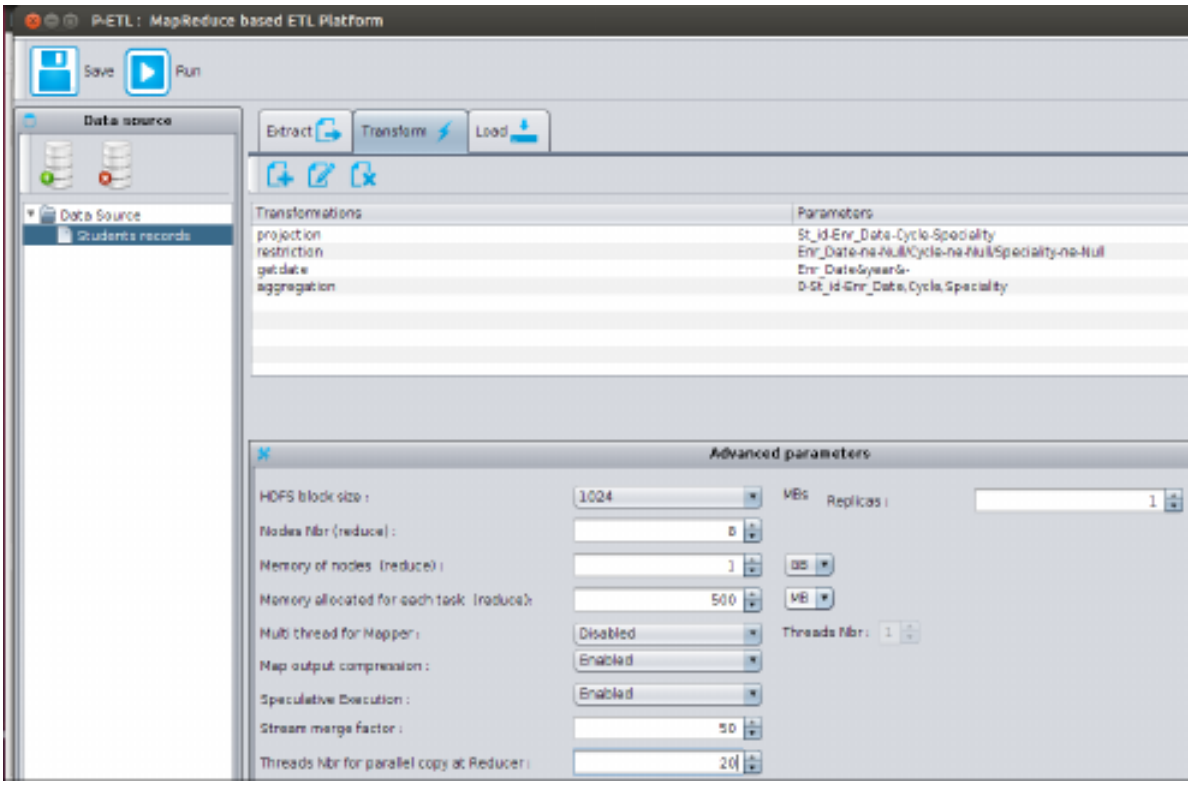

Figura 2.5 Interface de configuração do P-ETL (Adaptado de BALA (2014))

O P-ETL usa principalmente dois módulos do *framework* Apache Hadoop: (i) HDFS para o armazenamento distribuído e a alta vazão para o acesso aos dados das aplicações, e (ii) MapReduce para processar paralelamente. Futuramente, o P-ETL pretende adicionar outras funções de transformação para realizar processos mais complexos, e oferecer um ambiente na nuvem, mais precisamente, virtualizar e transformá-lo numa arquitetura orientada à serviço (SOA - *Service Oriented Architecture*) (BALA (2014)).

2.2.6 Big-ETL

BALA; BOUSSAID; ALIMAZIGHI (2015) propõe em seu trabalho uma abordagem chamada Big-ETL, no qual define que funcionalidades de ETL podem ser executadas em *cluster* utilizando o paradigma MapReduce (MR). O Big-ETL permite paralelizar e distribuir o processo de ETL em dois níveis: o processo de ETL (nível de granularidade maior), e a funcionalidade de ETL (nível de granularidade menor); dessa forma, melhorando o desempenho. Para testar a abordagem proposta, o autor utilizou o P-ETL com intuito de melhorar a ferramenta definindo o processo de ETL (nível de granularidade maior), e a funcionalidade de ETL (nível de granularidade menor) como níveis para o experimento, demonstrando ser uma boa alternativa para melhorar o desempenho nos processos de ETL.

Futuramente os autores pretendem apresentar um *benchmark* no qual comparará quatro abordagens: processo de ETL centralizado; processo de ETL distribuído, Big-ETL e uma abordagem híbrida.

2.2.7 FramETL

O FramETL é um *framework* para desenvolvimento de aplicações ETL. Ele oferece um ambiente programável e integrado para modelagem e execução de processos de ETL utilizando uma linguagem de programação. O autor utilizou conceitos de *frameworks* como flexibilidade, extensibilidade, reuso e inversão de controle para o desenvolvimento do FramETL. Por meio desses conceitos, utilizando o *framework*, o autor pode aplicar sua solução para construções de duas aplicações de ETL. Porém, SILVA (2012) não fez uso de BDs NoSQL, e o foco de sua ferramenta não era em dados *Big Data*.

2.2.8 Outras Ferramentas

Esta seção apresenta outras ferramentas de ETL presentes no mercado e na literatura, mas que não possuem um foco em BDs NoSQL e *Big Data*.

2.2.8.1 Pentaho

Pentaho Data Integration (conhecido também por ***Kettle***) é uma ferramenta *open source* para aplicações de ETL. Ela é composta basicamente por quatro elementos: extração de diferentes fontes, transporte de dados, transformação dos dados e carga no *data warehouse*. O *Kettle* pode ser implementado em um único nó, bem como na nuvem, ou em *cluster*. Ele pode carregar e processar *big data* de várias formas oferecendo flexibilidade e segurança (MALI; BOJEWAR (2015), INFORMATION (2017), INTEGRATION (2017)). Porém, por ser uma ferramenta genérica ela é de difícil customização.

2.2.8.2 Talend Studio

Talend Open Studio é uma plataforma de integração de dados que possibilita processos de integração, seu monitoramento e opera como um gerador de código, produzindo *scripts* de transformação. Ele possui um repositório de metadados no qual fornece os dados (definições e configurações relacionados a cada tarefa) para todos os seus componentes. O Talend Studio é comumente utilizado para migração de dados, sincronização ou replicação das bases de dados (MALI; BOJEWAR (2015), INFORMATION (2017)).

2.2.8.3 CloverETL

Clover é uma ferramenta ETL de código aberto considerada para transformação e integração, limpeza e distribuição de dados em aplicações, banco de dados e *data warehouses*.

Ela é baseada em Java e pode ser utilizada em linha de comando e é independente de plataforma (MALI; BOJEWAR (2015)).

2.2.8.4 Oracle Data Integrator (ODI)

Oracle Data Integrator é uma plataforma de integração de dados que atende diversos requisitos de integração, desde grandes volumes de dados até o carregamento em *batch*. As bases de dados de origem e destino podem incluir base de dados relacionais, arquivos XML, tabelas *Hive, Hbase*, arquivos *HDFS*, entre outros. Os usuários podem inserir filtros, junções, agregações, e outros componentes de transformação (SILVA (2016)). Porém, o ODI é uma ferramenta comercial e não permite a customização de suas aplicações.

2.3 Considerações Finais

Este capítulo discorreu a respeito das ferramentas de ETL encontradas na literatura. A maioria foca no desempenho ao lidar com *Big Data*, nenhuma delas apresentou uma solução alternativa para modelagem de BDs NoSQL ficando a cargo do desenvolvedor encontrar meios para esquematizar os dados. A ferramenta P-ETL (BALA (2014)) apresentou um arquivo "csv"como alternativa para obter diversos tipos de dados, porém não há um enfoque em BDs NoSQL. A abordagem PygramETL (THOMSEN; PEDERSEN (2009)) facilita a carga de dados, mas lida apenas com banco de dados relacionais. Para suportar processos de ETL em BDsNoSQL, propomos um *framework* integrado, flexível e programável de ETL para BDs NoSQL que permite o uso do processamento distribuído/paralelo e a integração das diversas bases de dados existentes por se tratar de um *framework* baseado em componentes.

3

O Framework ETL4NoSQL

Neste capítulo são apresentados os conceitos do framework ETL4NoSQL, que consiste numa plataforma de software para desenvolvimento de sistemas de ETL, mais especificamente uma ferramenta que auxilia a construção de processos de ETL buscando apoiar a modelagem e o desempenho dos processos.

O ETL4NoSQL 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 proposto foram definidas as estruturas de dados dos ambientes de origem, destino e da área de processamento de dados e suas respectivas linguagens de manipulação de dados, e também, as principais funcionalidades dos sistemas de ETL, chamados mecanismos de ETL. Para realizar os processos de ETL, por meio de seus mecanismos, foi definido um controlador de operações que é capaz de se comunicar com os ambientes e os mecanismos de ETL.

A seguir, são detalhados os requisitos de software, a arquitetura do sistema e a estrutura dos componentes utilizados no desenvolvimento do framework.

3.1 Requisitos de software do ETL4NoSQL

Requisitos de software são descrições de como o sistema deve se comportar, definidos durante as fases iniciais do desenvolvimento do sistema como uma especificação do que deveria ser implementado (SOMMERVILLE (2013)). Os requisitos podem ser divididos em funcionais e não funcionais, onde o primeiro descrevem o que o sistema deve fazer, ou seja, as transformações a serem realizadas nas entradas de um sistema, a fim de que se produzam saídas, já o outro expressa as características que este software vai apresentar.(SOMMERVILLE (2013)).

O ETL4NoSQL é um framework que tem como principal objetivo auxiliar na criação de processos de ETL ao se utilizar diversas estruturas de armazenamento de dados. Um sistema de software pode ter seus dados armazenados em bases relacionais, que seguem o modelo entidade e relacionamento, ou não relacionais, onde esta possui pouca definição de esquema, não segue um modelo específico e são regularmente chamados de NoSQL. As bases NoSQL possuem quatro paradigmas frequentemente utilizados: Chave-Valor, Família de Colunas, Documentos e Grafo.

As bases de dados relacionais utilizam uma linguagem de gerenciamento de dados padrão conhecida por SQL (Structure Query Language), porém as bases de dados NoSQL não possuem uma linguagem em comum, como as relacionais, cada estrutura de armazenamento possui sua própria linguagem de gerenciamento de dados. Por isso, é essencial que haja um mecanismo que integre a leitura e escrita dos diversos SGBDs NoSQL.

Outra importante características são os processos de ETL que possuem quatro etapas básicas: extração, limpeza/transformação e carga (KIMBALL; CASERTA (2004)). O fluxo do processo de ETL inicia-se com a extração dos dados a partir de uma fonte, que podem ser bases de dados relacionais, bases NoSQL ou arquivos textuais. A partir da extração, os dados passam para uma Área de Processamento de Dados (APD), onde é possível executar processos de limpeza e transformação por meio de mecanismos de junção, filtro, união, agregação e outros. Finalmente, os dados podem ser carregados em estrutura de dados como repositórios analíticos, data warehouses, ou até mesmo em arquivos Linguagem de Marcação Flexível (XML).

Dessa forma, o ETL4NoSQL possui um ambiente que importa os dados dos diversos SGBDs NoSQL, de arquivos textuais, além dos SGBDs relacionais, e que faz a leitura e escrita dos dados permitindo a execução dos processos de ETL. No quadro 3.1 é apresentado os principais requisitos elencados do ETL4NoSQL. Foi definido como importante as prioridades que são imprescindíveis para o desenvolvimento e funcionamento do framework, e desejável as funcionalidades que aprimoram o uso do framework, porém não interferem no seu principal objetivo.

O modelo de processo do funcionamento da ferramenta ETL4NoSQL, baseado nas notações da UML 2.0 (CLEMENTS et al. (2002)), é representado na figura 3.1. Esse modelo descreve o processamento dos dados nas atividades de identificação dos dados, obtenção das informações para a importação e o mapeamento dos dados para os esquemas desejados, e também, a atividade dos processos de ETL para por fim dar carga dos dados em DWs, repositórios

Quadro 3.1 Requisitos do ETL4NoSQL

Requisito	Prioridade
O sistema deve importar os dados de diversas bases relacionais e não relacionais	Importante
O sistema deve permitir a leitura e escrita dos dados importados	Importante
O sistema deve permitir mapear os dados no modelo relacional	Importante
O sistema deve permitir mapear os dados em quaisquer modelo desejado pelo usuário	Importante
O sistema deve possuir os mecanismos ETL mais conhecidos na literatura	Importante
O sistema deve possibilitar a criação de novos mecanismos ETL desejado pelo usuário	Importante
O sistema deve possuir um ambiente que possibilite a execução dos mecanismos de ETL em operações	Importante
O sistema deve permitir o reutilização dos seus mecanismos para vários cenários	Importante
O sistema deve permitir processamento distribuído	Desejável
O sistema deve permitir a importação de dados a partir de uma nuvem	Desejável

analíticos ou em arquivos XML.

Outro modelo importante para o entendimento do fluxo de processos da ferramenta ETL4NoSQL é o diagrama de atividades, que de acordo com a UML 2.0 tem como objetivo mostrar o fluxo de atividades em um único processo. O diagrama mostra como um atividade depende uma da outra. Na figura 3.2 o diagrama mostra a interação dos componentes ao executar um processo de ETL, onde o estágio inicial é a importação dos dados seguido pelo mapeamento, após a obtenção dos dados necessários é possível a execução dos diversos processos de ETL em uma área de processamento para finalmente os dados serem exportados para base de destino.

3.2 Arquitetura do ETL4NoSQL

SOMMERVILLE (2013), define o projeto de arquitetura como um processo criativo em que se tenta organizar o sistema de acordo com os requisitos funcionais e não funcionais. Um estilo de arquitetura é um padrão de organização de sistema (SHAW; GARLAN (1996), SOMMERVILLE (2013)), como uma organização cliente-servidor ou uma arquitetura em camadas. Porém, a arquitetura não necessariamente utilizará apenas um estilo, a maioria dos sistemas de médio e grande porte utilizam vários estilos. Para Garlan e Shaw, há três questões a serem definidas na escolha do projeto de arquitetura, a primeira é a escolha da estrutura, cliente-servidor ou em camadas, que permita atender melhor aos requisitos. A segunda questão é a respeito da decomposição dos subsistemas em módulos ou em componentes. E por fim, deve-se tomar a decisão de sobre como a execução dos subsistemas é controlada. A descrição da arquitetura pode ser representada graficamente utilizando modelos informais e notações como a

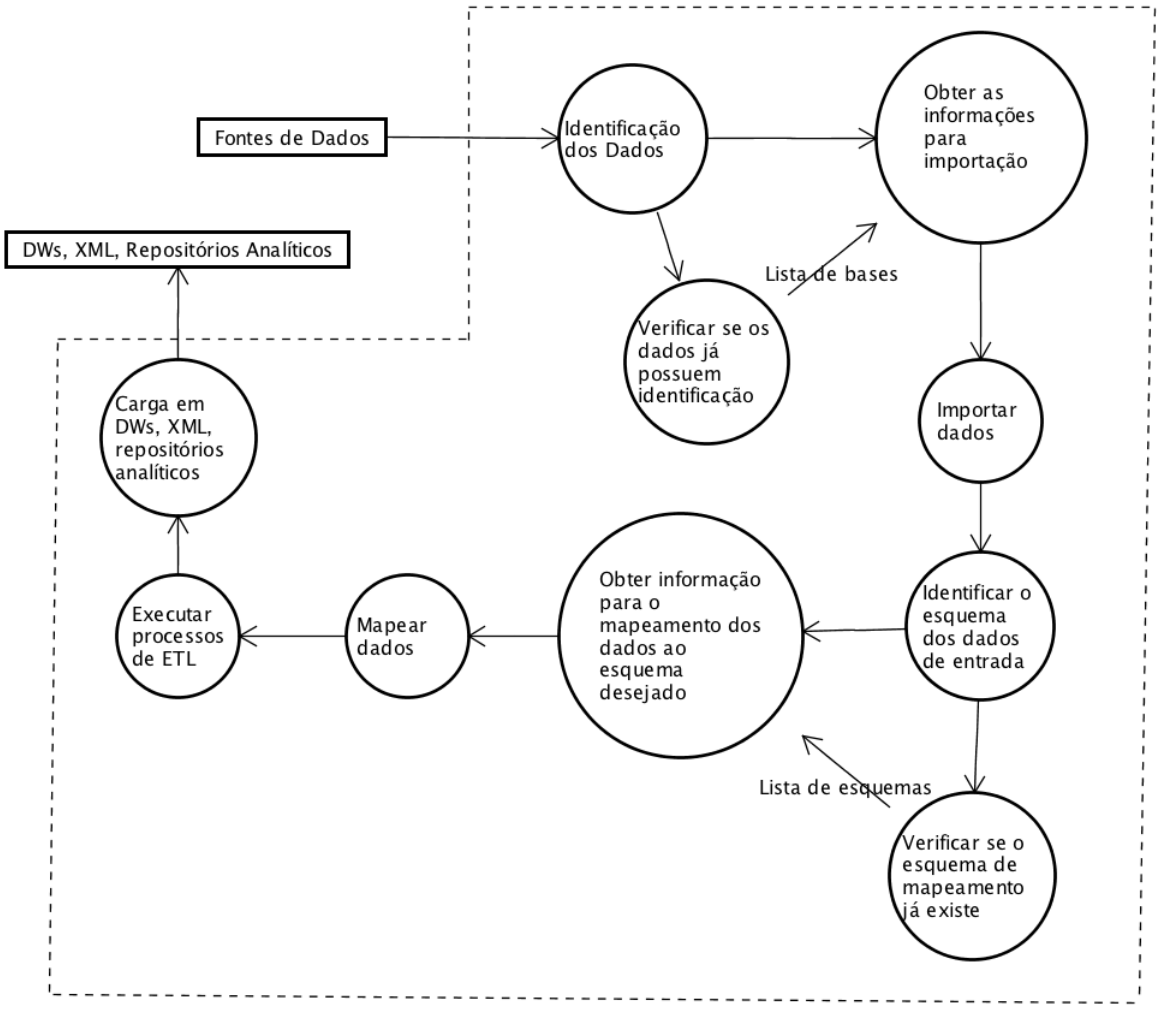

Figura 3.1 Modelo de Processos do ETL4NoSQL

UML (CLEMENTS et al. (2002), SOMMERVILLE (2013)).

A arquitetura do ETL4NoSQL, representada graficamente na figura 3.3, é baseada no requisito de reutilização. A possibilidade do reuso, reduz o trabalho repetitivo na implementação de componentes e o custo de manutenção (AOYAMA et al. (2002)), e sua estrutura é em camadas, onde há a camada de sistema e a camada de interface. A camada de sistema lida com todas as operações internas e a camada de interface faz toda a interligação do sistema com o ambiente externo. A decomposição dos subsistemas do ETL4NoSQL é em componentes, pois componentes podem ser subsistemas ou simples objetos que podem ser reusados (SOMMERVILLE (2013)). Os componentes que integram o framework e representados na figura 3.3 são os componentes de importação, mapeamento, mecanismos ETL e Operações. Estes componentes serão melhor detalhados na seção seguinte.

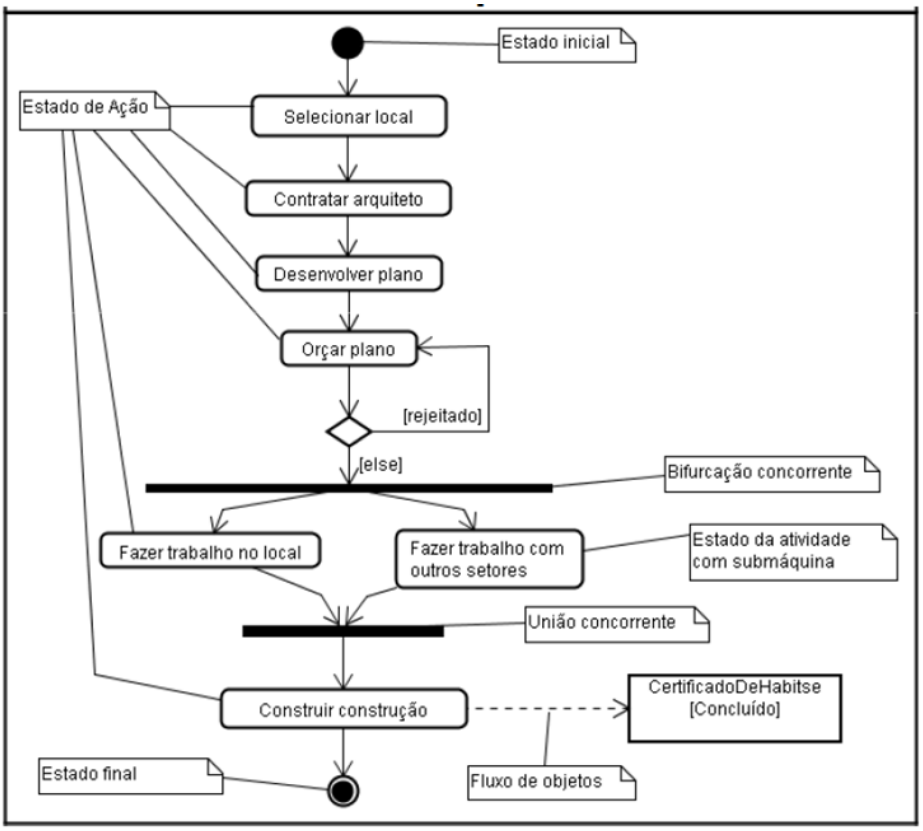

Figura 3.2 Diagrama de Atividades do ETL4NoSQL

3.3 Componentes do ETL4NoSQL

A engenharia de software baseada em componentes é uma abordagem fundamentada em reuso para desenvolvimento de sistemas de software, ela envolve o processo de definição, implementação e integração ou composição de componentes independentes não firmemente acoplados ao sistema. Os componentes são independentes, ou seja, não interferem na operação uns dos outros e se comunicam por meio de interfaces bem definidas, os detalhes de implementação são ocultados, de forma que as alterações de implementação não afetam o restante do sistema (SOMMERVILLE (2013)). Segundo SAMETINGER (1997), 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.

De acordo com os requisitos do ETL4NoSQL, foi possível identificar quatro importantes funcionalidades que podem ser definidas como componentes do sistema, a funcionalidade de importação, mapeamento de dados, mecanismos de ETL e o controlador de operações. Os componentes do ETL4NoSQL e suas características são apresentados nas seções seguintes, seguindo as características de componentes adotadas por HEINEMAN (2001).

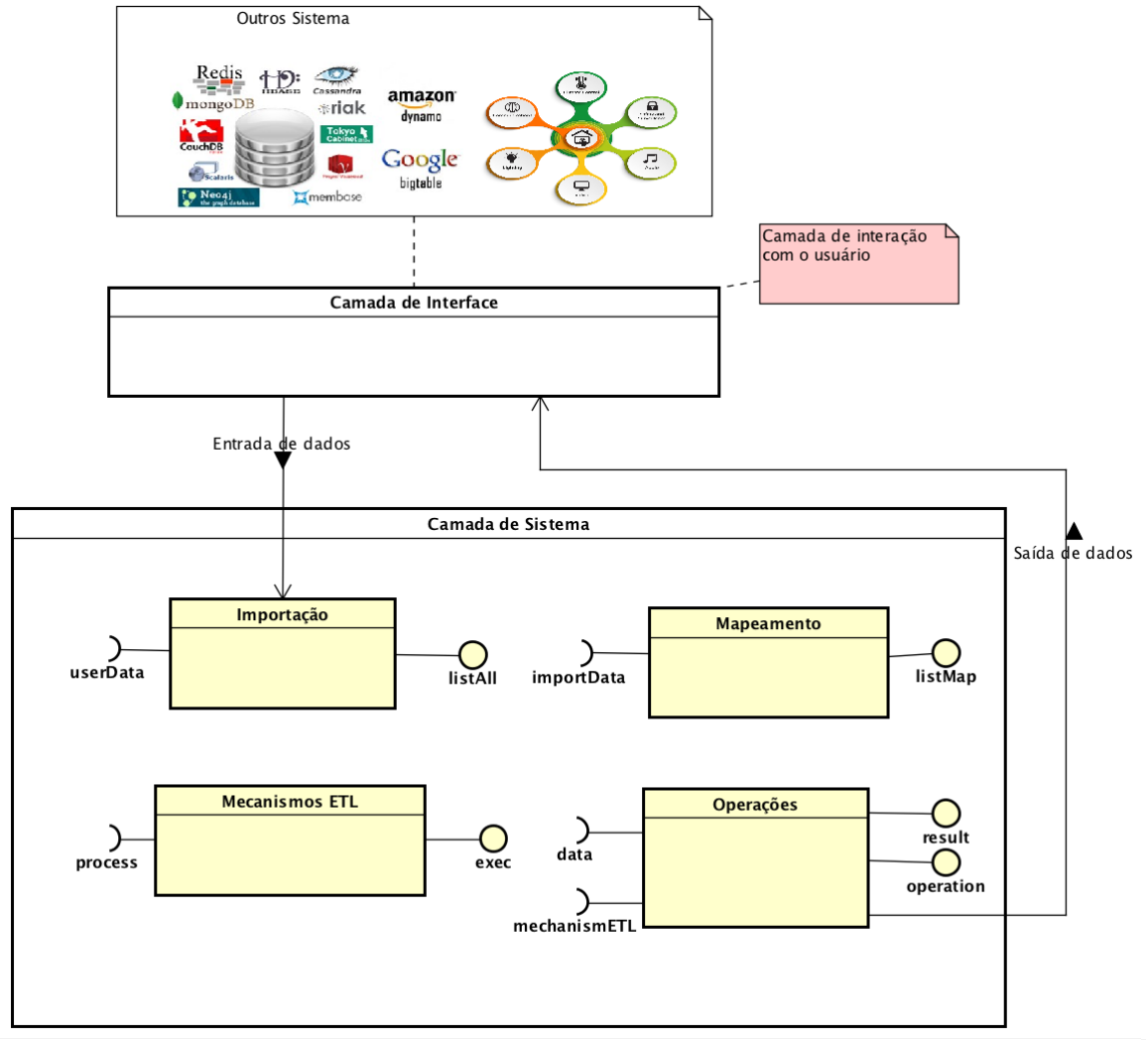

Figura 3.3 Arquitetura do Framework ETL4NoSQL

3.3.1 Componente de Importação

Um dos objetivos do framework ETL4NoSQL é possibilitar a integração de várias estruturas de dados, relacionais ou não relacionais, presentes nos sistemas modernos. Para isso, a ferramenta deve permitir a leitura e escrita dos diversos SGBDs existentes que aplicam essas estruturas. A solução encontrada para isso foi desenvolver um componente programável que possibilite a importação dos dados por meio de inserção de parâmetros em linha de comando. Este componente, por ser criado utilizando o paradigma de orientação a objetos, permite também sua extensão, por meio de especialização, para que atenda a especificidade de cada cenário. As características do componente são apresentadas a seguir.

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

b) Nomeação: Import.

c) Metadados: Este componente contém as informações da base origem como a lingua-

gem de manipulação de dados e meios para estabelecer a conexão com a base, requer uma interação com a interface para o usuário disponibilizar as informações e fornece os dados importados para outros componentes.

d) Interoperabilidade: Oferece comunicação com outros componentes por meio dos métodos listAll e userData.

e) Customização: Este componente permite customizar as formas de apresentar os dados importados, de acordo com a necessidade de cada sistema.

f) Suporte a evolução: Possibilita o suporte aos métodos de acordo com as mudanças de conexões e manipulações de bases de dados futuras.

g) Empacotamento e utilização: Os métodos são encapsulados e podem ser utilizados pela importação de sua classe e a interface com o usuário é por meio de linha de comando.

3.3.2 Componente de Mapeamento

Para viabilizar a organização dos dados em vários tipos de esquemas desejáveis pelo usuário o ETL4NoSQL oferece o componente de mapeamento. Este componente permite definir o esquema dos dados de acordo com a necessidade da aplicação almejada pelo usuário. Por meio de parâmetros de inserção em linha de comando é possível utilizar os esquemas de dados pré-definidos pelo componente, mas também, por utilizar o paradigma de orientação a objetos e as características de reusabilidade dos componentes, é possível especializar e customizar os esquemas conforme a conveniência do usuário.

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

b) Nomeação: Map.

c) Metadados: Este componente requer os dados de uma base de dados para efetuar o mapeamento.

d) Interoperabilidade: Oferece comunicação com outros componentes por meio dos métodos importData e listMap.

e) Customização: É possível customizar as regras de mapeamento para outros esquemas de dados.

f) Suporte a evolução: Possibilita o suporte aos métodos de acordo com a necessidade de alterar os esquemas dos dados.

g) Empacotamento e utilização: Os métodos são encapsulados e podem ser utilizados pela importação de sua classe e a interface com o usuário é por meio de linha de comando.

3.3.3 Componente de Mecanismos de ETL

O ETL4NoSQL é um framework de ETL que possibilita a integração de várias estruturas de dados, por isso ele deve apresentar mecanismos que viabilizem as principais operações de ETL conhecidas pela literatura. Dessa forma, para disponibilizar as operações de ETL, o ETL4NoSQL possui um componente de mecanismos de ETL que permite executar processos de ETL como extração, limpeza/transformação e carga de dados. Além das operações básicas de ETL, o componente permite a especialização e criação de mecanismos permitindo a customização das operações de ETL conforme a necessidade do usuário.

a) Interface: Componente que contém métodos que realizam as principais operações de ETL presentes na literatura.

b) Nomeação: MechanismETL.

c) Metadados: Este componente requer dados de controle para realizar as operações por meio de seus métodos.

d) Interoperabilidade: Oferece comunicação com outros componentes por meio dos métodos exec e process.

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

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

g) 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.

3.3.4 Componente de Operações

Para proporcionar o controle dos processos de ETL executados pelo framework, o ETL4NoSQL possui o componente de operações. Este componente é responsável pelo controle das operações dos processos de ETL, ele assegura a execução dos mecanismos de ETL de acordo com a necessidade do usuário. É possível também, customizar e especializar as operações deste componente.

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 mecanismos de ETL e deverá criar e executar processos de ETL.

d) Interoperabilidade: Deve possibilitar a comunicação entre outros componentes.

e) Customização: É possível customizar os processos de ETL criados.

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

g) 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.

3.4 Considerações Finais

4

Estudo Experimental de Software

Este capítulo provê o roteiro de experimentação de software para ferramentas de ETL utilizando dados estruturados, semi estruturados e não estruturados. A Engenharia de Software Experimental tem como objetivo aprimorar métodos, técnicas e ferramentas de Engenharia de Software a partir de métodos experimentais (Isaque Elcio de Souza, TESE - Um sistema de Inf para Geren de Projetos Experimentais em ES). As etapas definidas no processo de experimentação em Engenharia de Software proposto por [Amaral (),Isaque Elcio de Souza, TESE] consiste em etapas de definição, planejamento, operação, interpretação dos dados e empacotamento que serão melhor detalhados nas seções a seguir.

4.1 Objetivos do experimento

O objetivo principal da aplicação deste experimento é definir se o framework proposto por esta pesquisa de dissertação é uma ferramenta adequada para auxiliar no desenvolvimento de processos de ETL em dados estruturados, semi estruturados e não estruturados.

4.1.1 Objetivo da Medição

Tendo como base as ferramentas de ETL existentes na literatura, caracterizar:

1. Quais as principais funcionalidades que as ferramentas de ETL oferecem:
    (a) essas funcionalidades manipulam dados estruturados, semi estruturados e não estruturados.
    (b) essas funcionalidades não manipulam dados estruturados, semi estruturados e não estruturados.
2. Quais funcionalidades podem ser consideradas fundamentais para a produtividade na criação de processos de ETL:
    (a) quais necessitam manipular dados em grande escala.
    (b) quais não manipulam grande volume de dados.
3. Quais funcionalidades poderiam aprimorar as ferramentas de ETL.

4.1.2 Objetivos do Estudo

- Analisar as ferramentas de ETL para dados estruturados, semi estruturados e não estruturados;
- Com o propósito de caracterizar;
- Com respeito à intersecção das ferramentas de ETL existente;
- Do ponto de vista da literatura;
- No contexto de comparativo entre as ferramentas mais conhecidas no mercado atual.

4.1.3 Questões

Q1. Existem funcionalidades listadas pelas ferramentas pesquisadas que não estão presentes no ETL4NoSQL?

Métrica: A lista de funcionalidades que não estão presentes no ETL4NoSQL.

Q2. Existem funcionalidades oferecidas pelo ETL4NoSQL que não estão presentes nas ferramentas apresentadas pela literatura?

Métrica: A lista de funcionalidades que não estão presentes nas ferramentas da literatura.

Q3. Existem funcionalidades que não estão presentes no ETL4NoSQL e nas ferramentas da literatura que poderiam ser implementadas?

Métrica: A lista de funcionalidades que não estão presentes em nenhuma das ferramentas.

4.2 Planejamento

Na etapa de planejamento são definidas as hipóteses do estudo, a descrição da instrumentação, as métricas, seleção do contexto e dos indivíduos, as variáveis, a análise qualitativa e a validade do experimento. Todas elas serão descritas nas seções seguintes.

4.2.1 Definição das Hipóteses

Hipótese nula (H0): As funcionalidades oferecidas pelo ETL4NoSQL são similares às funcionalidades oferecidas pelas ferramentas presentes na literatura.

Fp - Funcionalidades do ETL4NoSQL

Fl - Funcionalidades das ferramentas da literatura

H0: Fl - (Fp $\cap$ Fl) = $\emptyset$

Hipótese alternativa (H1): A lista de funcionalidades oferecidas pelo ETL4NoSQL é diferente da lista de funcionalidades oferecidas pelas ferramentas presentes na literatura.

Fp - Funcionalidades do ETL4NoSQL

Fl - Funcionalidades das ferramentas da literatura

H1: Fl - (Fp $\cap$ Fl) $\neq \emptyset$

Hipótese alternativa (H2): A lista de funcionalidades que poderiam ser implementadas é diferente da lista de funcionalidades oferecidas pelas ferramentas na literatura e pelo ETL4NoSQL.

Fp - Funcionalidades do ETL4NoSQL

Fl - Funcionalidades das ferramentas da literatura

Fi - Funcionalidades que poderiam ser implementadas

H2: Fi - (Fp $\cap$ Fl $\cap$ Fi) $\neq \emptyset$

4.2.2 Descrição da instrumentação

Para cada funcionalidade presente nas ferramentas apresentada na literatura que são consideradas fundamentais para o funcionamento dos processos de ETL pode ser encontrada no quadro 4.1:

Para cada funcionalidade aplicar teste estatístico qui-quadrado para definir:

se pode considerar que essa funcionalidade é fornecida;

Quadro 4.1 Descrição da Instrumentação

Presença da Funcionalidade (P)	Melhoria da Funcionalidade (M)	Utilidade da Funcionalidade (U)
1. Não está presente 2. Está presente parcialmente 3. Está presente	1. Necessita melhorar 2. Não há necessidade de melhoria 3. Pode melhorar, mas não necessidade	1. É útil 2. Não é útil 3. É parcialmente útil

se pode considerar que essa funcionalidade é útil;

se pode considerar que essa funcionalidade necessita de melhoria.

Resultado: N funcionalidades com valores (P; M; U) onde P - presença 0 - não presente; 1 - presente; U - utilidade 0 - não é útil; 1 - é útil; melhoria 0 - não necessita melhorar; 1 - necessita melhorar.

4.2.3 Métricas

Na tabela 4.2 são apresentadas as métricas utilizadas neste experimento.

Quadro 4.2 Métricas

No	P	M	U	Descrição da Funcionalidade	Questões
1	0	0	0	Não está presente, não necessita melhorar, não é útil	N/A
2	0	0	1	Não está presente, não necessita melhorar, é útil	Q3
3	0	1	0	Não está presente, necessita melhorar, não é útil	N/A
4	0	1	1	Não está presente, necessita melhorar, é útil	Q3
5	1	0	0	Está presente, não necessita melhorar, não é útil	Q1, Q2
6	1	0	1	Está presente, não necessita melhorar, é útil	Q1, Q2
7	1	1	0	Está presente, necessita melhorar, não é útil	Q1, Q2
8	1	1	1	Está presente, necessita melhorar, é útil	Q1, Q2

4.2.4 Seleção do contexto

De acordo com Travassos (2002), o contexto pode ser caracterizado conforme quatro dimensões:

- o processo: on-line / off-line;
- os participantes: ferramentas de ETL;
- realidade: o problema real / modelado;

- generalidade: específico / geral.

Nosso estudo supõe o processo off-line porque as ferramentas não estão sendo testadas durante todo o tempo da utilização, mas em certo instante. Os participantes são as ferramentas de ETL encontradas na literatura. O estudo é modelado porque as funcionalidades das ferramentas não são caracterizadas durante a resolução do problema real, mas utilizando parâmetros subjetivos (ex. presença, utilidade e necessidade). As funcionalidades do ETL4NoSQL são comparadas com as ferramentas presentes na literatura, então, o contexto possui o caráter específico.

4.2.5 Seleção dos indivíduos

Como participantes para o estudo propõe-se utilizar as ferramentas encontradas na literatura. Assume-se que esses indivíduos estão presente em diversos estudos realizados e avaliados no meio acadêmico.

Para a escolha das ferramentas utilizadas neste estudo foi levado em consideração a semelhança da finalidade do uso com a ferramenta proposta. Seria conveniente utilizar para o estudo ferramentas que tem o objetivo de auxiliar processos de ETL em diversas estruturas de dados. Dessa forma, a seleção baseou-se nas características das ferramentas.

4.2.6 Variáveis

Variável independente: A lista de funcionalidades das ferramentas encontradas na literatura.

Variáveis dependentes:

1. A similaridade entre as funcionalidades oferecidas pela ferramenta proposta e as funcionalidades encontradas nas ferramentas da literatura.

   Pode receber os valores: Igual, quando todas as funcionalidades tem o valor PMU = { 1, X, X } (métricas 5-8); Diferente, quando todas as funcionalidades tem o valor PMU = { 0, X, X } (métricas 1-4) Similar, quando não se cumprem as condições de "Igual"e "Diferente". O grau de similaridade pode ser avaliado como: { 1, X, X } / { 0, X, X } + { 1, X, X } * 100%

2. A utilidade das funcionalidades similares. Mostra a parte útil das funcionalidades oferecidas pela ferramenta proposta: Parte útil: { 1, X, 1 } / { 1, X, X } * 100% Parte inútil: { 1, X, 0 } / { 1, X, X } * 100%

3. A melhoria das funcionalidades similares. Mostra a necessidade de melhoria nas funcionalidades oferecidas pela ferramenta proposta: Não necessita melhorar: { 1, 0, X } / { 1, X, X } * 100% Necessita melhorar: { 1, 0, X } / { 1, X, X } * 100%

4.2.7 Análise Qualitativa

Para analisar a informação referente às funcionalidades não oferecidas no ETL4NoSQL, mas que poderiam ser implementadas, propõe-se aplicar a análise qualitativa. Essa análise deve apresentar a lista de funcionalidades presentes nas ferramentas da literatura, que não estão presentes na ferramenta proposta, mas que são consideradas necessárias para facilitar a manipulação de dados estruturados, semi estruturados e não estruturados. Assim, essa análise deve considerar funcionalidades com valor PMU = 0, X, X (métricas 1-4) e a opção ”É útil” para ”utilidade da funcionalidade”.

4.2.8 Validade

Validade interna: como mencionado na parte "Seleção dos indivíduos"para o estudo se propõe a utilizar ferramentas presentes na literatura, que são validadas pelo meio acadêmico. Assim, assume-se que elas são representativas para a população de ferramentas de ETL.

Além disso, para redução da influência dos fatores que não são interesse do nosso estudo e, portanto, para aumento da validade interna do estudo supõe-se utilizar dados das ferramentas mais populares da literatura, cuja a validação já tenha passado por diversas avaliações.

Validade de conclusão: para receber os valores da presença, utilidade e melhorias o teste binomial será utilizado. A verificação de hipótese será feita por meio de simples demonstração de presença ou não de funcionalidades nas listas que representam as variáveis independentes.

Validade de construção: esse estudo está caracterizado pela conformidade das funcionalidades listadas na ferramenta proposta com as funcionalidades reais necessárias para a utilização de ferramentas de ETL. As características das ferramentas de ETL presentes na literatura representa a lista de funcionalidades que uma ferramenta de ETL deve apresentar para mostrar o desempenho adequado do ponto de vista da literatura. As funcionalidades, que tem o maior relacionamento com as ferramentas de ETL do ponto de vista dos pesquisadores, foram escolhidas do conjunto total de funcionalidades das ferramentas de ETL presentes na literatura.

Validade externa: como foi mencionado nas partes "Seleção dos indivíduos"e "Validade interna"os participantes do estudo em geral podem ser considerados representativos para a população da literatura apresentada pela academia. Para avaliação do nível de importância das funcionalidades analisadas foi levada em consideração a frequência que a funcionalidade apareceu nas ferramentas da literatura.

Os materiais utilizados no estudo podem ser considerados representativos e "em tempo"para o problema sob análise, porque se compõem das funcionalidades de ferramentas de ETL presentes na literatura atual.

4.3 Operação

A etapa de operação ocorre após a etapa de planejamento do estudo experimental. Nela é exercido o monitoramento do experimento para garantir que ele esteja ocorrendo conforme foi planejado (Souza Isaque, 2015). Nesta seção serão apresentados os questionários do perfil da ferramenta de ETL e o de Funcionalidades.

4.3.1 Questionário do Perfil da Ferramenta de ETL

O quadro 4.3 mostra as questões usadas para definir o perfil das ferramentas utilizadas como indivíduos deste experimento.

Quadro 4.3 Questionário do Perfil da Ferramenta de ETL

Nome da ferramenta de ETL:	
Possui código aberto?	Sim ◯ Não ◯
Possui uma marca reconhecida no mercado?	Sim ◯ Não ◯
Tem como finalidade utilizar bancos de dados NoSQL?	Sim ◯ Não ◯
Possui interface gráfica?	Sim ◯ Não ◯
É programável?	Sim ◯ Não ◯
É integrada?	Sim ◯ Não ◯
Qual o tipo de processamento que a ferramenta executa?	Distribuído ◯ Centralizada ◯ Híbrido ◯
É extensível?	Sim ◯ Não ◯
Para qual finalidade a ferramenta procura auxiliar melhor os processos de ETL?	Modelagem ◯ Desempenho ◯ Ambos ◯

4.3.2 Questionário de Funcionalidades

Sob o ponto de vista das características das ferramentas e considerando a finalidade da ferramenta indicada acima, avalie as colunas correspondentes segundo as escalas abaixo, a presença, utilidade e melhorias quanto às funcionalidades das ferramentas apresentadas nos seus respectivos trabalhos de pesquisa, das funcionalidades listadas no questionário:

Quadro 4.4 Instrumentação para aplicar o questionário

Presença da Funcionalidade (P)	Melhoria da Funcionalidade (M)	Utilidade da Funcionalidade (U)
1. Não está presente 2. Está presente parcialmente 3. Está presente	1. Necessita melhorar 2. Não há necessidade de melhoria 3. Pode melhorar, mas não necessidade	1. É útil 2. Não é útil 3. É parcialmente útil

N	Funcionalidade	Descrição	Presença			Utilidade			Melhoria		
			1	2	3	1	2	3	1	2	3
	Processo										
1	Suporte à plataforma	Ser independente de plataforma.									
2	Suporte à fonte	Ser capaz de ler diretamente da fonte de dados, independente do seu tipo, saber se é uma fonte RDBMS, arquivo de texto, XML ou NoSQL.									
3	Suporte ao destino	Ser capaz de carregar diretamente os dado no destino, independente do seu tipo, saber se o destino é RDBMS, arquivo de texto, XML ou NoSQL.									
4	Suporte à modelagem	Apoiar na extração de dados de múltiplas fontes, na limpeza dos dados, e na transformação, agregação, reorganização e operações de carga.									
5	Paralelismo	Apoiar as operações de vários segmentos e execução em paralelo, internamente. A ferramenta deve ser capaz de distribuir tarefas entre múltiplos servidores.									
6	Apoio ao nível de debugging	Apoiar o tempo de execução e a limpeza da lógica de transformação. O usuário deve ser capaz de ver os dados antes e depois da transformação.									
7	Programável	Apoiar o agendamento de tarefas de ETL e ter suporte para programação em linha de comandos usando programação externa.									
8	Implementação	Suportar a capacidade de agrupar os objetos ETL e implementá-los em ambiente de teste ou de produção, sem a intervenção de um administrador de ETL.									
9	Reutilização	Apoiar a reutilização dos componentes do framework e da lógica das transformações para evitar a reescrita.									
10	Garantia da qualidade	Ser capaz de estabelecer processos, métricas e avaliações que possibilitem e garantam a qualidade de software.									

Figura 4.1 Questionário de Funcionalidades

4.3.3 Resultado do Estudo

A figura 4.2 apresenta o gráfico da quantidade de presença para cada funcionalidade de acordo com cada ferramenta de ETL. Já a figura 4.3 mostra o nível de importância que as ferramentas dão para cada funcionalidade e a figura 4.4 indica necessidade de melhorar funcionalidade em cada ferramenta.

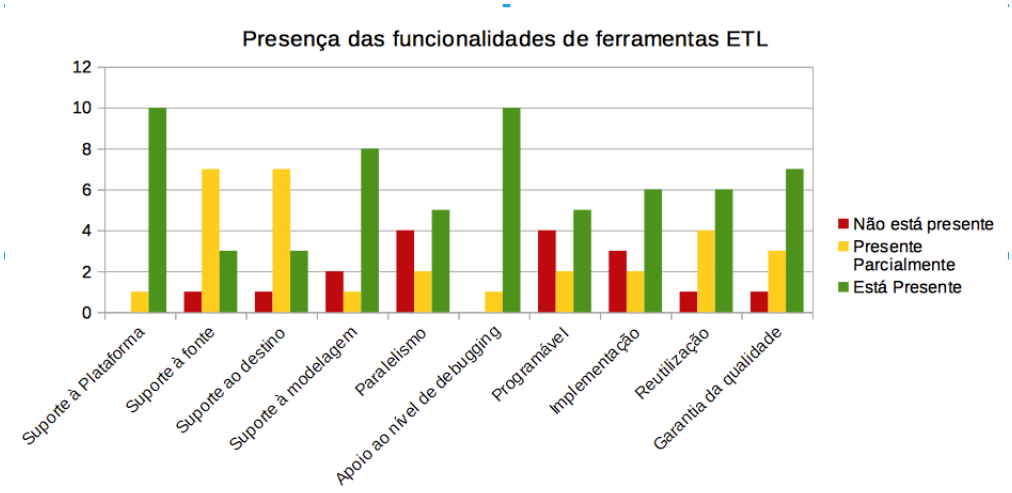

Figura 4.2 Quantidade de Presença para cada funcionalidade

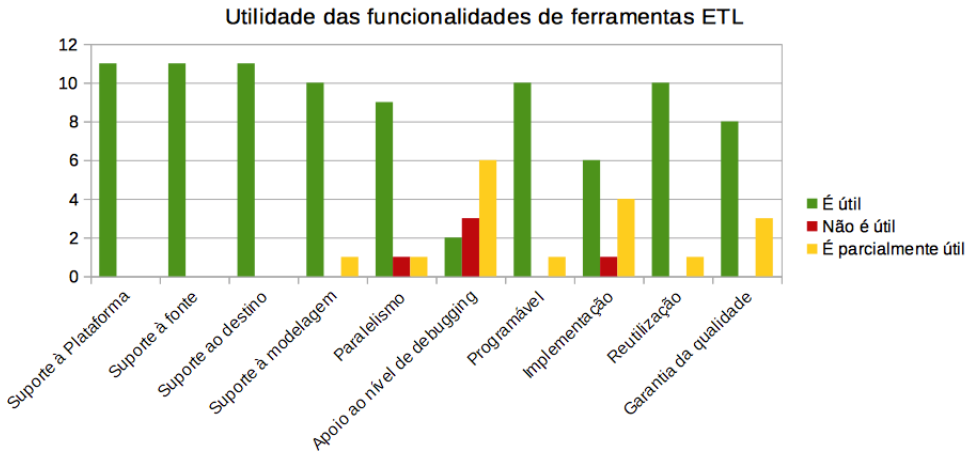

Figura 4.3 Níveis de utilidade para cada funcionalidade

O perfil dos participantes é apresentado no quadro 4.5, bem como a legenda para leitura é exibida no quadro 4.6. Na figura 4.5 é possível comparar por meio do gráfico os perfis das ferramentas analisadas por este estudo experimental.

Das onze ferramentas de ETL apresentadas neste estudo dez possuem código aberto, sete não possui uma marca reconhecida no mercado, nenhuma foi desenvolvida para suportar BDs NoSQL, seis possuem uma GUI, apenas quatro não são programáveis, duas não são integradas, nove delas oferece alguma alternativa para o processamento distribuído, um pouco mais da metade são extensíveis e a finalidade da maioria é modelagem tentando aliar ao desempenho.

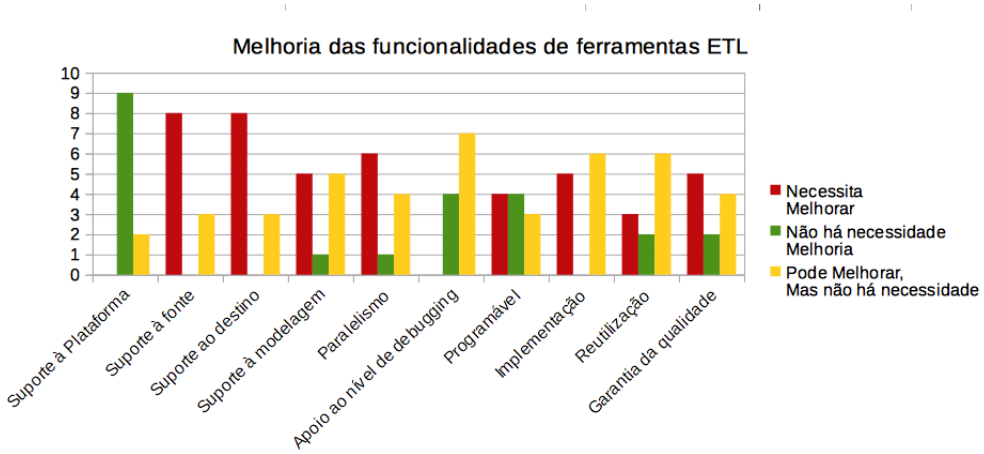

Figura 4.4 Necessidade de melhoria para cada funcionalidade

Quadro 4.5 Resultado do Perfil dos participantes

Ferramenta	Código Fonte	Marca de mercado	Para NoSQL	GUI	Programável	Integrada	Processamento	Extensível	Finalidade
PygramETL	1	2	2	2	1	1	2	1	2
ARKTOSII	1	2	2	1	2	1	3	2	1
Big-ETL	1	2	3	2	1	2	1	1	3
ETLMR	1	2	2	2	1	1	1	1	2
CloudETL	1	2	3	2	1	1	1	1	2
P-ETL	1	2	3	1	2	1	3	2	3
FramETL	1	2	2	2	1	1	2	1	1
Talend Studio	1	1	3	1	1	1	3	2	3
Pentaho Kettle	1	1	3	1	2	1	3	2	1
CloverETL	1	1	3	1	1	2	3	2	3
Oracle (ODI)	2	1	3	1	2	1	3	2	3

Quadro 4.6 Legenda

Código Fonte	Marca de mercado	Para NoSQL	GUI	Programável	Integrada	Processamento	Extensível	Finalidade
1-Aberto	1-Reconhecido	1-Sim	1-Possui	1-Sim	1-Sim	1-Distribuído	1-Sim	1-Modelagem
2-Fechado	2-Não reconhecido	2-Não	2-Não possui	2-Não	2-Não	2-Centralizado	2-Não	2-Desempenho
		3-Possui, mas não é o foco				3-Híbrido		3-Ambos

4.4 Análise e interpretação dos resultados

Nesta seção é apresentada a análise e interpretação dos resultados conforme as regras estatísticas escolhidas para serem aplicadas neste estudo.

4.4.1 Estatística descritiva

As medidas de tendência central como os valores "Presença", "Melhoria"e "Utilidade"são da escala ordinal, por isso é possível definir as métricas de "moda"e "mediana". As métricas de presença estão no quadro 4.7, as métricas de utilidade são apresentadas no quadro 4.9 e as métricas de melhoria podem ser vistas no quadro 4.8.

Considerando as respostas recebidas durante o estudo e utilizando os resultados de estatística descritiva podemos dividir as perguntas em 4 grupos. O primeiro grupo são as funcionalidades que estão presente, mas são parcialmente úteis; já o segundo grupo são as funcionalidades presentes e úteis; o terceiro grupo consiste na presença da funcionalidade com

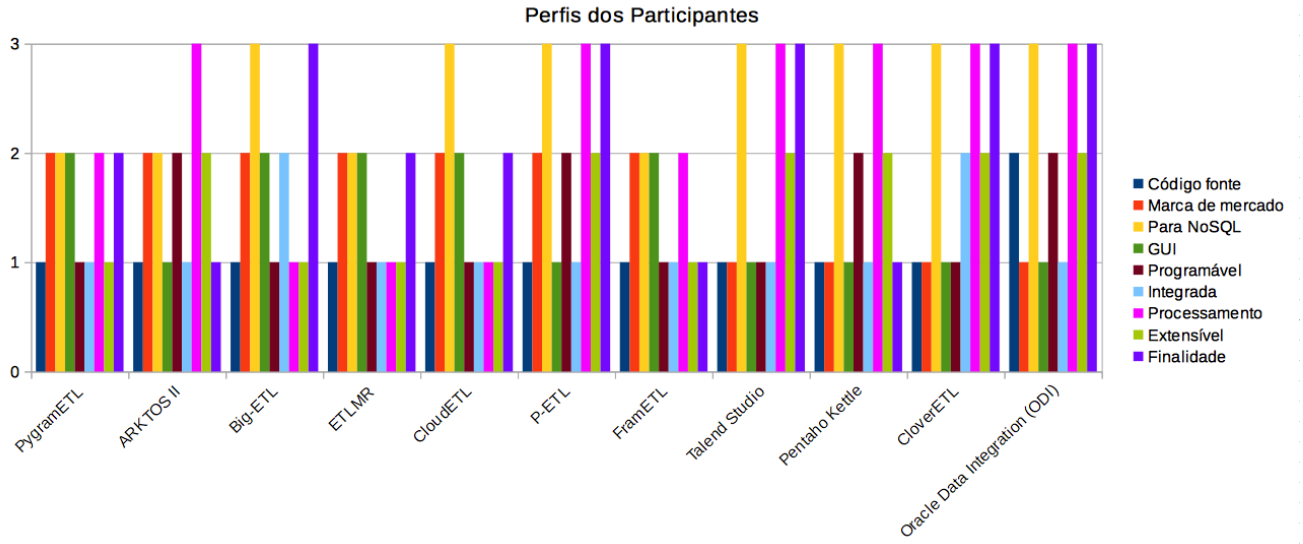

Figura 4.5 Perfil das Ferramentas de ETL

Quadro 4.7 Estatística Descritiva da Presença de Funcionalidades

Presença										
Funcionalidades	1	2	3	4	5	6	7	8	9	10
Mediana	3	2	2	3	2	3	2	3	3	3
Moda	3	2	2	3	3	3	3	3	3	3

necessidade de melhoria; e finalmente, o quarto grupo que representa as funcionalidades que estão presentes e não necessitam de melhoria. Todos os grupos podem ser analisados nos quadros 4.10, 4.11, 4.12 e 4.13.

Os valores na tabela significam P - presente:não presente; M - necessita melhoria:não necessita melhoria; U - útil;inútil.

4.4.2 Aplicação do teste estatístico

Para cada funcionalidade foi aplicado o teste de associação Qui-quadrado,com o objetivo de verificar se existe associação entre a ferramenta e a presença da sua funcionalidade. O teste do qui-quadrado pode ser usado em pesquisas de amostras independentes com a variável de resposta qualitativa (categórica) (BARBETTA (2012)). O quadro 4.14 mostra os resultados da quantidade de funcionalidades presentes nas ferramentas do estudo.

Dessa forma, calculamos o valor esperado para cada funcionalidade através da fórmula de frequências esperadas:

E = (total da linha) x (total da coluna) / (total geral)
E = 63 x 11 / 110 = 6,3
E = 47 x 11 / 110 = 4,7

Quadro 4.8 Estatística Descritiva da Melhoria de Funcionalidades

Melhoria										
Funcionalidades	1	2	3	4	5	6	7	8	9	10
Mediana	2	1	1	2	1	3	2	3	3	2
Moda	2	1	1	{1,3}	1	3	{1,2}	3	3	1

Quadro 4.9 Estatística Descritiva da Utilidade de Funcionalidades

Utilidade										
Funcionalidades	1	2	3	4	5	6	7	8	9	10
Mediana	1	1	1	1	1	3	1	1	1	1
Moda	1	1	1	1	1	3	1	1	1	1

Quadro 4.10 Funcionalidade presente e parcialmente útil

N.	Funcionalidade	P	M	U	Característica
6	Apoio ao nível de debugging	10:1	0:11	8:3	- A funcionalidade existe na maioria das ferramentas, mas não dão o enfoque nela. - É uma funcionalidade trivial, pois é possível comparar os dados da fonte e destino mesmo sem debugging. - Pode ser útil para evitar o trabalho de fazer a comparação da fonte com o destino, além de poder dar uma resposta em tempo de execução.

Quadro 4.11 Funcionalidade presente e é útil

N	Funcionalidade	P	M	U	Características
2	Suporte à fonte	10:1	8:3	11:0	- As ferramentas oferecem algum suporte em relação à leitura e carga da fonte e destino, porém muitas apenas lidam com RDBMS. - Deve haver um enfoque nas novas soluções de BDs como os BDs NoSQL. - Algumas ferramentas não oferecem a opção para utilizar linha de comando, o qual é considerado importante para ter maior liberdade na implementação de processos de ETL. -Dependendo do foco da ferramenta, alguma delas não dão importância ao processamento em paralelo, porém em se tratando de Big Data, o processamento em paralelo é fundamental para possibilitar a execução de processos de ETL em grandes volumes de dados.
3	Suporte ao destino	10:1	8:3	11:0	
5	Paralelismo	7:4	6:5	10:1	
7	Programável	7:4	4:7	11:0	

Para a frequência das funcionalidades existentes o valor esperado é E = 6,3 e para frequência não existentes o valor esperado é E = 4,7.

A fórmula estatística para o teste qui-quadrado é definida por:

Quadro 4.12 Funcionalidade presente e necessita melhorar

N	Funcionalidade	P	M	U	Características
4	Suporte à modelagem	9:2	5:6	11:0	- A maioria das ferramentas dão suporte à modelagem dos processos de ETL, porém algumas focam apenas no desempenho necessitando melhoria no processo de modelagem. - Poucas ferramentas dão importância para garantir qualidade aos processos de ETL ou ao seu desempenho.
10	Garantia de qualidade	10:1	5:6	11:0	

Quadro 4.13 Funcionalidade presente e não necessita melhoria

N	Funcionalidade	P	M	U	Características
1	Suporte à plataforma	11:0	2:9	11:0	- As ferramentas demonstraram ser independentes de plataforma por terem características como código compilável e serem programáveis. - Criar processos de ETL demonstrou ser complexo, necessitando de algum tipo de experiência na área. Dessa forma, não é de fundamental importância que haja suporte à implementação para usuários não administradores. - A maioria das ferramentas, por serem de código aberto e programáveis, oferecem a possibilidade de reutilização de código ou algumas até mesmo oferecem possibilidade de reusar os processos de ETL já criados.
8	Implementação	8:3	5:6	10:1	
9	Reutilização	10:1	3:8	11:0	

$\chi^2 = \sum \frac{(O-E)\hat{}2}{E}$

onde: a soma se estende a todas as células da tabela de contingência; O representa a frequência observada na célula; e E representa a frequência esperada na célula.

Somando os valores das parcelas do χ^2, temos:

$\chi^2 = 20{,}765$

com

$gl = (l - 1) \,.\, (c - 1) = (2 - 1)\,(10 - 1) = 9$

Quadro 4.14 Quadro de resultado do valor observado de existência da funcionalidade

Funcionalidade	1	2	3	4	5	6	7	8	9	10	Total
Existe	10	3	3	8	5	10	5	6	6	7	63
Não existe	1	8	8	3	6	1	6	5	5	4	47
Total	11	11	11	11	11	11	11	11	11	11	110

Quadro 4.15 Quadro do resultado de χ^2

Funcionalidade	1	2	3	4	5	6	7	8	9	10
Existe	2,17	1,72	1,72	0,45	0,26	2,17	0,26	0,014	0,014	0,077
Não existe	2,91	2,31	2,31	0,61	0,36	2,91	0,36	0,02	0,02	0,1

Pela tabela de distribuição qui-quadrado, verificamos que a probabilidade de significância ρ é inferior a 0,01. Então, para qualquer nível usual de significância ($\alpha = 0,05$), o teste detecta associação entre as funcionalidades e as ferramentas (pois, $\rho < \alpha$). Em outras palavras, o teste qui-quadrado mostrou que as ferramentas em estudo são diferentes quanto às suas funcionalidades.

4.4.2.1 Análise quantitativa

Para verificar a similaridade entre as funcionalidades existentes nas ferramentas de ETL na literatura e as funcionalidades do ETL4NoSQL é necessário calcular o valor de variável dependente.

1. Os conjuntos de funcionalidades existentes nas ferramentas de ETL da literatura e as funcionalidades do ETL4NoSQL não podem ser consideradas iguais (a quantidade de funcionalidades com valor PMU 1, X, X ?? < 10), nem diferentes (a quantidade de funcionalidas com valor PMU 0, X, X < 10). Assim, precisamos calcular o grau de similaridade segundo a fórmula do variável dependente 1:

grau de similaridade = ?? / 10 * 100

2. Para verificar a utilidade das funcionalidades similares, é necessário calcular o valor da variável dependente 2:

parte útil das funcionalidades similares = ??? / ?? * 100

parte inútil das funcionalidades similares = ??? / ?? * 100

3. Para verificar a necessidade de melhoria das funcionalidades similares é necessário calcular o valor de variável dependente 3:

parte que necessita de melhoria das funcionalidades similares =

parte que não necessita de melhoria das funcionalidades similares =

4.4.3 Análise qualitativa

4.4.4 Verificação das hipóteses

4.5 Considerações Finais

5

Conclusão

5.1 Principais Contribuições

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea

dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

5.2 Discussão

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

5.3 Resultados

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat

a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

5.4 Trabalhos Futuros

Referências

AOYAMA, M. et al. Web Services Engineering: promises and challenges. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, 24., New York, NY, USA. **Proceedings...** ACM, 2002. p.647–648. (ICSE '02).

BALA, M. P-ETL: parallel-etl based on the mapreduce paradigm. **IEEE**, [S.l.], Nov. 2014.

BALA, M.; BOUSSAID, O.; ALIMAZIGHI, Z. Big-ETL: extracting-transforming-loading approach for big data. **Int'l Conf. Par. and Dist. Proc. Tech. and Appl.**, [S.l.], 2015.

BARBETTA, P. A. **Estatística aplicada às Ciências Sociais**. [S.l.]: Editora da UFSC, 2012.

CARVALHO SCABORA, L. de. **Avaliação do Star Schema Benchmark aplicado a bancos de dados NoSQL distribuídos e orientados a colunas**. 2016. Dissertação (Mestrado em Ciência da Computação) — USP - São Carlos.

CHAUDHURI, S.; DAYAL, U. An overview of data warehousing and OLAP technology. **SIGMODRecord,v.26,n.1,p.65–74**, [S.l.], 1997.

CHEVALIER, M. et al. Implementing Multidimensional Data Warehouses into NoSQL. **Proceedings of the 17th International Conference on Enterprise Information Systems, p.172-183, April 27-30**, [S.l.], 2015.

CLEMENTS, P. et al. **Documenting Software Architectures**: views and beyond. [S.l.]: Pearson Education, 2002.

DARMONT, J. et al. An architecture framework for complex data warehouses. **7th International Conference on Enterprise Information Systems (ICEIS'05), Miami, USA, pages 370–373**, [S.l.], 2005.

E., F. M.; C., S. D. Object-Oriented Application Frameworks. **Communica- tions of the ACM 40 (10): 32–38**, [S.l.], 1997.

PRATES, R. (Ed.). **NoSQL Essencial Um guia conciso para o mundo emergente de Persistência Poliglota**. [S.l.]: Novatec, 2013. Primeira Edição.

HEINEMAN, G. T. **Component-Based Software Engineering**: putting pieces together. [S.l.]: Addison-Wesley, 2001.

INFORMATION, E. T. URL: http://etl-tools.info/en/pentaho/kettle-etl.htm.

INTEGRATION, P. D. URL: http://www.pentaho.com/product/data-integration.

KAUR, K. Modeling and querying data in NoSQL databases. **Big Data, 2013 IEEE International Conference on**, [S.l.], 2013.

ELLIOT, R. (Ed.). **The Data Warehouse ETL ToolKit**. [S.l.]: Robert Ipsen, 2004.

ELLIOT, R. (Ed.). **The Data Warehouse Toolkit**. [S.l.]: Robert Ipsen, 2002. Second Edition.

LIU, X.; THOMSEN, C.; PEDERSEN, T. B. ETLMR: a highly scalable dimensional etl framework based on mapreduce. **DB Tech Reports**, [S.l.], Aug. 2011.

LIU, X.; THOMSEN, C.; PEDERSEN, T. B. CloudETL: scalable dimensional etl for hive. **DB Tech Reports**, [S.l.], July 2013.

MALI, N.; BOJEWAR, S. A Survey of ETL Tools. **International Journal of Computer Techniques**, [S.l.], Oct. 2015.

NASHOLM, P. **Extracting Data From NoSQL Databases**. 2012. Tese (Doutorado em Ciência da Computação) — Chalmers University of Technology, SE-412 96 Goteborg Sweeden.

PREE, W.; SIKORA, H. **Design Patterns for Object Oriented Software Developiment.** [S.l.: s.n.], 1997.

WILEY, J.; SONS (Ed.). **Business Intelligence Success Factors**: tools for aligning your business in the global economy. EUA: Inc, 2009.

SAMETINGER, J. **Software Engineering with Reusable Componets**. [S.l.]: Springer, 1997.

SCHIMIDT, D. C.; GOKHALE, A.; NATARAJAN, B. Leveraging Application Frameworks. **ACM Queue, v. 2**, [S.l.], 2004.

SHAW, M.; GARLAN, D. **Software Architecture**: perspectives on an emerging discipline. [S.l.]: Prentice Hall, 1996. Prentice Hall Ordering Information.

SILVA, L. M. M. **ETL na era do Big Data**. 2016. Dissertação (Mestrado em Ciência da Computação) — Técnico Lisboa.

SILVA, M. S. da. **Um Framework para Desenvolvimento de Sistemas ETL**. 2012. Dissertação (Mestrado em Ciência da Computação) — Universidade Federal de Pernambuco.

SOMMERVILLE, I. **Engenharia de Software**. [S.l.]: Pearson, 2013.

TALIGENT, I. **Building Object-Oriented Frameworks.** [S.l.: s.n.], 1994.

THOMSEN, C.; PEDERSEN, T. B. pygrametl: a powerful programming framework for extract-transform-load programmers. **DB Tech Reports**, [S.l.], Nov. 2009.

VASSILIADIS, P. et al. A generic and customizable frameworkfor the design of ETL scenarios. **Information Systems - Special issue: The 15th international conference on advanced information systems engineering 30 (7): 492–525**, [S.l.], 2005.

Apêndice