申请上海交通大学硕士专业学位论文

**面向交通物流大数据处理的**

**元数据管理系统**

姓 名：葛晖

学 号：1120379069

班 级：B1203793

专 业：软件工程

导 师：饶若楠

答辩日期：2015年1月14日

上 海 交 通 大 学 软 件 学 院

2014年12月

**A Dissertation Submitted to Shanghai Jiao Tong University for Master Degree**

**METADATA MANAGEMENT SYSTEM FOR TRANSPORT AND LOGISTICS BIG DATA PROCESS**

**Author:** Hui Ge

**Advisor:** Ruonan Rao

**Specialty:** Software Engineering

School of Software

Shanghai Jiao Tong University

Dec，2014

上海交通大学

学位论文原创性声明

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

学位论文作者签名：

日期： 年 月 日

上海交通大学

学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定，同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权上海交通大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

保密□，在 年解密后适用本授权书。

本学位论文属于

不保密☑。

（请在以上方框内打“√”）

学位论文作者签名： 指导教师签名：

日期： 年 月 日 日期： 年 月 日

面向交通物流大数据处理的元数据管理系统

# 摘 要

在交通物流领域中，建立统一、高效的数据管理方式是降低运营成本、提高数据质量、规范物流管理的迫切需要。本文以某省交通物流云平台项目为背景，该项目包含一系列交通物流大数据处理需求，如对物流企业、应用开发商及平台数据中心的数据进行数据整合并提供统一管理。目前，在数据整合及管理的过程中，主要存在以下问题：1）业务元数据和技术元数据没有建立映射关系，无法对企业异构数据源进行统一管理；2）没有将业务元数据与技术元数据结合，指导数据整合过程；3）由于涉及到安全及隐私问题，数据整合时对企业私有数据访问困难。

针对上述问题，本论文在对元数据标准SDMX信息模型、RDF数据立方词汇表以及数据整合等相关技术进行深入研究的基础上，提出了一个面向交通物流领域的业务元数据模型，建立了技术元数据与业务元数据的映射，并与ETL工具相结合，设计并实现一个面向交通物流大数据处理的元数据管理系统。测试及应用表明，该系统是可行及有效的。

与其他类似系统相比，本文工作具有以下特点：

1）结合SDMX信息模型及RDF数据立方词汇表，提出了一个面向交通物流领域的业务元数据模型，对交通物流领域的术语、业务实体等进行建模；同时建立并维护业务元数据与技术元数据的映射关系。

2）提出了基于元数据的交通物流数据整合方法，将业务元数据应用到交通物流领域的数据整合。在用户开发数据整合任务时，利用业务元数据与技术元数据的映射，自动建立整合字段源和目标的匹配，简化了开发过程。

3）设计并实现了元数据管理系统，该系统包括元数据管理工具与数据整合工具两部分。元数据管理工具提供技术元数据获取、元数据管理、元数据映射、前置机管理及系统管理功能；数据整合工具提供数据整合任务开发与数据整合任务执行功能。应用情况表明，该系统增强了数据管理能力并提高了数据整合效率。

**关键词：**元数据管理、数据整合、SDMX信息模型

**METADATA MANAGEMENT SYSTEM FOR TRANSPORT AND LOGISTICS BIG DATA PROCESS**

# ABSTRACT

In the field of transport and logistics, establishing a unified, efficient data management way is an urgent need of reducing operating costs, improving data quality and standardizing logistics management. The paper is on the background of transport and logistics cloud platform project of a province, which contains a series of transport and logistics big data process requirements like integrating data from logistics enterprises, application developers and data center as well as managing them. So far, in data integration and management, some problems exist: 1) no mapping is established between business metadata and technical metadata thus no unified management is taken among heterogeneous enterprise data sources; 2) business metadata and technical metadata are not combined to guide data integration; 3) as it relates to safety and privacy issues, the access to enterprise private data sources in data integration is difficult.

To solve these problems, we first study deeply in metadata standard SDMX information model, RDF data cube vocabulary, data integration and other related technologies; and propose a business metadata model for the transport and logistics field; then we establish the mapping between business metadata and technical metadata; and with the combination of an ETL tool, we design and implement a metadata management system for transport and logistics data process. Test and application indicates that our system is feasible and effective.

Compared with other similar systems, our work has following features:

1) The paper combines SDMX information model and RDF data cube vocabulary, proposes a business metadata model for the transport and logistics field and build models for terminology and business entities in the field of transport and logistics as well as establishes and maintains the mapping between business metadata and technical metadata.

2) The paper proposes a transport and logistics data integration approach based on metadata by using business metadata in data integration. When users develop data integration tasks, the mapping between business metadata and technical metadata can help to automatically create the matching between source and target integrating fields, which simplified the development process.

3) The paper designs and implements a metadata management system, including a metadata management tool and a data integration tool. The metadata management tool provides technical metadata harvesting, metadata management, metadata mapping, agent management and system management functions while the data integration tool provides data integration tasks developing and data integration executing functions. Application result shows that the system enhances the data management capabilities and improves the data integration efficiency.

**KEY WORDS:** Metadata Management, Data Integration, SDMX Information Model

**目 录**

[摘 要 I](#_Toc407728184)

[ABSTRACT II](#_Toc407728185)

[目 录 IV](#_Toc407728186)

[第一章 绪论 1](#_Toc407728187)

[1.1 研究背景及意义 1](#_Toc407728188)

[1.2 国内外研究现状 2](#_Toc407728189)

[1.2.1 数据整合的研究 2](#_Toc407728190)

[1.2.2 元数据管理的研究 3](#_Toc407728191)

[1.3 研究目标及内容 4](#_Toc407728192)

[1.4 本文章节安排 5](#_Toc407728193)

[第二章 交通物流业务元数据模型与数据整合方法 6](#_Toc407728194)

[2.1 元数据标准、分类及管理模式 6](#_Toc407728196)

[2.1.1 元数据标准 6](#_Toc407728197)

[2.1.2 企业元数据及ETL元数据 7](#_Toc407728198)

[2.1.3 元数据管理模式 8](#_Toc407728199)

[2.2 SDMX信息模型 10](#_Toc407728200)

[2.2.1 信息模型元素 10](#_Toc407728201)

[2.2.2 SDMX-ML 11](#_Toc407728202)

[2.3 RDF数据立方词汇表 12](#_Toc407728203)

[2.3.1 数据立方体 12](#_Toc407728204)

[2.3.2 词汇表元素 14](#_Toc407728205)

[2.4 交通物流业务元数据模型 15](#_Toc407728206)

[2.4.1 交通和物流领域业务元数据分类 15](#_Toc407728207)

[2.4.2 基于SDMX信息模型扩展的交通物流业务元数据模型 16](#_Toc407728208)

[2.4.3 交易订单实例 18](#_Toc407728209)

[2.5 业务元数据与技术元数据的映射 23](#_Toc407728210)

[2.5.1 交通物流技术元数据 23](#_Toc407728211)

[2.5.2 元数据映射模型 24](#_Toc407728212)

[2.6 基于元数据的交通物流数据整合方法 27](#_Toc407728213)

[2.6.1 数据整合流程 27](#_Toc407728214)

[2.6.2 ETL相关技术分析 29](#_Toc407728215)

[2.7 本章小结 34](#_Toc407728216)

[第三章 系统需求分析与架构设计 35](#_Toc407728217)

[3.1 业务分析 35](#_Toc407728219)

[3.2 系统需求分析 37](#_Toc407728220)

[3.2.1 功能性需求 37](#_Toc407728221)

[3.2.2 非功能性需求 40](#_Toc407728222)

[3.3 系统架构设计 40](#_Toc407728223)

[3.3.1 技术架构 40](#_Toc407728224)

[3.3.2 逻辑架构 41](#_Toc407728225)

[3.3.3 部署视图 43](#_Toc407728226)

[3.3.4 系统实现环境 44](#_Toc407728227)

[3.4 本章小结 45](#_Toc407728228)

[第四章 系统核心功能模块的设计与实现 46](#_Toc407728229)

[4.1 技术元数据获取 46](#_Toc407728231)

[4.1.1 技术元数据获取方法 46](#_Toc407728232)

[4.1.2 技术元数据获取模块 48](#_Toc407728233)

[4.2 业务元数据管理模块 51](#_Toc407728234)

[4.3 元数据映射模块 52](#_Toc407728235)

[4.4 字段匹配模块 53](#_Toc407728236)

[4.5 ETL前置机管理模块 57](#_Toc407728237)

[4.6 本章小结 60](#_Toc407728238)

[第五章 系统测试及应用 61](#_Toc407728239)

[5.1 系统测试 61](#_Toc407728241)

[5.1.1 功能测试 61](#_Toc407728242)

[5.1.2 性能测试 62](#_Toc407728243)

[5.2 系统应用 64](#_Toc407728244)

[5.2.1 运行实例 64](#_Toc407728245)

[5.2.2 应用效果及分析 70](#_Toc407728246)

[5.3 本章小结 72](#_Toc407728247)

[第六章 总结与展望 73](#_Toc407728248)

[6.1 工作总结 73](#_Toc407728250)

[6.2 下一步工作 73](#_Toc407728251)

[参 考 文 献 74](#_Toc407728252)

[致 谢 77](#_Toc407728253)

[攻读硕士学位期间已发表的学术论文 78](#_Toc407728254)

# 绪论

## 研究背景及意义

交通物流行业在当前社会的进步与发展中所处的地位正在不断升高，同时其运营成本也在社会经济中产生了巨大影响。面对信息化产生的海量级别的存储数据，建立统一、高效的数据管理方式是降低运营成本、提高数据质量、规范物流管理的迫切需要。我国物流企业的发展尚未完善，企业对各自数据的管理标准不一致，在数据处理环节中缺乏统一的数据管理手段。因此，在交通物流领域统一数据管理是一个亟待解决的问题。

本文以某省交通厅的交通物流云平台为背景，该平台建设了统一的数据中心，将大量的业务数据，包括日常运营、订单跟踪、综合执法、运政服务信息等数据整理、归类，形成符合平台业务和商业需要的主题数据库，同时为交通物流行业中物流企业、政府部门、行业协会等提供增值服务。如物流企业可以通过交易信息主题数据库中历史货源信息、车源的统计分析情况，决定是否对物流生产加大投入，这样，开发商就可以通过调用交易信息主题数据库的服务，进行相关数据分析功能开发，提供给物流企业使用。行业协会可以通过货运市场诚信主题数据库、交易信息主题数据库等对交通物流行业的运行情况进行分析，得出行业的运行情况，开发商可以对货运市场诚信主题数据库、交易信息主题数据库等进行调用开发出一些满足行业协会统计的报表、行业健康情况分析等功能。主题数据库从平台外部的各家物流企业系统（如综合执法系统、道路运政信息管理系统等）及平台内的系统（如车辆定位系统、订单跟踪系统、运营管理系统等）中整合包括日常运营、订单跟踪、交易信息、运政服务信息等数据，并按照不同需求对这些数据加以分析和处理，最后以数据服务的方式提供给所有平台用户。然而，在构建数据中心以及从不同的物流企业整合数据的过程中，遇到了以下难题：

目前，在从各物流企业向数据中心进行的数据整合过程中，为了使各企业的业务人员能够理解来自其他企业的数据含义，交通厅指定了一套数据库建设规范，通过文档描述的方式规定了字段命名、类型、长度等基本元数据，并在主题数据库的建立中投入使用。然而，对于改造物流企业的私有数据库结构的改造开销非常大，并且这种方法无法适用于未来的不可预测的变动，一旦物流行业的某些建设规范改变，则需要重新改造已有的数据库。因此，如何建立合适的、符合规范的并且可扩展的数据管理方案，成为了一大问题。

## 国内外研究现状

### 数据整合的研究

数据整合通过集成位于不同数据源的数据，为用户提供一个统一的视图[1]。数据整合过程中通常涉及到ETL（Extract-Transform-Load）[2]技术，ETL用来描述数据从来源经过抽取、转换并装载到目标数据仓库的过程。在数据整合的研究中，主要有对传统ETL技术的优化与改进，以下列举一些研究成果。

Oracle提出了E-LT体系结构，其充分利用了数据库的能力和吞吐量，而仅仅受限于其部署服务器的特性，保证了优异的性能和可扩展性[3]。

另外，从ETL的出发点考虑，数据需要首先通过抽取进入临时数据区，在这个过程中，每次抽取的数据量决定了这一过程的处理时间。在实际生产环境中，大部分的ETL工具都提供了数据的全量抽取和增量抽取两种方式，从一定程度上避免了重复数据抽取造成的不必要的开销。相应地，在完成一次数据抽取后，需要定期同步源和目标数据，保证数据的一致性[4]。

文献[5]认为在数据整合的过程中，理解、分析并使用这些数据通常是非常困难的。过多的数据会引起巨大的整合开销，低质量的数据甚至会降低数据质量。因此，合理选择数据源成为一个不可忽视的问题。该文献随后提出了利用经济学理论中的边际主义，计算整合任务的单位开销，并在整合开始前优先选取收益与成本差最大的数据源子集，当边际收益小于边际成本时，停止选取新的数据源。该文献通过这种方法来平衡整合数据的质量和整合开销。

文献[6]指出，如今的数据量正在急速增长，而这些数据通常是由对应的元数据加以描述的，对于文件而言，元数据通常指文件名、内容概要、创建时间、作者等信息。目前的工具在应对海量数据时，表现出的可扩展性欠佳，同在对于初始的数据装载过程中，消耗的时间过于冗长。针对这一问题，该文献提出了Lazy ETL的方式，在初始数据装载时，仅装载数据量较小的元数据来保证良好的可扩展性。当用户发出查询请求时，才进行实际数据的抽取、转换和装载。这种方式减少了构建新数据资料库时的开销，被认为是在“接近实时ETL领域[7]”做出的重要一步。

文献[8]认为，真实世界中的数据都是“脏的”：世界顶尖企业中的关键数据有超过25%是有缺陷的。针对这一问题，文献[9]提出了一种通用的数据清洗系统，该系统允许从ETL工具中导入清洗规则或用户自定义规则，通过数据监测和清洗核心模块对规则进行编译，识别数据中存在的问题，并进行自动修复。

文献[10]提出了对企业中大量ETL过程的管理，并实现了对如映射，合并等ETL过程的管理组件。文献[11]提出了一种维度式ETL框架ETLMR，该框架基于MapReduce保证其可扩展性。文献[12]认为如今的ETL工具必须具有足够的灵活性，已支持传统的迭代型以及增量型的数据整合，同时需要对海量数据源保持其可扩展性，因此该文献提出了通过注解的方式在查询时执行映射规则的方法。文献[13]从数据量、数据变化速度、数据种类、数据质量四个方面分析了海量数据的特征，并总结了在ETL在海量数据中遇到的困难和挑战。

### 元数据管理的研究

元数据作为描述数据的数据[14]，在企业中可被分为业务元数据和技术元数据两个分支[15]，它们之间相互关联、相互参照，具体、全面地描述了数据仓库中的数据含义，同时能使这些数据得到正确、及时的检验与维护。目前，不同机构针对各自领域对元数据定义了多种标准，通过遵循标准，大大降低了数据收集、处理以及交换过程中不必要的时间开销，主要有公共仓库元模型CWM[38]，统计数据及元数据交换SDMX[18]，面向语义Web的本体语言OWL[32]等。同时针对上述标准，许多商业化工具提供了对其的支持，采集并同步不同来源的元数据，便于企业对庞大的业务数据进行系统的、有效的管理，如IBM公司的InfoSphere Metadata Workbench，Informatica公司的PowerCenter Metadata Manager，Pentaho公司的Metadata Editor等。

另外，国内外研究者在元数据管理工具方面提出了多种不同的设计与实现。文献[21]提出了一种分布式的元数据管理，解决了负载和带宽导致的元数据获取效率问题，提升了可扩展性。文献[22]提出了一种数据-元数据的数据转换体系，应用于数据交换中。文献[17]基于CWM标准，采用了页面元数据自动抽取技术，并在电子政务元数据的指导下，构建了元数据管理系统。研究[19][20]基于开源软件提出了一套联邦的元数据采集、搜索和获取工具Mercury，该工具针对存储规范的元数据采用直接将元数据从外部数据库采集到Mercury服务器中并响应用户请求的方式；同时，对于分散的，非规范化存储的元数据，Mercury对用户提供这些元数据的获取URL，由用户直接联系元数据服务器获取。通过这两种不同架构的并存，Mercury为用户提供了快速、高效的数据查询。

## 研究目标及内容

本论文以实验室承担的某省交通厅交通物流云平台项目为背景，针对平台及物流企业对多源异构数据统一管理及整合的需求，结合SDMX信息模型及RDF数据立方词汇表，提出了一个面向交通物流领域的业务元数据模型，建立了技术元数据与业务元数据的映射，并与ETL工具相结合，设计并实现一个面向交通物流大数据处理的元数据管理系统，并通过系统测试及试运行应用，以验证该系统的可行性及有效性。

围绕上述研究目标，本文的研究内容主要有：

1）业务及需求分析：

a）交通物流云平台及平台物流企业、应用开发商需要对多源异构数据进行统一管理与整合；

b）对数据的管理方式需要有效支持大数据处理任务；

c）在外部物流企业到平台数据中心的数据整合过程中，需要改进ETL工具的任务设计步骤，实现数据转换时，源字段与目标字段匹配的自动建立；

2）相关技术分析研究：分析并研究元数据、SDMX信息模型、RDF数据立方词汇表及ETL等相关技术；

3）系统的设计与实现：设计并实现元数据管理系统，该系统可有效支持交通物流数据处理任务，主要有以下特点：

a）结合SDMX信息模型及RDF数据立方词汇表，提出一个面向交通物流领域的业务元数据模型，并设计一体化交通物流信息元数据库，建立交通物流业务元数据与技术元数据的映射；

b）设计并实现元数据管理功能，包括技术元数据获取模块、元数据管理模块、元数据映射模块、数据整合任务管理模块、前置机管理模块以及系统管理模块；

c）提供元数据管理访问接口，包括技术元数据获取接口、元数据管理接口、元数据映射接口、数据整合任务管理接口及系统管理接口等；

d）设计并实现Web方式的元数据管理界面，提供业务元数据管理、技术元数据获取、技术元数据管理、元数据映射、数据整合任务管理及前置机管理等功能；

e）改进已有的ETL工具，提供对元数据的支持，设计并实现数据整合任务开发时源字段与目标字段的自动匹配，为任务执行时的数据转换提供业务元数据指导。

4）系统测试及验证：通过系统测试及试运行应用，验证该系统的可行性及有效性。

## 本文章节安排

本文一共分为6个章节，各章节内容安排如下：

第一章，绪论。本章阐述了本文的研究背景及意义，综述了国内外研究现状，明确了本文的研究目标及内容。

第二章，交通物流业务元数据模型与数据整合方法。本章首先对建立交通物流业务元数据模型中涉及到的技术进行分析与研究。之后结合SDMX信息模型及RDF数据立方词汇表，提出了一个面向交通物流领域的业务元数据模型，然后建立了技术元数据与业务元数据的映射。最后，提出了基于元数据的交通物流数据整合方法。

第三章，系统需求分析及架构设计。本章首先对交通物流元数据管理系统进行了需求分析，包括业务分析、功能性需求分析以及非功能性需求分析。然后从系统的技术架构、逻辑架构、部署视图以及系统实现环境等四个方面进行了系统架构设计。

第四章，系统核心功能模块的设计与实现。本章重点介绍了元数据管理系统核心功能模块的设计与实现，包括技术元数据获取模块、业务元数据管理模块、元数据映射模块、字段匹配模块及ETL前置机管理模块。

第五章，系统测试及应用。本章首先对元数据管理系统进行了功能测试和性能测试，然后设计系统运行实例，并对系统应用效果进行分析。

第六章，总结与展望。本章对本文内容进行总结，并对后续工作提出展望。

# 交通物流业务元数据模型与数据整合方法



## 元数据标准、分类及管理模式

### 元数据标准

元数据是描述数据的数据，主要描述数据属性的信息[14]。目前，不同机构针对各自领域对元数据定义了多种标准，通过遵循标准，大大降低了数据收集、处理以及交换过程中不必要的时间开销，具有代表性的元数据标准如下：

* 公共仓库元模型CWM

CWM规范是由OMG（对象管理组织）在2000年发布的。它主要基于UML统一建模语言、MOF元对象工具以及XMI元数据交换这三个工业标准，提供标准的语言描述元数据语法和语义，其目的是标准化数据仓库和商业智能领域元数据的定义和交换机制，以描述一个完整的数据仓库系统的所有组成部分[38]，它具有以下特点：

1）规范了数据仓库的元数据定义和格式，并提供了标准的交换方式；

2）面向数据仓库平台，数据仓库服务提供商、管理者及用户。

* SDMX

SDMX全称为统计数据及元数据交换（Statistical Data and Metadata Exchange），是由国际清算银行（BIS）、欧洲中央银行（ECB）、欧洲统计局、国际货币基金组织、经济合作与发展组织（OECD）、联合国（UN）、世界银行在2001年共同发起的用于统计信息交换的一套标准[18]，它主要具有以下优点：

1）它提供了描述数据并结构化统计内容的通用模型；

2）它是一个国际标准；

3）它拥有标准化软件工具的支持，并被权威机构所广泛使用；

* RDF数据立方词汇表（RDF Data Cube Vocabulary）

资源描述框架RDF是一种用于描述Web资源的标记语言，其使用XML语法来将元数据描述为数据模型[29]。在许多情况下，数据呈现多维度状态，如统计数据等。发布这样的数据，需要将其关联到相关的数据集以及元数据。数据立方词汇表利用RDF标准，支持多维度数据的发布，并且它与SDMX提供的数据模型相兼容[30]。

RDF数据立方词汇表具有以下特点：

1）RDF数据立方词汇表以SDMX2.0信息模型为基础构建，兼容SDMX建立的数据模型；

2）RDF数据立方词汇表关注于解决Web上多维度数据的发布，除了支持统计数据的发布外，它还支持其他多维度数据的扩展；

3）RDF数据立方词汇表中沿用了现有的RDF词汇表：概念方案SKOS[23]、核心统计结构SCOVO[24]、元数据Dublin Core Terms[25]、数据访问VoiD[26]、关联代理FOAF[27]、组织ORG[28]等。

SDMX及RDF数据立方词汇表提出的元数据模型很好地表达了元数据的结构以及关联，因此本文结合以上两种技术进行交通物流元数据模型的设计。

### 企业元数据及ETL元数据

1）企业元数据模型

文献[15]中指出，企业元数据可分为业务元数据和技术元数据，图2-1描述了这两个分支相对应的三层概念关系：



图2-1 企业元数据模型[15]

Fig 2-1 Enterprise Business Metadata Model[15]

* 业务元数据

业务元数据顶层表示为“主题域”，往往在分析业务需求时界定；在中层，每一个主题域被分解为业务实体或业务交易，并由底层的业务术语提供细节描述。例如，在交通物流领域中，业务元数据的主题域通常包含综合执法、运输政务、货运市场诚信等，其相应的业务实体可归纳为车辆、违规案例等，由业务术语如车辆牌照、车辆颜色、违规类型、违规时间等加以描述。

* 技术元数据

在技术元数据顶层，技术系统对应业务元数据的主题域进行开发，形成“系统”层；中层通过技术对象来存储业务元数据中的业务实体，同时底层以技术元描述业务术语。例如，对应上述交通物流领域中的业务元数据，技术元数据中的系统可描述为运政系统、诚信系统、综合执法系统等信息管理系统，其技术对象可分为车辆信息表、违规记录表等，技术元素则包括了车辆牌照字段、违规时间字段等。

在实际生产环境中，业务元数据和技术元数据并非相互独立的，它们之间相互关联、相互参照，具体、全面地描述了数据仓库中的数据含义，同时能使这些数据得到正确、及时的检验与维护。

2）ETL元数据

ETL元数据指ETL过程中，所有关于数据抽取、清洗、转换和装载规则的元数据，这些元数据组成ETL任务，通常存储于ETL工具资料库中。

ETL过程应当是采用声明式设计来定义的，即用户需要在ETL任务开始前定义好相关的描述信息和控制信息，在任务执行开始后不再需要人工参与。因此，文献[16]认为，在理想情况下，点到点的数据加载过程应该是由元数据驱动的，所以这些元数据可分为ETL的任务元数据和控制元数据：

* 任务元数据

任务元数据指的是描述ETL任务过程的元数据，包括数据源与数据目标的位置，需要整合的数据项，源和目标的映射关系，完成ETL任务的代理信息，数据加载的方式等，每一项可以根据具体情况加以扩展。

* 控制元数据

控制元数据指的是描述ETL对数据处理方式的元数据。具体来说，控制元数据包括数据转换的规则，清洗的方式，过滤的条件以及检验的方法等元数据。

本论文将研究业务元数据的构建方法及其与技术元数据的映射关系，并建立一体化交通物流元数据库对业务元数据、技术元数据以及ETL元数据进行管理与维护。

### 元数据管理模式

元数据管理涉及到对元数据的获取、存储、维护以及共享的整个生命周期。在交通物流领域中，元数据分布在物流企业以及应用开发商的企业元数据库中，研究[20]指出，元数据获取及管理的方式主要可分为集中式管理和分布式管理两种，如图2-2所示：

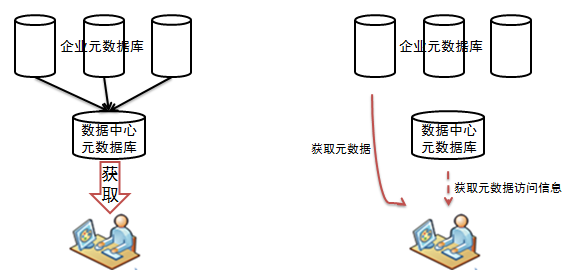


图2-2 元数据获取及管理方式

Fig 2-2 Approaches for metadata harvest and management

1）集中式管理：对于采用数据存储标准而进行规范化存储的元数据，数据中心元数据库可通过数据整合技术，直接从企业元数据库中采集到中心服务器并进行存储管理，进而响应用户对元数据的访问需求。同时，集中式管理方式适用于企业元数据库处于安全因素考虑屏蔽普通用户访问的场景。数据中心通过身份授权及安全验证获取企业允许公开的元数据，提供给外部用户使用。

集中式管理方式具备如下特点：

* 外部企业仅对数据中心元数据库提供元数据，由数据中心对这些元数据进行统一存储于管理，并对外提供元数据访问服务；
* 对于企业元数据的变更，需要采用版本控制的方式对变更进行追踪与管理。

2）分布式管理：对于分散的、非规范化的存储的企业元数据，数据中心元数据库仅维护这些企业元数据的获取方式，包括存储位置、访问条件等。对于用户访问元数据请求，数据中心服务器仅提供这些元数据的获取方式，用户直接连接企业元数据库中获取相应元数据。分布式管理的方式适用于企业元数据允许直接对外公开的场景。

由于本论文研究的元数据管理系统中，处于隐私保护以及安全考虑，企业数据源都是不提供对外开放的，因此本文对企业技术元数据的获取以及管理方式将使用集中式管理方式。

## SDMX信息模型

### 信息模型元素

SDMX提供了描述数据并结构化统计内容的通用模型，提供了一种建模统计数据的方法，并定义了一套元数据结构来支持这个方法[18]，同时这些元数据结构组成了SDMX信息模型。如图2-3所示，SDMX信息模型主要包含以下元素：

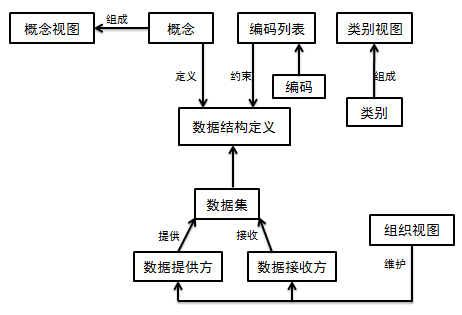


图2-3 SDMX信息模型元素[18]

Fig 2-3 SDMX information model elements[18]

1）数据集：数据被组织成多个离散的集合，每一个都代表着对于一个特定时间段的观察值。一个数据集是覆盖一个特定时间段，共享相同结构的相似数据的集合；

2）数据结构定义：每个数据集都有一组结构化的元数据，在SDMX中，他们被称为数据结构定义（DSD）。它们可以包括概念定义与度量、维度、属性间的关联关系，同时它们还可以包括数据展现方式的信息以及相关的其他结构化或非结构化的元数据信息；

3）概念：概念可分为维度、属性以及度量，作为元数据用于描述数据。其中维度用于描述和标识数据，属性作为附加信息用以补充说明数据，而度量表示了数据内容的呈现方式，如浮点数、正整数或者是字符串方式；

4）概念视图：一个概念视图是一组被维护的概念，它们被使用在数据结构定义以及元数据结构定义中。可以有多个概念视图同时存在，其中有一个核心的代表；

5）编码列表：编码列表枚举代表维度，属性以及其他结构化的信息的一组值。它们可以与其他的结构化元数据结合使用，从而将编码以分层结构组织；

6）分层编码列表：分层编码列表支持编码的分层。编码本身引用到编码列表，因此分层编码列表定义了编码间的层次关系，但其自身不定义编码；

7）类别视图：类别视图由一个类别的层次构成，在SDMX中，它可能包括任何类型的数据和元数据的有用分类。如一个统计主题领域在SDMX中实现为一个类别视图；

8）数据提供方/数据接收方：生成/接收数据或元数据引用的机构称为数据提供方/数据接收方；

9）组织视图：组织以及组织结构可以通过组织视图来定义。组织视图用于维护机构，数据提供方，数据接收方等之间的关系。

表2-1交易数据

Table 2-1 The trade data

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **交易对象** | **B公司** | | | **C公司** | | | **D公司** | | |
| **商品/时间** | **3月** | **4月** | **5月** | **3月** | **4月** | **5月** | **3月** | **4月** | **5月** |
| **电视机** | 20 | 14 | 1 | 5 | 4 | 5 | 25 | 6 | 8 |
| **电脑** | 176 | 18 | 5 | 15 | 12 | 12 | 45 | 24 | 8 |
| **空调** | 17 | 18 | 18 | 17 | 13 | 5 | 8 | 8 | 4 |

例如，表2-1表示A公司在2014年3-5月与BCD公司就电视机、电脑和空调3种商品发生的交易数据。在SDMX信息模型中，如上数据可以被描述为数据集和相应的数据结构定义，表2-2为上述数据的数据集：

表2-2 数据集

Table 2-2 The data set

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 20 | 14 | 1 | 5 | 4 | 5 | 25 | 6 | 8 |
| 176 | 18 | 5 | 15 | 12 | 12 | 45 | 24 | 8 |
| 17 | 18 | 18 | 17 | 13 | 5 | 8 | 8 | 4 |

数据结构定义用以描述元数据，对应SDMX信息模型中的概念，以上例子中交易对象、商品、时间以及数据可被识别为维度，其中时间通常是最常用的维度。

### SDMX-ML

SDMX-ML作为SDMX信息模型的表现形式，提出了一组XML的标签和格式定义，使统计数据及其元数据可以通过XML的方式展现。

数据结构定义以SDMX-ML格式的形式，通过一个通用元数据XML表述，用来理解和处理数据集以及包含分类视图以及组织信息的元数据。

在数据结构定义中，除了对信息模型中的元素进行描述之外，还定义了各元素之间的分组及关联关系：

1）维度和属性等概念在数据结构定义中分组为键簇出现，提供一组特定数据的识别。通常，对于一个特定的数据集，若数据集代表了时间段的统计数据，那么统计频率是第一个描述概念，其他的概念按顺序排列。

2）概念可以进行分组，表示一个有效的数据组。可以根据业务需要通过分组，对某些概念和属性以一个整体进行观察。

3）概念与编码列表的绑定在定义键簇时进行。在数据结构定义中，概念和编码列表是分别进行定义的，只有当某个概念被识别为维度或者属性，并且具有可枚举的取值时，这个概念才会与某个编码列表进行关联。

## RDF数据立方词汇表

### 数据立方体

数据立方体是一类多维矩阵，是对二维表格的扩展[34]，用以从多个角度描述、观察和分析数据。在RDF数据立方词汇表中，数据集是由一系列对数据从不同角度的观察值组成的，这些观察的角度可以被描述为数据的维度，同时通过属性和度量对维度进行补充说明。因此，与SDMX信息模型类似，RDF数据立方词汇表认为数据是由维度、属性及度量组成的，并将它们统称为组件（Component）。

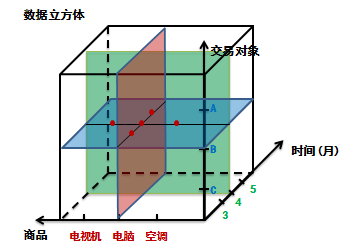


图2-4 2014年3-5月A公司商品交易情况数据立方体

Fig 2-4 Data cube for product trading of company A between March and May in 2014

在对数据集进行观察时，如果指定某个维度的取值，那么数据集将呈现出“切片（Slice）”形态[35]，它代表了对数据集观察的角度，通过切片，数据集会以不同的方式展现出来。

图2-4通过数据立方体的方式展示了表2-1中交易数据的多维度概念。每一个交易数据在图中被表示为一个单独的点，当观察者通过切片来观察数据集时，数据集会以不同的方式展现出来。

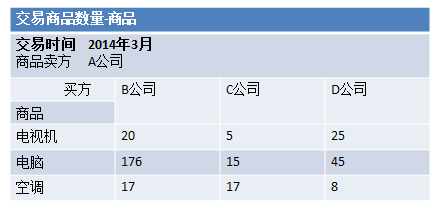


图2-5 2014年3月A公司商品交易量

Fig 2-5 The trading volume of company A in March, 2014

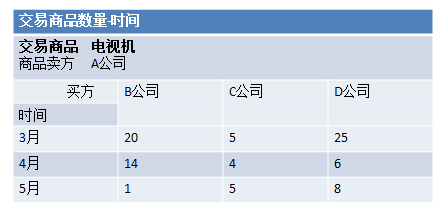


图2-6 2014年3-5月A公司电视机交易量

Fig 2-6 The trading volume in TV of company A in March, 2014

图2-5中，观察者在时间维度上选择了3月份，数据集表现为各买方对每种商品的购买量；而在图2-6中，观察者在交易商品维度上选择了电视机，数据集表现为各买方在不同时间段的购买量。因此，当观察者选择特定维度取值作为切片时，数据集可以呈现出不同的结果。

### 词汇表元素

RDF数据立方词汇表建立在SDMX2.0信息模型的基础上，同时对SDMX信息模型进行了扩展，用于描述多维度数据。RDF数据立方词汇表元素如图2-7所示：

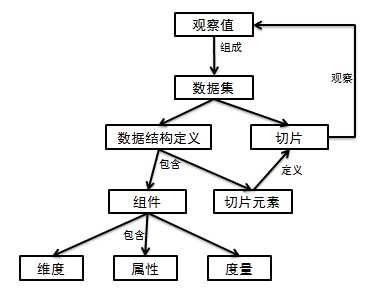


图2-7 RDF数据立方词汇表元素

Fig 2-7 RDF data cube vocabulary elements

1）数据集：数据被组织成多个离散的集合，每个都代表着对于一个特定的切片的观察值，并由数据结构定义描述其元数据信息；

2）切片：切片通过定义切片元素，指定某个维度的取值，用于从不同角度观察数据集得到不同的观察值；

3）切片元素：切片元素指数据结构定义中描述的对数据集进行切片观察的特定维度；

4）观察值：根据切片指定的维度，对数据集从不同角度进行观察，得到侧重点不同的观察值。

5）组件：组件包含维度、属性和度量，并允许由SDMX面向内容指南（SDMX COG）[33]中的通用统计概念进行定义；

6）数据结构定义：数据结构定义由组件和切片元素组成，包括数据展现及观察方式的信息以及相关的其他结构化或非结构化的元数据信息；

如图2-8所示，数据结构定义中，对时间维度的定义可以表示为一个由资源、属性类型和属性值构成的RDF三元组[30]：

* qb前缀：RDF数据立方词汇表命名空间；
* sdmx-concept前缀：SDMX COG中定义概念；
* sdmx-dimension前缀：SDMX COG概念中的维度。

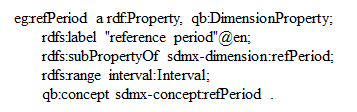


图2-8 RDF三元组

Fig 2-8 RDF triple

在上述三元组中，时间维度的定义直接重用了SDMX COG中的时间定义，并以sdmx-dimension:refPeriod表示。

由于RDF定义的元数据以RDF文档的形式存储，并通过SparQL[31]查询语言进行查询，而本文提出的元数据管理系统通过关系型数据库存储业务元数据，因此本文将结合SDMX信息模型与RDF数据立方词汇表，设计一套适用于交通物流业务领域的元数据模型，并以关系型数据库方式进行存储。

## 交通物流业务元数据模型

### 交通和物流领域业务元数据分类

交通和物流两大业务领域的数据可以根据其相应的业务元数据进行分类，按照企业元数据模型的三层结构，它们的业务元数据可进行如表2-3所示划分：

表2-3交通物流业务元数据分层

Table 2-3 Layers of transport and logistics business metadata

|  |  |  |
| --- | --- | --- |
| **层级** | **交通业务元数据** | **物流业务元数据** |
| 顶层  主题域 | 交通出行、交通违法违规、运输政务、综合执法、车辆维修与救援、陆海运输、甩挂运输管理等 | 货运市场诚信、订单跟踪、物流企业生产信息化等 |
| 中层  业务实体 | 车辆、运输线路、违规案例等 | 货物、物流企业、订单等 |
| 底层  业务术语 | 车辆牌照、路况、违规日期等 | 货物数量、订单日期、订单金额等 |

在交通物流领域的各个主题域中，数据是允许共享的。然而，在技术层面上，数据结构的规范由交通物流企业或管理机构等提供，因此在共享的过程中会出现数据双方规范不一致的情况，造成数据语义不明等问题。本文认为，对于在技术元数据层面规范不一致的问题，应当从业务层面为技术元数据提供统一的业务描述与支持，建立统一的业务元数据，并将各物流企业或管理机构的技术元数据与业务元数据映射，维持数据处理过程中语义的一致性。

### 基于SDMX信息模型扩展的交通物流业务元数据模型

由于交通物流业务数据是多维度的，并且需要支持从不同维度对数据的分析，因此本文结合RDF数据立方词汇表和SDMX信息模型，设计了一个面向交通物流业务领域的元数据模型。该模型基于SDMX信息模型进行扩展，扩展类包括Observation类、Dataset类、DataStructureDefinition类、Slice类、Slicekey类及Dimension类，整体类图如图2-9所示：

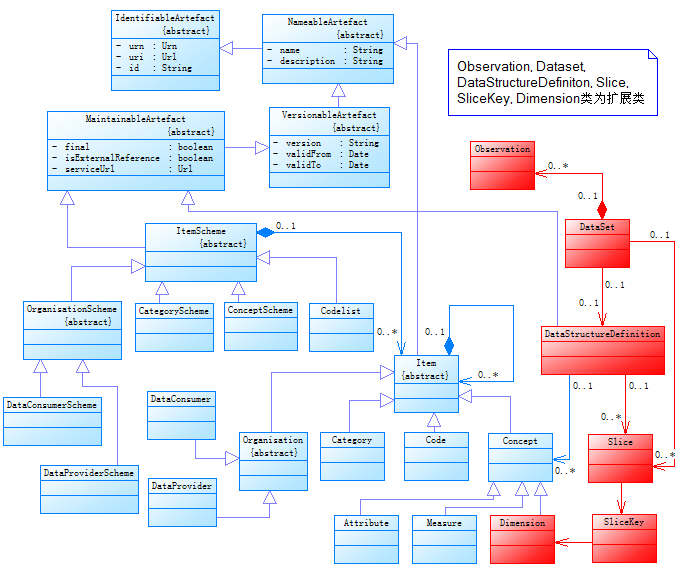


图2-9 交通物流业务元数据模型类图

Fig 2-9 Class diagram of business metadata model

1）IdentifiableArtefact抽象类：该类属于SDMX信息模型中的基本类，表示元数据模型元素的可标识性，其属性有urn表示统一资源名称，uri表示统一资源标识符及元素唯一标识id；

2）NameableArtefact抽象类：该类属于SDMX信息模型中的基本类，继承IdentifiableArtefact抽象类，添加了属性name表示元素名及description表示元素的描述；

3）VersionableArtefact抽象类：该类属于SDMX信息模型中的基本类，继承NameableArtefact抽象类，添加了用于进行版本管理的version属性及版本有效期validFrom和validTo属性；

4）MaintainableArtefact抽象类定义：该类属于SDMX信息模型中的基本类，继承VersionableArtefact抽象类，添加了属性final表示维护元素是草案还是定案，isExternalReference表示元素是否外部引用，serviceUrl表示当该元素是外部引用时，可获取的Web服务地址；

5）ItemScheme抽象类：该类属于SDMX信息模型中的基本类，包含了一系列Item，其主要作用是定义一套分类元素的机制，将元素组织为视图。该类被CategoryScheme、ConceptScheme、OrganisationScheme、Codelist等类继承，与它们相关的Category、Concept、Organisation和Code等类则为Item类的子类；

6）Item抽象类：该类属于SDMX信息模型中的基本类，定义包含在ItemScheme中的元素，如Category、Concept、Organisation和Code等类。同时，Item类具有层级结构，可包含一个或多个子元素；

7）DataSet类：该类为本文基于SDMX信息模型进行的扩展，描述数据集，一个数据集包含一系列观察值，并拥有相应的数据结构定义DataStructureDefiniton和切片Slice；

8）Observation类：该类为本文基于SDMX信息模型进行的扩展，描述数据集的观察值，通常一个数据集包含一系列观察值；

9）DataStructureDefiniton类：该类为本文基于SDMX信息模型进行的扩展，描述数据结构定义。数据结构定义与Concept和Slice相关联，Concept描述数据结构定义中的概念，Slice描述观察的切片维度；

10）Concept类：该类属于SDMX信息模型中的基本类，描述概念，该类由Dimension类、Attribute类及Measure类继承，分别代表概念中的维度、属性及度量；

11）Slice类：该类为本文基于SDMX信息模型进行的扩展，描述数据观察的切片，SliceKey描述切片维度，用于对数据进行不同角度观察与分析；

12）SliceKey类：该类为本文基于SDMX信息模型进行的扩展，描述切片的固定维度，一个切片需要指定一个特定的维度取值。

### 交易订单实例

本节以订单跟踪业务主题域中的交易订单实例为线索，根据交通物流业务元数据模型，构建业务元数据。

如图2-10所示，编号为100000000001的订单完成了一笔交易，交易日期为2014年5月1日，交易双方分别为卖方A公司与买方B公司，订单中包含的交易物品为电视机10台以及音乐CD2张，其单价分别为3000元及30元，该订单总价为30060元。

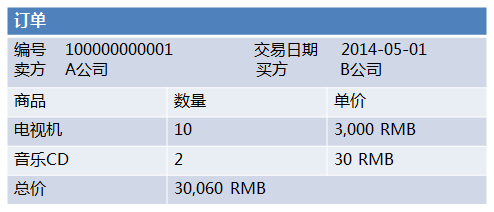


图2-10 交易订单实例

Fig 2-10 An order example

1）概念

概念用于描述此订单，因此，可以根据订单信息首先提取出8个显而易见的概念，分别是：订单编号、交易日期、卖方、买方、交易商品、数量、单价以及订单总价。除此之外，一笔正式的交易订单必须遵循数据的填写规范，即日期格式以及价格单位。

以上10个概念需要进一步分为维度和属性，其中维度用以描述和标识数据，属性作为附加信息用以补充说明数据。当观察者从不同角度去观察订单时，会呈现出不同侧重点的数据，而这样的角度可以用维度来表示。因此，维度包括订单编号、交易日期、卖方、买方、交易商品以及订单总价。通过维度观察，可以清晰地了解到比如“在2014年5月1日发生了多少笔交易”或是“2014年中A公司完成了多少笔交易”等数据。

属性表示了数据集的其他特征，通常用以对数据进行补充说明，如数据的单位。在交易订单实例中，数量、单价、日期格式以及价格单位可被识别为属性，如数量和单价用以补充说明交易商品，日期需要遵循YYYY-MM-DD格式，而价格单位为RMB。图2-11列出了交易订单实例中概念的分类结果：维度包括订单编号、交易日期、卖方、买方、交易商品以及订单总价，属性包括数量、单价、日期格式以及价格单位。



图2-11 订单实例概念

Fig 2-11 Concepts of order example

2）概念视图

概念视图用于维护已经识别出的概念，即订单编号、交易日期、卖方、买方、交易商品、数量、单价、订单总价、时间格式以及价格单位。当不同物流企业的数据公用概念时，这些多源异构的数据就从业务角度进行了关联，可在数据整合过程中被自动识别和匹配。

3）编码列表

在描述上述订单的所有概念中，部分维度和属性的取值可以通过枚举列出，如价格单位以及时间格式。编码列表用以枚举代表维度，属性以及其他结构化的信息的一组值，尤其对于属性而言，最常用的定义属性取值的方法是添加该属性的取值列表。另外，在编码列表中，编码值是需要符合国际标准的，其取值需要遵循ISO标准的定义。

对于属性价格单位，在本例中可以使用人民币RMB、港币HKD、新台币TWD以及美元USD；对于时间格式，通常使用“YYYY-MM-DD”来表示，但是对于持续性的交易，其交易发生在一个时间段中，因此需要使用到以年、月、日等为周期的时间，具体编码列表如图2-12所示：

在业务元数据模型中，使用编码列表来枚举每个概念的具体取值，并以若干编码来进行标识，如对于时间格式的编码列表，P1Y代表1年的时间段，P1M代表1月的时间段，P7D代表1周的时间段，P1D代表1天时间段，同样，这些编码来源于国际标准ISO 8601[37]。

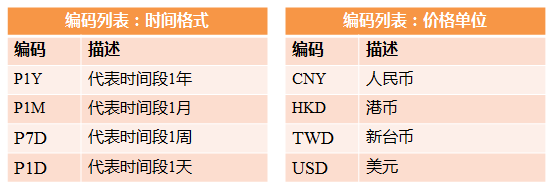


图2-12 订单实例编码列表

Fig 2-12 Code lists of order example

4）数据提供方

数据提供方作为提供数据的机构，通常是交通物流企业或管理机构等，在交易订单实例中，提供订单信息的A公司即为数据提供方。

5）数据结构定义

在业务元数据模型中，数据结构定义可以将数据的结构化元数据表示出来，数据结构定义支持通过一种通用的元数据XML来表达，用来描述和处理数据集中包含的元数据。

在数据结构定义中，除了对信息模型中的元素进行描述之外，还定义了各元素之间的分组及关联关系：

* 维度和属性等概念在数据结构定义中分组为键簇出现，提供一组特定数据的识别。通常，对于一个特定的数据集，若数据集代表了某个时间段的统计数据，那么统计频率是第一个描述概念，其他的概念按顺序排列。
* 概念可以进行分组，表示一个有效的数据组。可以根据业务需要通过分组，对某些维度和属性以一个整体进行观察。
* 维度可以定义切片，用于指定对数据集的一个特定观察角度，根据切片的不同，观察到的结果也会不同。

订单实例中的概念元素通过XML格式在数据结构定义中的表达如下：

< Concepts >

<Concept agencyID="ENTERPRISE\_A" id="Order\_Number">

<Name xml:lang="zh">订单编号</Name>

</Concept>

<Concept agencyID="ENTERPRISE\_A" id="Date">

<Name xml:lang="zh">交易日期</Name>

</Concept>

……

<Concept agencyID="ENTERPRISE\_A" id="Date\_Format">

<Name xml:lang="zh">时间格式</Name>

</Concept>

<Concept agencyID="ENTERPRISE\_A" id=" Price\_Format">

<Name xml:lang="zh">价格单位</Name>

</Concept>

</ Concepts>

由于企业A根据订单的数据结构定义提供了订单数据，因此在描述中必须包含数据提供方，即企业A的机构ID“ENTERPRISE\_A”。所有的概念拥有各自ID以及对自身的描述。

交易订单实例中的编码列表元素通过XML格式在数据结构定义中的表达如下：

<CodeLists>

<CodeList agencyID="ENTERPRISE\_A" id="CL\_Date">

<Name xml:lang="zh">时间格式编码列表</Name>

<Code value="P1Y">

<Description xml:lang="zh">时间段1年</Description>

</Code>

<Code value="P1M">

<Description xml:lang="zh">时间段1月</Description>

</Code>

<Code value="P7D">

<Description xml:lang="zh">时间段1周</Description>

</Code>

<Code value="P1D">

<Description xml:lang="zh">时间段1天</Description>

</Code>

</CodeList>

……

</CodeLists>

同样，通过列举的方式，编码列表中包含对某个概念的枚举值，每个编码包括编码值和描述。上例中，id为“CL\_Date”的编码列表中定义了时间格式的编码值，分别代表某个时间段的表达方式。

在数据结构定义中，可以通过对概念分组将某些维度和属性作为一个整体进行观察，而切片指定了一个特定观察角度，其XML表达如下：

<KeyFamilies>

<KeyFamily agencyID="ENTERPRISE\_A" id="KF\_ORDER" >

<Name xml:lang="zh">订单</Name>

<Components>

<Dimension conceptRef="Order\_Number" />

<Dimension conceptRef="Date" />

……

<Attribute conceptRef="Date\_Format" codelist="CL\_Date" assignmentStatus=" Ma­nd­at­ory "/>

<Attribute conceptRef="Quantity" assignment­Statu­s="Ma­nd­atory" />

<Attribute conceptRef="UnitPrice" assignment­Statu­s="Ma­nd­atory" />

<Group id="Group"> <DimensionRef>Item</DimensionRef>

<AttributeRef>Quantity</AttributeRef>

<AttributeRef>UnitPrice</AttributeRef>

</Group>

<Slice id="Slice">

<DimensionRef>Date</DimensionRef>

</Slice>

</Components>

</KeyFamily>

</KeyFamilies>

在键簇中包含了所有描述订单信息的概念以及概念与编码列表的关联关系。其中，概念可以通过分组作为一个整体进行观察，上例中，交易商品、数量和单价被分为一组，在数据集中，这个分组可以表示订单中某一商品的整体。切片指定某个维度的取值，对数据集进行观察，上例中日期维度作为一个切片，观察者可以用过这个切片观察特定数据，如在一个数据集中分析“公司A在2014年5月1日销售商品的详情”，这种做法排除了在业务上无关数据的影响。

如图2-13所示，应用交通物流元数据模型构建订单的业务元数据后，订单数据可表现为订单业务元数据各元素及其相互关系：

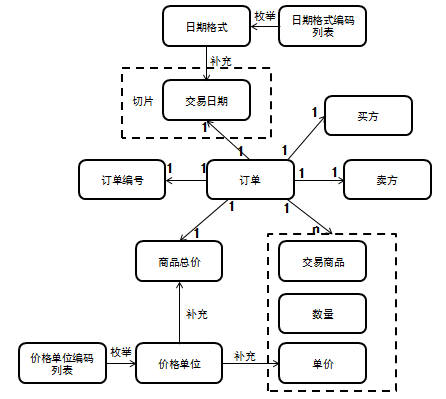


图2-13 订单业务元数据

Fig 2-13 Business metadata of the order

## 业务元数据与技术元数据的映射

### 交通物流技术元数据

交通物流技术元数据用于规范描述交通物流领域技术数据的层次结构关系，如图2-14所示：

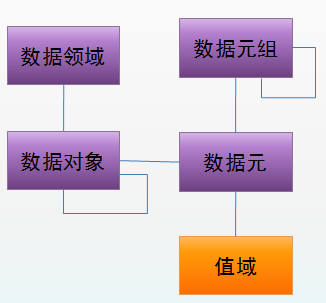


图2-14 数据层次结构

Fig 2-14 The data structure

* 数据领域：用于说明交通物流技术数据的应用领域，并表现为相应的系统对数据进行管理和维护。例如在交易订单实例中，数据应用领域代表交易信息领域，并由企业交易信息管理系统对交易数据进行管理和维护；
* 数据对象：分为简单数据对象和复杂数据对象。简单数据对象是只包含数据元的数据对象，可用于描述数据服务中的数据结构；复合数据对象包含子数据对象和数据元，可用于描述数据通讯中的数据结构。例如在交易订单实例中，企业交易系统中的订单信息表为一个数据对象，它描述了订单的组成结构，包括如交易日期、交易双方之类的数据元以及如商品之类的子数据对象。
* 数据元组：由一系列数据元构成，具有语义完整性和可重用性的特点，并允许嵌套。
* 数据元：位于数据结构的最底层，是可以通过定义、标识、表示和允许值等一系列属性进行赋值的最小、不可再细分的数据单元。例如在交易订单实例中，数据元可以是订单日期字段、商品总价字段等。
* 值域：值域定义数据元的允许值。例如在交易订单实例中，订单信息表中的交易日期字段存在取值范围。

按照企业元数据模型的三层结构表示技术元数据，可进行如表2-4所示划分：

表2-4交通物流技术元数据分层

Table 2-4 Layers of transport and logistics technical metadata

|  |  |  |
| --- | --- | --- |
| **层级** | **交通技术元数据** | **物流技术元数据** |
| 顶层  系统 | 数据领域：交通违法违规信息系统、运输政务服务系统、综合执法系统、车辆维修与救援服务系统等 | 数据领域：诚信管理系统、订单跟踪系统、物流企业生产信息化管理系统等 |
| 中层  技术对象 | 数据对象：执法案例表、车辆信息表等 | 数据对象：企业信息表、货物信息表、订单表等 |
| 底层  技术元素 | 数据元、值域：执法时间字段、当事人证件号码字段、车牌号字段等 | 数据元、值域：企业名称字段、货物分类字段、订单日期字段等 |

### 元数据映射模型

本文在分析了交通物流业务元数据和技术元数据结构的基础上，介绍业务元数据与技术元数据的映射模型。两者间的关系如下：

1）类别视图与数据领域

在交通物流领域的业务元数据中，类别视图表示一个业务主题域，如订单跟踪主题域，该主题域中可包含一系列类别，如某个类别可以包含交易中与订单、商品相关的元数据；而在技术元数据中，数据领域用于说明数据的应用领域，并表现为相应的系统对数据进行管理和维护，如订单跟踪系统。因此，类别视图与数据领域分别从业务和技术角度描述了交通物流数据的划分。

2）数据结构定义与数据对象

数据结构定义将概念、编码、数据提供方等业务元数据通过分组、绑定等方式描述业务数据，如订单数据结构定义；而数据对象从技术角度将数据元或子数据对象组合，以数据库表的形式描述数据，如订单信息表。两者都用于对数据结构进行描述。

3）概念、编码、编码列表与数据元

概念分为维度、属性和度量，是数据结构定义中的重要组成部分，编码组成编码列表，用以枚举概念的取值，而数据元通过定义、允许值等一系列属性来描述数据。因此，技术元数据中的数据元描述的数据信息涵盖了从数据表字段、字段属性、字段取值以及字段格式等全部范围；在业务层面上，业务元数据中的概念以及编码用以描述数据信息。

在交易订单实例中，按照企业元数据对业务元数据和技术元数据进行映射，如图2-15所示：

1）顶层：业务层面上订单所属的交易信息领域与技术层面的交易信息管理系统相映射；

2）中层：订单数据结构定义与数据库中的订单信息表ORDER相映射；

3）底层：组成订单数据结构定义的概念、维度、编码等元素与数据表中的字段、数据类型等相映射，在图中字段ORDER\_NUM与维度Order\_Number相映射，字段TOTAL\_PRICE及其数据类型与维度Total\_Price、属性Price\_Format及编码列表CL\_Price相映射。

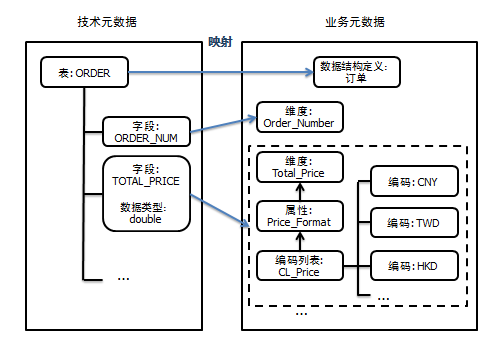


图2-15 订单元数据映射

Fig 2-15 Metadata mapping of the order

因此，技术元数据与业务元数据的映射不仅发生在表、字段与数据结构定义、维度和属性之间，还存在于更细层次元数据之间，并且可能存在技术元数据中多个数据字段类型映射到同一属性的情况，如对于技术元数据中数据库表描述时间的字段，时间格式可映射到同一个业务元数据的属性，并使用相同的编码列表。对于这样的元数据映射关系，如图2-16所示，本文提出了一种元数据映射模型，其特点如下：

1）从顶层到底层，上层元素包含下层所有元素，并且每个层次的元素都有相应层次级别的映射。如在技术元数据中，系统包含所有相关的表，表包含所有相关的字段，系统级别与表级别都存在映射。

2）由于业务元数据与技术元数据的映射需要细分到底层，即编码与数据元级别，因此在存储中需要将复合型表示的元数据进行拆分。如业务元数据的数据结构定义可以包含若干对概念/编码列表，在存储时中，需要将数据结构定义中的概念与编码列表分离，同时通过数据结构绑定维护它们的对应关系。

3）业务元数据与技术元数据的映射建立在拆分后的元数据上，映射之间不存在交叉，如数据结构定义拆分为数据结构定义信息（DSD信息）、概念、编码列表以及它们的绑定关系（DSD绑定），且数据表拆分为字段之后，数据结构信息与数据表的映射不对其他元素产生交叉影响。

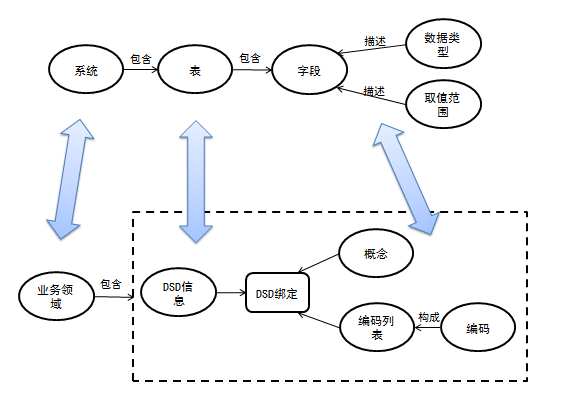


图2-16 元数据映射模型

Fig 2-16 Metadata mapping model

上述映射模型可表示为如图2-17所示的类图：

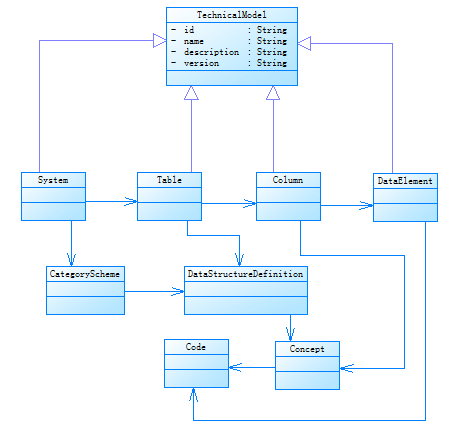


图2-17 元数据映射模型类图

Fig 2-17 Class diagram of metadata mapping model

1）TechnicalModel类：该类定义技术元数据各元素的基本属性，包括id、name、description和version。id表示元素的标识，name为元素名，description为元素的描述，version为元素版本；

2）System类、Table类、Column类及DataElement类：这些类继承TechnicalModel类，分别表示技术元数据中的系统、数据表、字段及描述字段的数据元，它们分别与业务元数据中的CatagoryScheme类、DataStructureDefinition类、Concept类及Code类关联形成映射。

## 基于元数据的交通物流数据整合方法

### 数据整合流程

基于元数据的交通物流数据整合流程如图2-18所示：

1）数据整合任务开发

数据整合利用ETL工具，将数据从数据源经过抽取、转换、装载等过程，存储到目标数据仓库。数据整合任务可描述为一系列ETL元数据，在ETL过程中，根据ETL元数据中的转换规则，源字段中的数据会通过临时数据区进行格式转换，然后根据匹配的目标字段装载到目标数据仓库中去。基于元数据的数据整合是将业务元数据与技术元数据的映射应用到ETL过程中去，在用户开发数据整合任务时，利用业务元数据与技术元数据的映射，自动建立整合字段源和目标的匹配，简化了开发过程。

2）ETL元数据与业务元数据的映射

在业务元数据的构成元素中，包含数据提供方与数据接收方，用于管理数据的流动。ETL任务元数据描述了数据源与数据目标的位置、整合的数据项等信息，因此在开发数据整合任务时需要将被整合的数据的数据源/目标与数据提供方/接收方映射。

3）数据整合任务执行

ETL工具根据开发好的整合任务，完成数据整合过程。在整合过程中，数据从源流向目标，由于存在业务元数据的映射，保证在完成整合后数据在语义上能保持先后一致性，并且数据的来源及去路是可追踪的。

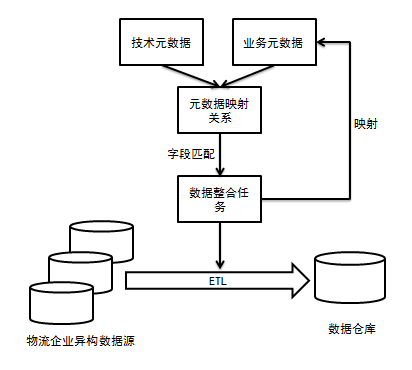


图2-18 基于元数据的交通物流数据整合流程

Fig 2-18 Process of data integration

ETL工具按功能可划分为任务开发工具和任务执行工具。在任务开发时，任务开发工具与一体化元数据库交互，得到元数据映射关系并以此设计任务步骤。基于元数据的数据整合流程顺序图如图2-19所示：

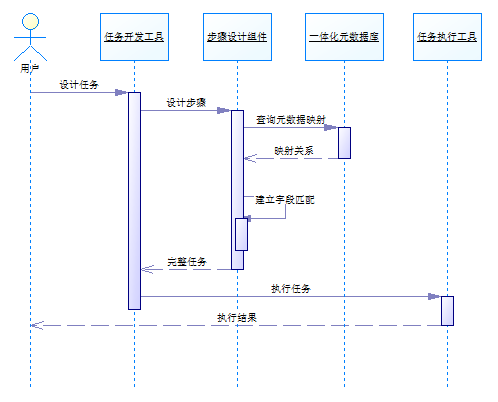


图2-19 基于元数据的数据整合流程顺序图

Fig 2-19 Sequence diagram of data integration based on metadata

首先，用户通过ETL工具的任务开发工具进行数据整合任务开发，包括一系列任务步骤的设计；在步骤设计中，步骤设计组件通过与一体化元数据库的交互，查询得到技术元数据与业务元数据的映射关系，并以此建立整合数据源和目标的字段匹配；在步骤设计完成后，任务开发工具将完整的数据整合任务提交给任务执行工具执行，同时执行结果将返回给用户。

### ETL相关技术分析

在数据整合中，数据源通常在物理上是分散的，在技术上是种类繁多且异构的，因此，首先需要解决不同来源数据中存在的数据唯一性冲突、数据歧义等问题；其次，在数据整合中，保证数据的实时性非常重要，然而数据的定期甚至实时更新存在困难。

ETL技术高效、自动化地解决了上述难题。它可以批量快速地完成数据的抽取、清洗、转换、装载等任务，不仅满足了用户整合异构数据源的需求，同时支持数据的增量更新，一体化地解决了数据整合过程中遇到的各种困难。

ETL中数据转换流程采用生产者-消费者模型，它将转换的步骤分为节点，节点之间通过阻塞队列逐条传递数据，转换执行流程如图2-20所示：

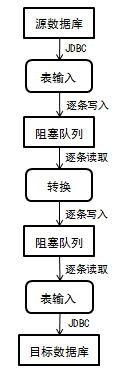


图2-20 转换执行流程图

Fig 2-20 Flow diagram of transformation

1）表输入：表输入以JDBC方式连接数据库查询数据，并将数据返回到一个结果集中，若数据库支持分批读取，当预期结果集较大时，可以分批将数据返回到结果集中，然后从结果集中逐条读取数据并写入到阻塞队列中。表输入的执行流程如图2-21所示：

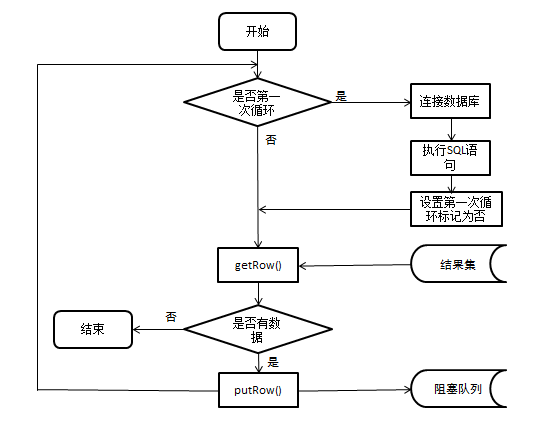


图2-21 表输入执行流程图

Fig 2-21 Flow diagram of table input

2）表输出：表输出将进入阻塞队列的数据通过转换规则转换后，把该次转换包含的数据插入或更新到目标数据库，插入更新是批量执行的；若在整个数据整合任务中存在多步转换，则将该次转换完成的数据继续输出到阻塞队列，为下一步转换提供数据集。表输出的执行流程如图2-22所示：

其中，转换读取转换规则中描述源数据字段和目标数据字段匹配的key-value对，根据阻塞队列中的数据项，生成存储到目标数据库中的对应SQL语句；若转换规则中包含数据合并，数据筛选等规则，则在SQL语句中添加相应的约束。

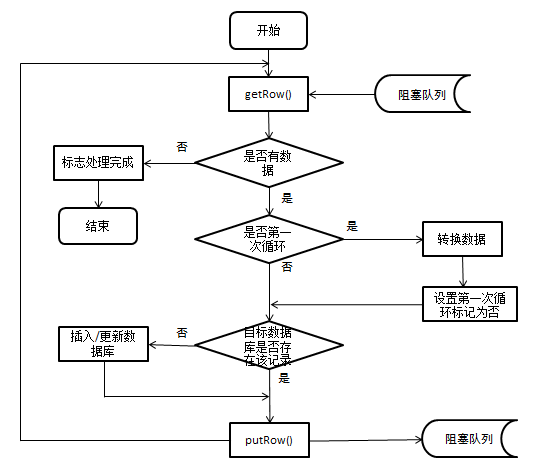


图2-22 表输出执行流程图

Fig 2-22 Flow diagram of table output

用于数据整合的商用ETL软件主要有以下：

1）Oracle公司的Oracle Data Integrator（ODI）

ODI提供了一系列针对组建、部署和管理复杂数据仓库或者作为SOA或商业智能环境中以数据为中心的体系结构的一部分的完整解决方案。另外，ODI结合了所有数据整合的元素（数据流动、数据同步、数据质量、数据管理以及数据服务）来保证信息的实时性、准确性，以及跨复杂系统的一致性。其优点包括对大量数据的高效转换能力以及变更数据的实时捕获能力，并将各种数据整合技术统一，提供了强大的数据完整性控制能力，确保数据的一致性和正确性。

2）IBM公司的InfoSphere

InfoSphere集成了ETL工具、商业智能平台、数据仓库、数据质量工具以及数据整合套件。其中的DataStage扮演了ETL的角色，其优点包括使开发、维护和调试变得容易，可视化流程控制开发，任务调度、审查，资源使用预测，ETL任务性能分析，图形化监控，顺序开发、并行执行，采用基于组件的体系结构等。

3）Informatica公司的PowerCenter

PowerCenter作为专业级数据整合工具，以服务器集群方式提供分布式数据整合，负载均衡，分配工作流和会话等。该工具通过效率驱动、管理变化，确保了数据的准确性，提高数据质量。

在数据整合开源软件中，被企业应用较广泛的是Pentaho公司提供的开源ETL工具Pentaho Data Integrator[36]。它以步骤为单位，用于完成数据的抽取、转换和加载操作，并通过提供步骤管理为用户提供更加个性化的ETL任务设计和执行功能。该工具可通过使用集群操作提高数据传输的性能并且添加对各类型的数据源和目标数据的支持，工具的主要功能如下：

1）ETL任务设计

ETL任务就是对数据的抽取转换和加载的整个过程的描述，ETL执行引擎根据ETL任务执行ETL操作。一个ETL任务包括了数据库连接、任务步骤、节点连接、错误处理、集群schemas、日志记录等部分。ETL任务的可视化设计范例如图2-23所示：

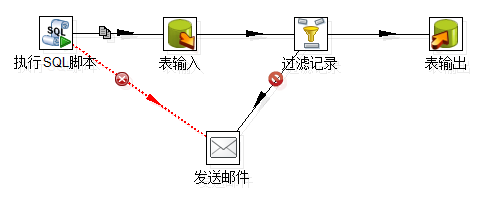


图2-23 ETL可视化任务范例

Fig 2-23 An ETL example

一个ETL任务由多个步骤组成，包括执行SQL脚本步骤、表数据输入步骤、过滤记录步骤、表数据输出步骤和发送邮件等步骤。步骤之间通过“节点连接”相连。“节点连接”包括数据传输节点连接和异常处理节点连接，SQL脚本的执行和表数据的读写通过数据库连接进行设置。

2）ETL步骤使用和管理

ETL步骤作为ETL任务中的基本单位，定义了清洗、转换、加载等操作的具体实现，并支持用户的自定义和可扩展的步骤管理。是本工具的主要特色。

ETL步骤包括如下几个部分：

* 输入：ETL工具支持不同类型的数据源作为输入，支持的数据源类型包括多类型文件、多关系型与非关系数据库、多消息类型以及其他自定义数据类型。
* 输出：ETL工具支持对不同类型的数据目标进行输出，支持的数据目标与输入步骤相同。
* 转换：ETL工具支持对输入步骤获取的数据进行转换操作，并支持转换操作的自定义。转换操作的功能例如记录去重、拆分字段、排序记录、行转列等操作。
* 连接：ETL工具支持对不同的数据来源的数据进行连接操作、包括Merge Join、Sorted Merge、XML Join、合并记录和记录关联等操作。
* 脚本：ETL工具提供多种脚本支持数据处理，脚本类型包括SQL脚本、Javascript脚本、正则表达式、Java代码、公式等。
* 数据仓库支持：支持数据仓库的维度的更新和查询功能。
* 工作流步骤：支持基本的工作流步骤，如邮件通知、流程暂停、过滤记录等。

3）ETL增量数据更新

ETL工具通过在数据源建立触发器或记录时间戳两种方式实现数据的增量更新。建立触发器可以记录源表中数据的变化，记录时间戳可以对比数据更新的时间来确定数据是否需要抽取。

4）ETL集群

对于大数量级的数据进行处理和抽取的时候需要消耗很大的计算资源，ETL工具通过使用集群将数据分区、进行并行处理，提高转换效率。

由于本论文研究的系统需要支持物流企业异构数据的处理，对数据源类型多样性要求较高，并且需要根据业务流程在数据整合流程中添加通知、记录等步骤，因此，本论文选用Pentaho Data Integrator进行数据整合。

## 本章小结

本章首先介绍了元数据标准、分类及管理模式，其次对SDMX信息模型及RDF数据立方词汇表进行分析，之后结合SDMX信息模型及RDF数据立方词汇表，提出交通物流业务元数据模型，并以交易订单实例介绍业务元数据模型的应用，然后阐述业务元数据与技术元数据的映射模型。最后，提出基于元数据的交通物流数据整合方法。

# 系统需求分析与架构设计

本章对某省的元数据管理及数据整合实际业务需求进行分析，并提出元数据管理系统的架构设计方案。



## 业务分析

本文以某省交通物流云平台项目为背景，该平台支撑全省1200家物流企业的日常物流数据的管理与共享。平台建设了统一的数据中心，将大量的业务数据，包括日常运营、订单跟踪、综合执法、运政服务信息等数据整理、归类，形成符合平台业务和商业需要的主题数据库，同时为交通物流行业中物流企业、政府部门、行业协会等提供增值服务。因此应用平台服务层（PaaS）需支持目标数量的物流企业和物流相关人员在平台上发布物流服务供求信息、企业信息、物流订单信息、物流作业信息，而且这些信息都需要统一管理，并通过数据整合后进行分析处理，比如提供给决策分析、KPI考核、运价指数的计算等。

本文提出的系统需要有效支持交通物流数据处理。数据处理通常包含多个方面：数据整合、数据存储以及数据分析挖掘等，而元数据管理涉及到了这些数据处理过程的整个生命周期。本系统主要针对元数据管理及数据整合进行研究，交通物流元数据管理及数据整合业务需求如下：

* 数据中心建设

数据中心建设包括主题数据库的建设与管理，统一提供对平台数据、外部企业数据、每日业务数据等的管理与共享服务。

* 多源数据整合服务

数据提供方包括各应用开发商及平台物流企业用户，数据整合服务需要支持结构化数据、非结构化数据、语义数据和海量数据等多种形式多源异构数据的一体化整合。

本论文提出的系统以满足上述需求而进行设计，同时，本系统还需要解决当前平台及物流企业在元数据管理及数据整合中存在的问题：

1）交通物流数据来自多个不同的数据库中，包括诚信企业，运政、执法等政府部门，以及各类平台企业信息系统。数据中心需要对这些数据进行统一管理与维护。然而这些数据存储的建立以企业内部制定的标准为基础，从结构上存在差异，无法从业务角度达到统一。因此需要通过业务元数据指导管理交通物流数据。

2）平台数据中心整合各企业数据，统一分析处理并对企业提供数据共享等服务，而此过程中的数据整合信息缺乏统一管理手段，造成数据追踪困难。因此需要有效地管理数据整合过程中的数据来源、目标以及数据转换规则等ETL元数据。

3）物流企业的私有数据通常不对外公开，并且通过防火墙阻止外部访问，因此在元数据管理与数据整合时需要设计安全可行的企业私有数据访问方式。

针对上述问题，本文对交通物流元数据管理及数据整合的思路如下：

1）建立交通物流业务元数据

本文对结合SDMX的信息模型与RDF数据立方词汇表，从业务角度将交通物流数据按照类别、维度、属性等加以描述，为存储于数据库中的技术数据添加业务层面的支持，并维护业务元数据与技术元数据之间的关系，以达到有效管理数据的目的。

2）基于元数据的数据整合

在已有的ETL工具的基础上添加对元数据的支持，设计并实现数据整合任务开发时源字段与目标字段的自动匹配，为任务执行时的数据转换提供业务元数据指导。在传统的数据整合过程中，技术元数据指导数据的抽取、转换和加载过程。多个异构数据源的数据通过技术元数据层面的关联，形成语义上的统一，然而这个过程缺乏业务层面的指导，并且需要借助数据字典等工具，因此需要对现在有数据整合工具进行扩展。

用户可通过扩展后的ETL工具，利用技术元数据和业务元数据的映射关系自动建立源字段与目标字段的匹配，同时可以跟踪数据从源到目标的流动过程，保证数据质量。

3）利用前置机作为代理获取企业私有数据

通过设置前置机，并得到物流企业的安全认证及授权后，可以跨越防火墙隔离，获取元数据管理及数据整合任务中所需要的数据。

4）对数据整合任务的统一管理

本文提出的系统需要对数据整合任务运行状态、运行时间、运行周期等信息以及执行整合任务的前置机等硬件资源进行统一管理，保证正常、稳定地为平台物流企业及应用开发商提供元数据管理和数据整合服务。

## 系统需求分析

### 功能性需求

对交通物流数据进行管理，需要熟悉业务的数据管理人员、数据整合开发人员，同时也需要对前置机资源和系统进行管理和维护的系统管理人员。本系统的使用需要以上人员之间的协作，因此，在本系统中，首先有如下描述：

1）一体化元数据库：一体化元数据库是存储包括技术元数据、业务元数据的企业元数据，包括ETL任务元数据、控制元数据的数据整合任务，系统信息以及前置机信息的统一资料库。

2）数据整合任务：数据整合开发人员对数据整合的业务需求进行分析，明确需求后利用数据整合工具进行数据整合任务的开发与使用。

3）前置机：前置机作为从企业私有数据库向数据中心传输数据的代理，应当由系统管理员通过系统进行管理和维护。

根据以上描述，交通物流元数据管理系统功能可分为两部分：元数据管理工具与数据整合工具。系统用户可分为三类：数据管理人员、数据整合开发人员和系统管理人员。数据管理人员、数据整合开发人员和系统管理人员这三类角色的职责如图3-1所示：

1）数据管理人员

* 获取技术元数据：数据管理人员利用系统提供的元数据管理工具进行数据源配置，获取技术元数据至一体化元数据，并进行更新、删除等操作。
* 管理业务元数据：数据管理人员利用系统提供的元数据管理工具对交通物流业务领域的概念、术语等进行建模，并进行更新、删除等操作。
* 管理元数据映射：数据管理人员利用系统提供的元数据管理工具为技术元数据和业务元数据添加映射，并进行更新、删除等操作。

2）数据整合开发人员

* 开发数据整合任务：数据整合开发人员利用数据整合工具开发数据整合任务并制定执行计划。
* 执行数据整合任务：数据整合开发人员利用数据整合工具执行数据整合任务及管理数据整合任务运行状态。
* 管理数据整合任务：数据整合开发人员利用数据整合工具进行数据整合任务的增加、更新及删除等操作。

3）系统管理人员

* 管理前置机：系统管理人员利用系统提供的元数据管理工具进行前置机的部署和监控操作，以及ETL前置机上数据整合引擎的启动和停止操作。
* 管理用户：系统管理人员利用系统提供的元数据管理工具进行数据管理人员和数据整合开发人员的管理。
* 管理一体化元数据库：系统管理人员利用系统提供的元数据管理工具进行系统管理。

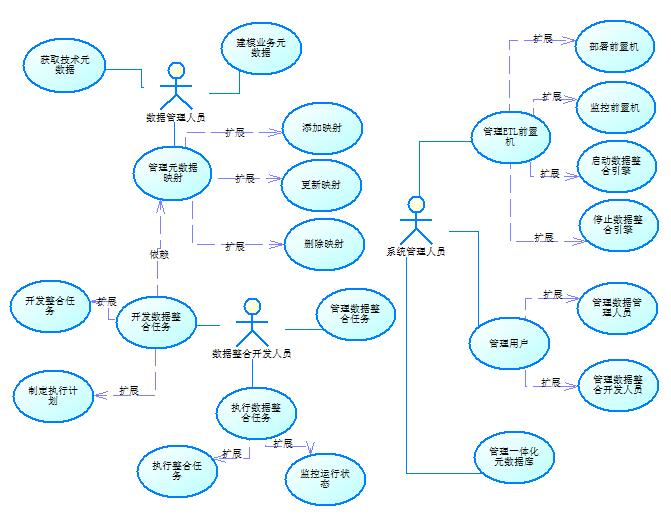


图3-1 元数据管理系统用例

Fig 3-1 Use cases of metadata management system

以上三类角色在数据整合与管理中协作流程如图3-2所示：

1）元数据管理

该过程涉及应用开发商、平台物流企业以及平台数据中心，由数据管理人员获取技术元数据、构建业务元数据，并建立元数据映射，这些管理操作通过元数据管理引擎反映到一体化元数据库中。

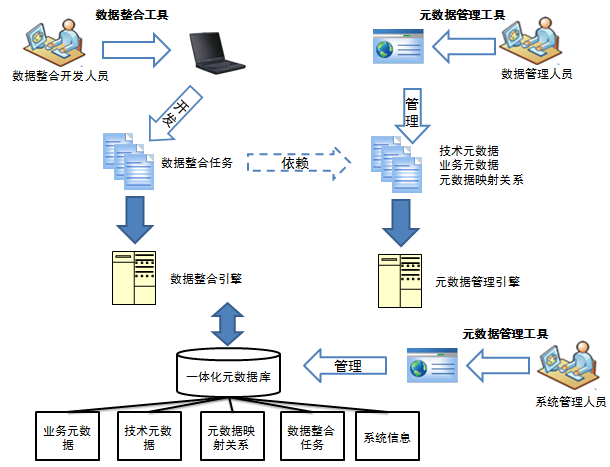


图3-2 元数据管理系统用户协作流程

Fig 3-2 Process of user cooperation

2）数据整合业务分析和整合任务开发

该过程首先由数据整合开发人员通过线下讨论对数据迁移及数据采集等数据整合业务需求进行分析，在明确需求后，通过数据整合工具开发数据整合任务，利用数据管理人员事先建立好的元数据映射，自动建立源字段与目标字段的匹配后，将任务提交给数据整合引擎执行。同时包含数据整合任务配置信息、步骤信息等的数据整合模型以及执行得到的结果等存储于一体化元数据库中。

在数据整合任务开发中，协作主要体现在以下几个步骤：

* 数据管理人员首先从企业元数据库获取技术元数据，并按照业务规则将其与业务元数据相映射；
* 当数据整合开发人员开发数据整合任务时，可以获取技术元数据与业务元数据的映射关系，并以此自动形成数据源字段与目标字段的匹配。

3）系统管理

该过程由系统管理人员根据一体化元数据库中存储的信息，对系统进行管理。

### 非功能性需求

本论文研究并实现的元数据管理系统需要支持拥有1200多家物流企业用户的某省交通物流云平台的日常物流数据的管理及处理，因此，系统在非功能性需求上主要有如下几点要求：

1）可靠性：对企业异构数据的整合要求系统是高可靠的。企业数据在实时变化，因此通过数据整合采集到数据中心这一过程也是实时进行的。为了保证数据能够尽量同步，保持数据的一致性，数据整合工具应当对企业数据的变动进行监控并保持实时更新。由此，系统应当具备高可靠性。

2）性能：由于平台数据量大，访问频繁，因此对数据的管理、整合及处理需要在性能上满足对用户请求的快速响应。在数据整合中将数据整合工具与元数据管理工具相结合，需要保证数据整合任务执行时，能够维持数据整合工具原有的处理速度与处理精度。

3）安全性：由于系统管理的数据涉及到应用开发商及平台物流企业的私有数据库，因此操作必须是安全的，不能影响或泄露企业的私有数据。

## 系统架构设计

### 技术架构

交通物流元数据管理系统技术架构视图如图3-3所示：

1）表现层采用jsp作为页面技术，并通过tiles框架管理页面布局，将复数的jsp页面组合成一个页面进行展示，便于对页面机能进行变更和维护。同时，还采用Spring MVC中的Model、View和Controller以及JQuery管理页面参数传递以及页面显示。

2）业务逻辑层主要包括Service的接口、实现以及Pentaho Data Integrator软件技术。该层定义了一系列访问业务逻辑组件的接口，以及一系列实现了接口的逻辑组件，并调用相应的数据访问组件来完成一系列的业务逻辑处理。这些数据访问组件是由Spring IoC容器统一管理，并注入到相应的业务逻辑组件中去。Pentaho Data Integrator除了通过持久层与使用MySQL的一体化元数据库交互外，还可以通过JDBC直接访问外部企业数据库获取数据。

3）持久层包括DAO接口、SDMX Base、 Hibernate事务处理及持久化以及MetaModel框架。该层定义了一系列通数据访问组件的接口，以及一系列实现了结构的数据访问组件，并通过DAO对外提供数据访问功能。MetaModel是Apache的一个开源数据访问框架，为不同类型的数据存储提供统一的访问方式。SDMX Base为SDMX信息模型的基础包，定义信息模型的基本元素。

4）公共组件包括日志组件及异常处理组件。日志组件用于记录系统运行过程中产生的日志信息，异常处理组件用于检测系统运行过程中出现的异常并进行处理。

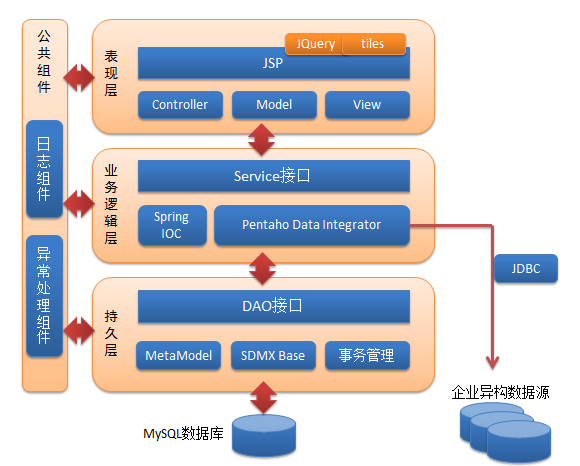


图3-3 元数据管理系统技术架构视图

Fig 3-3 Technical architecture of metadata management system

### 逻辑架构

交通物流元数据管理系统逻辑架构视图如图3-4所示，自顶向下共包括应用程序、服务层、逻辑层及数据层四个部分：

1）应用层由两部分组成，包括Web和客户端：

* Web：为系统用户提供便捷、可视化的Web管理界面。包括技术元数据获取、元数据管理（技术元数据管理与业务元数据管理）、元数据映射、数据整合任务管理、前置机管理以及系统管理。用户可通过Web管理界面管理元数据、建立映射、管理数据整合任务、管理前置机以及管理系统等。
* 客户端：数据整合客户端基于ETL工具Pentaho Data Integrator客户端进行扩展，通过为该客户端开发支持元数据的相关组件，使得通过Web管理界面建立的技术-业务元数据的映射关系能够应用到数据整合中的转换过程，为数据整合提供了业务元数据指导。

2）接口层：接口层为应用层提供接口服务，包括技术元数据获取接口、元数据管理接口（技术元数据管理接口与业务元数据管理接口）、数据整合任务管理接口、前置机管理接口和系统管理接口。

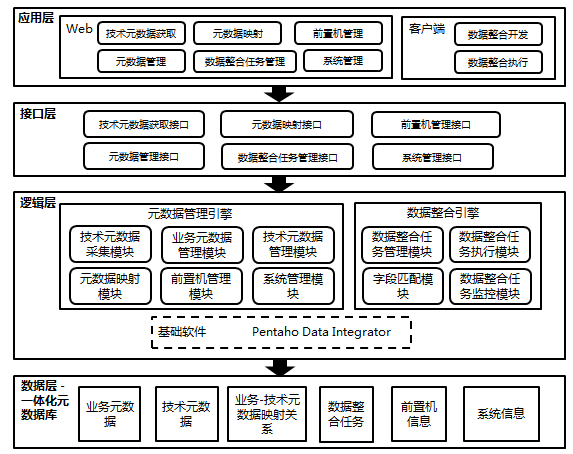


图3-4 元数据管理系统逻辑架构视图

Fig 3-4 Logical architecture of metadata management system

3）逻辑层：逻辑层由两大核心引擎以及基础软件组成：

* 元数据管理引擎：元数据管理引擎包括技术元数据获取、业务元数据管理、技术元数据管理、元数据映射、前置机管理以及系统管理等六个模块，在对业务元数据建模并关联技术元数据后，可与数据整合引擎协作开发数据整合任务。
* 数据整合引擎：数据整合引擎是基于开源ETL引擎进行开发的，在其基础上加入了与元数据管理引擎的协作，支持业务元数据指导的数据整合任务开发。功能模块包括数据整合任务管理模块、数据整合任务执行模块、数据整合任务监控模块及字段匹配模块。
* 基础软件：基础软件为元数据管理系统提供基础的软件服务，包括开源ETL工具Pentaho Data Integrator。

4）数据层：数据层由一体化元数据库构成，为整个元数据管理系统提供数据支持。一体化元数据库存储业务元数据、技术元数据、业务-技术元数据映射关系、数据整合任务、前置机信息以及系统信息，为逻辑层的两大引擎的功能模块服务。

### 部署视图

交通物流元数据管理系统的部署视图如图3-5所示。按照功能划分，服务器资源可分为数据整合服务器、ETL前置机集群、元数据管理服务器、元数据管理前置机以及系统管理服务器。按资源从属划分，上述服务器资源属于系统内服务器，系统外资源包括提供企业数据及元数据的各应用开发商及企业的私有数据库服务器。

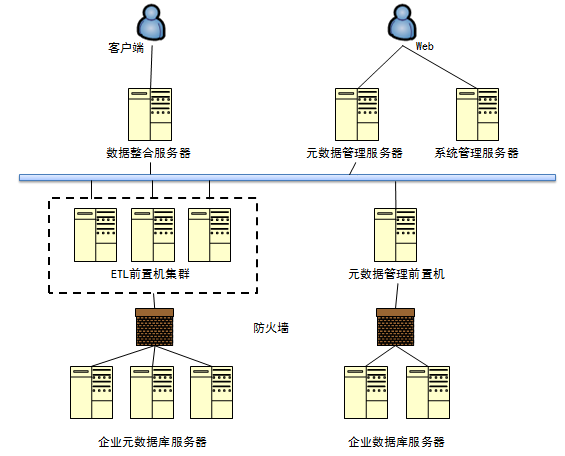


图3-5 元数据管理系统部署视图

Fig 3-5 Deployment view of metadata management system

前置机需要通过企业授权认证，其设置目的是保证能够跨越企业防火墙或网络隔离，安全获取对外部屏蔽的企业数据及元数据。

在交互上，数据整合开发人员通过客户端与数据整合服务器进行交互，提交开发好的数据整合任务，由数据整合服务器对任务进行验证后，提交给ETL前置机，执行数据整合任务；同时，数据管理人员通过Web与元数据管理服务器进行交互，在获取技术元数据时，需要通过元数据管理前置机从企业元数据库中获取相应的技术元数据；系统管理人员通过Web与系统管理服务器进行交互，进行前置机管理及系统管理。

### 系统实现环境

交通物流元数据管理系统的实现环境如表3-1所示：

表3-1 交通物流元数据管理系统实现环境

Table 3-1 System implementation environment

|  |  |
| --- | --- |
|  | 系统实现环境 |
| JVM | Sun JRE/JDK 7.0 |
| 数据整合客户端 | 基于Pentaho Data Integrator客户端进行开发 |
| Web容器 | Tomcat 7.0.42 |
| Web前端 | Bootstrap 2.3.2 |
| Web后端 | Spring MVC 3.2.4 |
| 一体化元数据库 | MySQL 5.5 |
| 企业数据库 | Oracle Database 11g |
| 企业元数据库 | Oracle Database 11g |
| 数据整合引擎 | 基于Pentaho Data Integrator引擎进行开发 |

系统实现环境具体说明如下：

1）元数据管理系统应用层实现基于Java语言，使用JDK版本为7.0。

2）数据整合工具的客户端基于Pentaho Data Integrator客户端进行开发，在原有的基础上开发元数据映射相关组件，并以此开发数据整合任务存储于一体化元数据中。

3）元数据管理工具提供的Web管理界面的实现上，前端采用Bootstrap，后端采用Spring MVC，并以Tomcat作为Web容器。

4）数据整合工具中的数据整合引擎基于Pentaho Data Integrator引擎进行开发，在原有基础上加入了与元数据管理引擎的协作，添加了元数据映射获取、字段自动匹配等功能。

5）应用开发商及平台物流企业私有数据库与元数据库使用Oracle数据库，一体化元数据库使用MySQL。

## 本章小结

本章首先对交通物流元数据管理系统进行了需求分析，包括业务分析、功能性需求分析以及非功能性需求分析。然后从系统的技术架构、逻辑架构、部署视图以及系统实现环境等四个方面进行了系统架构设计。

# 系统核心功能模块的设计与实现



## 技术元数据获取

### 技术元数据获取方法

企业技术元数据的获取方法可分为两种：静态获取与动态更新。

1）静态获取

在传统的元数据管理工具中，技术元数据的抽取是通过对数据库中元数据表的查询操作获取数据库表结构、数据类型等元数据。

由于各企业使用的数据库产品种类繁多，对元数据的定义和获取方式各不相同，因此通常需要通过适配器进行技术元数据的抽取。如图4-1所示，适配器通过对不同数据库访问接口的包装，为用户提供统一的数据库访问方式。

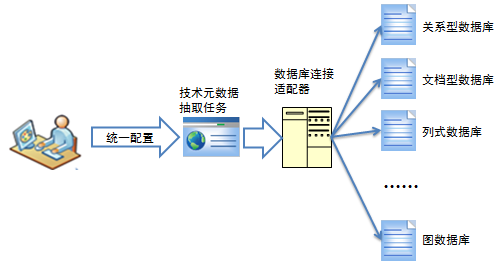


图4-1 技术元数据静态获取方式

Fig 4-1 Static harvest of technical metadata

然而静态获取元数据的方式存在一定的问题：

* 元数据无法支持增量更新

各物流企业以及应用开发商对数据的管理方式各不相同，并且随着业务的变更可能造成对数据库的修改或者重构，然而这种更新一旦发生，如果没有人工进行通知重新获取元数据，那么数据中心中存储并管理的企业技术元数据将仍然处于未更新状态。

未更新的元数据是无效的，并且使用这样的元数据可能会对数据整合过程造成影响，从而导致数据的错误。

* 无法进行元数据关联的更新

对技术元数据的改动通常会造成更大的影响，如已经关联过业务元数据的技术元数据一旦变更，关联关系也必须同步更新，否则会造成语义上的含义不明或是多重含义等问题。

本文针对上述问题，结合数据整合技术，对企业技术元数据的传统获取方式提出改进，对于变更进行动态更新，同时维护元数据之间映射关系的有效性。

2）动态更新

动态更新方式利用数据整合技术，动态捕捉企业技术元数据层面发生的变更，并将变更实时反馈至数据中心的一体化元数据库，保证元数据的一致性。

本文设计了一个动态更新的方法，用以解决静态获取技术元数据中存在的问题：

* 设计数据整合任务从需要提供技术元数据的物流企业或应用开发商的数据源中抽取其数据库技术元数据；
* 将技术元数据按照数据中心的存储方式进行转换并加载到数据中心一体化元数据库中；
* 同时更新一体化元数据库中存储的元数据映射关系，对任何变更的关系进行更新；
* 利用日志记录动态更新过程中产生的错误，便于分析与解决。

图4-2为动态更新技术元数据的数据整合过程示例，STRAT步骤定义任务的循环方式，在每一次任务中，分别完成技术元数据的更新与元数据映射关系的更新，保证数据中心管理的元数据的一致性。同时任务通过循环执行，可以确保数据的实时性，避免数据过期或者失效，保证物流企业与数据中心整体的数据一致性。

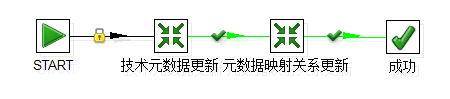


图4-2 技术元数据动态更新示例

Fig 4-2 An example of dynamic technical metadata update

图4-3为上述任务中技术元数据更新的具体分解步骤，在企业数据源向数据中心元数据库传输数据的过程中，可以通过完整性验证保证数据传输的可靠性。

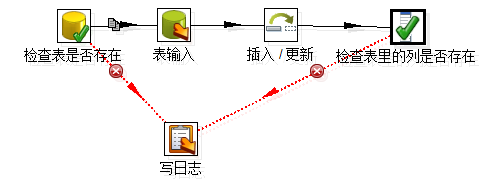


图4-3 技术元数据更新步骤分解

Fig 4-3 Steps of technical metadata update

由于动态更新使用了数据整合技术，相比于静态的元数据抽取方式，其具备以下优点：

* 动态更新利用数据整合的增量数据更新特性，可实现自动化的元数据抽取；
* 对于映射关系复杂的元数据，动态更新的方式能够自动识别每一项变更，并且对任何与变更项相关的数据进行更新，避免了静态更新中可能导致的疏漏；
* 动态更新的错误反馈机制完善，利用数据整合工具的执行日志，可以追踪元数据抽取过程中可能发生的错误。

### 技术元数据获取模块

技术元数据获取模块实现的功能为，定义数据源、设计获取步骤、生成获取任务并执行。技术元数据获取模块的类图如图4-4所示：

1）Step类：Step类表示获取任务的步骤，类中包括以下属性：name为步骤名称，datasource为步骤中用到的数据源，schema为数据源中存储数据的方案，description为步骤的具体信息。

2）Task类：Task类表示技术元数据获取的任务，类中包括以下属性：name为任务名称，steps为任务步骤的集合。

3）StepDesigner类：StepDesigner类负责任务步骤的设计，该类包含以下关键方法：

* setStep方法：setStep方法用于为添加步骤配置信息；
* setPeriod方法：setPeriod方法用于为步骤添加执行周期，以满足周期性的业务元数据动态更新。

4）StepChecker类：StepChecker类用于对步骤进行验证。它的关键方法为checkStep方法，该方法对步骤的具体信息进行验证，检测步骤是否符合ETL步骤设计的要求。

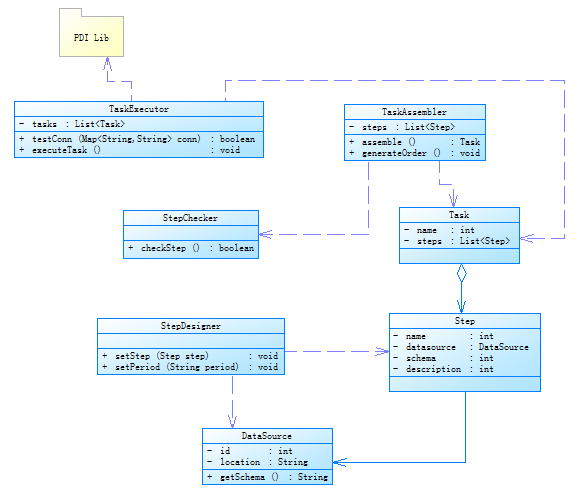


图4-4 技术元数据获取模块类图

Fig 4-4 Class diagram of technical metadata harvest module

5）TaskAssembler类：TaskAssembler类用于将步骤组装成任务。类中包含的属性为：steps为任务步骤的集合，该类包含以下方法：

* assemble方法：assemble方法用于对步骤进行组装，形成一个业务元数据获取任务；
* generateOrder方法：generateOrder方法根据任务，为每一个步骤进行排序，确定其先后顺序；

6）TaskExecutor类：TaskExecutor类作为任务执行器，负责任务的执行。该类收集需要执行的业务元数据获取任务，并依赖PDI库执行任务。类中包含的属性为：tasks为任务的集合，该类包含以下方法：

* testConn方法：testConn方法用于检测任务步骤中所有相关的数据源是否可访问，该方法的设计目的是在任务实际执行前先进行网络检测，在数据源都是可连接的状态下才进行任务执行，避免执行时由于数据无法获取导致的任务挂起，造成资源浪费；
* executeTask方法：executeTask方法依赖PDI库执行任务。

技术元数据获取模块的构建顺序图如图4-5所示。

首先客户端调用StepDesigner进行技术元数据获取任务的步骤设计，添加步骤信息以及每个步骤的执行周期。在完成步骤设计后，客户端调用TaskAssembler对设计完成的步骤进行组装，在组装过程中，TaskAssembler于StepChecker进行交互，检查步骤的完整性，并将通过检查的步骤按照执行顺序进行组装，得到一个完整的任务。最后，客户端调用TaskExecutor将任务提交并执行，并得到返回结果。

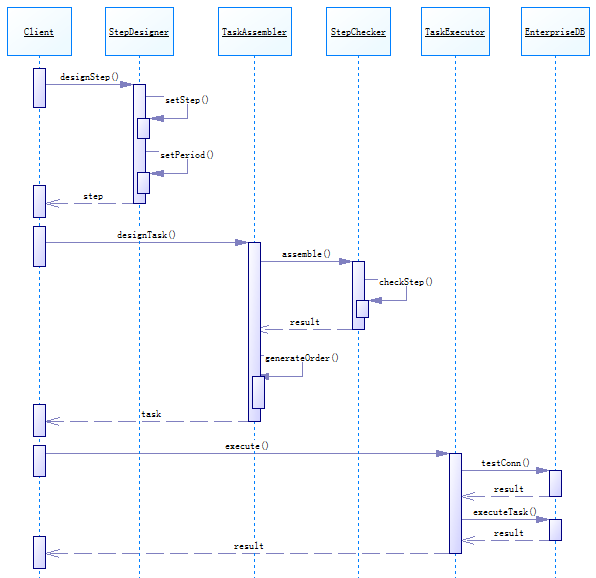


图4-5 技术元数据获取顺序图

Fig 4-5 Sequence diagram of technical metadata harvesting

## 业务元数据管理模块

业务元数据管理模块用于管理一体化元数据库中存储业务元数据构建中的信息模型元素及其关系。业务元数据管理模块所用到的数据库表及其之间的关系如图4-6：

数据库的设计遵循复合型元数据拆分存储的原则，agency表用于存储业务元数据提供组织信息，因此，概念、编码列表、数据结构定义、类别视图等元素在存储时需要指定定义该业务元数据的组织，例如平台数据中心。category与categoryscheme表存储类别与类别视图，concept表与conceptscheme表存储概念与概念视图，它们通过conceptmapping表形成多对多关系。同样地，code表与codelist表存储编码与编码列表，并通过codemapping形成多对多关系。datastructuredefinition表存储数据结构定义的基本信息，并通过dsdmapping与概念及编码列表形成绑定关系。slice表与concept表及datastructuredefinition表通过slicemapping表形成多对多关系，用来存储数据结构定义中的切片。

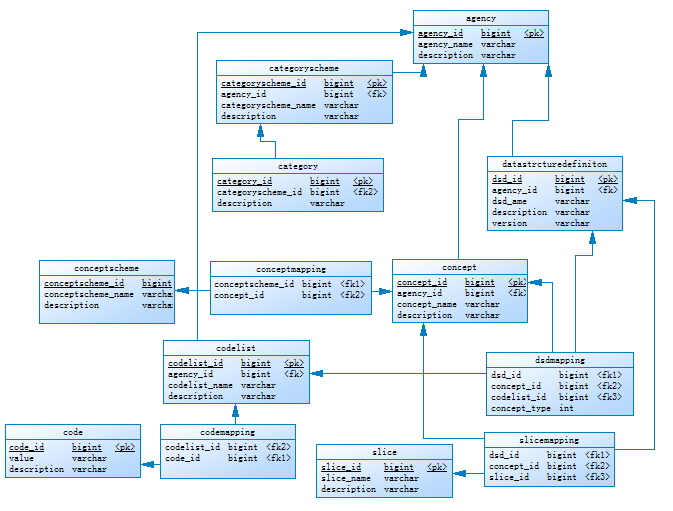


图4-6 业务元数据管理模块数据库设计

Fig 4-6 Database design of business metadata management module

## 元数据映射模块

元数据映射模块用于将技术元数据映射到对应业务元数据上，该模块按照业务元数据和技术元数据分层，将每个层级的相应元素通过映射进行关联。

元数据映射模块的类图如图4-7所示：

1）TechnicalModel类：TechnicalModel类为所有技术元数据构成元素的父类，定义每个技术元数据构成元素的基本信息，该类主要属性为：id为元素标识，name为元素名，description为其描述，version为技术元数据构成元素的版本。

2）System类：System类描述技术元数据中的系统，该类主要属性为：db\_url为系统数据库地址，db\_type为数据库类型，username为数据库登录用户名，password为登录口令，schema为数据库Schema。该类包含以下关键方法：

* containTable方法：containTable方法用于为系统添加相关的数据表；

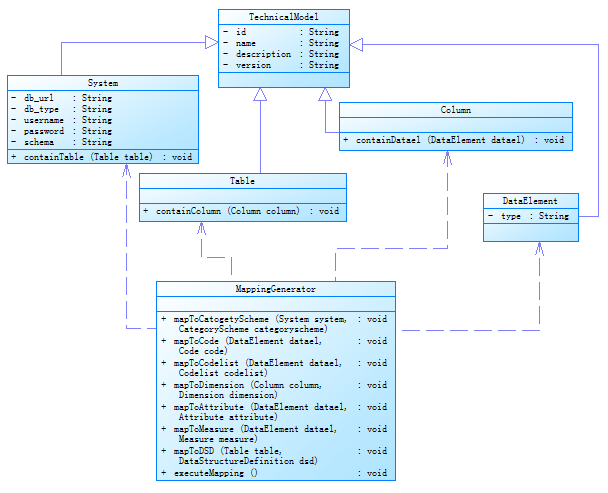


图4-7 元数据映射模块类图

Fig 4-7 Class diagram of metadata mapping module

3）Table类：Table类描述技术元数据中的数据表，该类包含以下关键方法：

* containColumn方法：containColumn方法用于为数据表添加相关的数据字段；

4）Column类：Column类描述技术元数据中的数据字段，该类包含以下关键方法：

* containDatael方法：containDatael方法用于为数据字段添加描述字段的数据元，如数据类型、值域等；

5）DataElement类：DataElement描述技术元数据中的数据元，该类主要属性有type，该属性描述数据元分类，如数据类型、值域等。

6）MappingGenerator类：MappingGenerator类负责将业务元数据映射到技术元数据的相关处理逻辑，是元数据映射建立子模块中最重要的类，该类包含以下关键方法：

* mapToCategoryScheme方法：mapToCategoryScheme方法用于将技术元数据中的系统映射到业务元数据中的类别视图；
* mapToCode方法：mapToCode方法用于将技术元数据中的数据元映射到业务元数据中的编码；
* mapToCodelist方法：mapToCodelist方法用于将技术元数据中的数据元映射到业务元数据中的编码列表；
* mapToDimension方法：mapToDimension方法用于将技术元数据中的表字段映射到业务元数据中的维度；
* mapToAttribute方法：mapToAttribute方法用于将技术元数据中的数据元映射到业务元数据中的属性；
* mapToMeasure方法：mapToMeasure方法用于将技术元数据中的数据元映射到业务元数据中的度量；
* mapToDSD方法：mapToDSD方法用于将技术元数据中的数据表映射到业务元数据中的数据结构定义；
* executeMapping方法：executeMapping方法用于执行元数据的映射，建立映射关系。

## 字段匹配模块

字段匹配模块的作用是，将转换规则中描述源数据字段和目标数据字段匹配的key和value依据元数据映射进行自动匹配，即若源和目标字段映射到同一业务元数据，那么作为key的源字段将自动与作为value的目标字段匹配。本文以开源ETL工具Pentaho Data Integrator为基础进行改进，字段匹配模块的类图如图4-8所示：

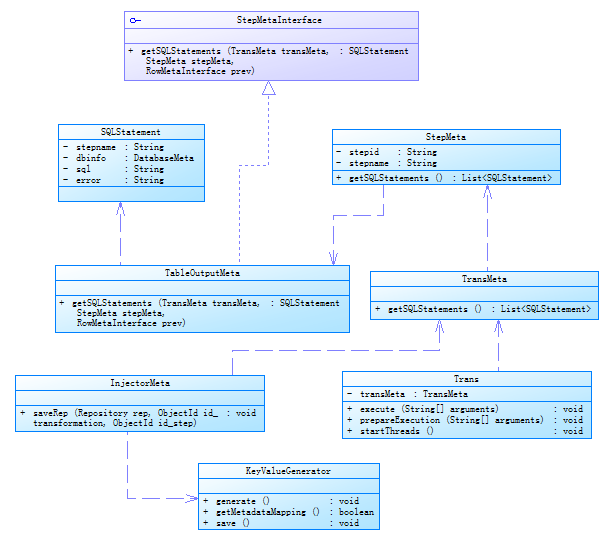


图4-8 字段匹配模块类图

Fig 4-8 Class diagram of field mapping module

1）StepMetaInterface接口：StepMetaInterface接口定义转换步骤基本方法，它有许多实现类，在字段匹配中涉及到的主要方法声明为：

* getSQLStatements方法：getSQLStatements方法用于得到转换数据时在目标数据库执行的SQL语句；

2）SQLStatement类：SQLStatement类描述转换数据时在目标数据库执行的SQL语句，包含下列属性：stepname为步骤名称，dbinfo为需要执行SQL语句的数据库元信息，sql为相应的SQL语句，error为错误信息。

3）TableOutputMeta类：TableOutputMeta类描述表输出步骤信息，该类实现StepMetaInterface接口，其关键方法为：

* getSQLStatements方法：该方法为接口中getSQLStatements方法的具体实现。该方法从一体化元数据中获取转换步骤中源与目标数据字段的key-value对，并按照转换规则得到需要执行的SQL语句；

4）StepMeta类：StepMeta类描述转换步骤信息，包含下列属性：stepid为转换步骤识别id，stepname为转换步骤名，该类的关键方法为：

* getSQLStatements方法：getSQLStatements方法得到步骤中包含的需要执行的SQL语句的列表；

5）TransMeta类：TransMeta类描述转换任务信息，该类的关键方法为：

* getSQLStatements方法：getSQLStatements方法根据转换任务中包含的步骤，得到每一个步骤包含的需要执行的SQL语句的列表；

5）Trans类：Trans类代表一个转换任务，该类的主要属性为transmeta，没一个转换任务通过转换任务信息描述，其关键方法为：

* execute方法：execute方法将执行一个转换任务，在转换任务执行开始时，先做一系列准备，如改变任务状态为“准备中”、设置任务执行时间、记录日志等，之后启动线程执行任务；
* prepareExecution方法：prepareExecution方法处理转换任务执行准备过程中的逻辑，它将一个转换任务拆分为多个步骤，得到每个步骤的信息，并将它们排序为顺序执行步骤和并行执行步骤；
* startThreads方法：startThreads方法在执行准备工作完成后，启动线程执行转换任务，包括数据的读及写，转换规则的分析处理等。

6）InjectorMeta类：InjectorMeta类将定义好任何关于数据整合的信息写入一体化元数据库中，在转换过程中，其关键方法为：

* saveRep方法：saveRep方法将用户设计的转换任务存入一体化元数据；

7）KeyValueGenerator类：KeyValueGenerator类用于在转换任务中根据元数据映射对源和目标字段进行匹配，其关键方法为：

* getMetaDataMapping方法：getMetaDataMapping方法根据一体化元数据库中存储的元数据映射关系，根据源字段映射的业务元数据，查询目标字段中存在相同业务元数据映射的字段；
* generate方法：generate方法根据getMetaDataMapping方法检查得到的结果，为源和目标字段添加key-value匹配。
* save方法：save方法将建立好匹配的转换任务存入一体化元数据。

ETL字段匹配模块中，字段匹配执行流程顺序图如图4-9所示。

客户端首先向InjectorMeta发送转换保存请求，InjectorMeta与KeyValueGenerator交互请求转换字段的匹配，KeyValueGenerator查询一体化元数据，根据元数据映射关系生成匹配后调用save方法，将整合任务保存到一体化元数据库。

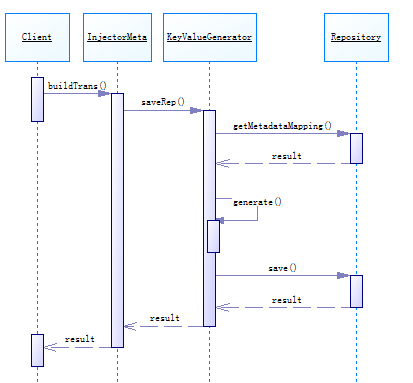


图4-9 字段匹配构建顺序图

Fig 4-9 Sequence diagram of technical field mapping building

ETL字段匹配模块中，转换执行时根据匹配生成SQL语句的执行流程顺序图如图4-10所示。

客户端首先向Trans发送转换执行请求，Trans与TransMeta交互进行转换执行准备。之后TransMeta根据字段匹配信息，想StepMeta请求涉及到字段匹配步骤中匹配执行的SQL语句，StepMeta对步骤中的表输出提出获取匹配执行的SQL语句，最后由 TableOutputMeta从一体化元数据库中得到SQL语句。在完成所有转换执行准备工作之后，进行转换执行并将结果返回给客户端。

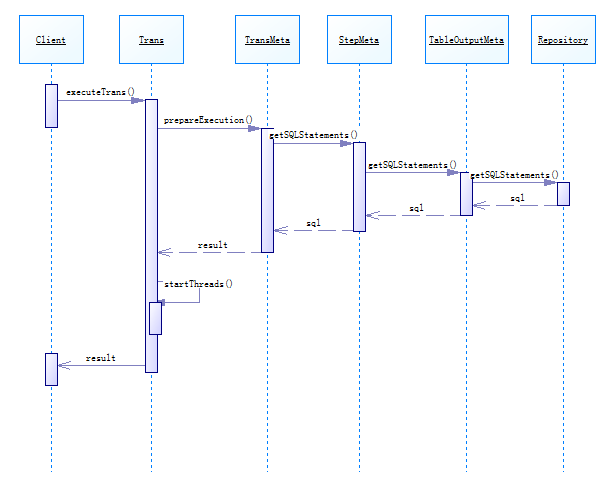


图4-10 SQL语句生成顺序图

Fig 4-10 Sequence diagram of SQL generating

## ETL前置机管理模块

ETL前置机是企业私有数据库向数据中心传输数据的代理服务器，数据整合开发人员通过数据整合客户端，将开发好的数据整合任务发送到ETL前置机上，然后由数据整合引擎执行相关任务。数据整合客户端与ETL前置机之间通过SSH方式进行通信，整合任务的传输通过直接远程执行SSH命令的方式进行。

ETL前置机以集群的方式部署，采用Master/Slave架构，如图4-11所示。每个前置机节点上运行一个数据整合引擎实例，数据整合开发人员首先将任务发送给ETL前置机集群中的Master节点，Master根据整合任务，将需要处理的数据进行分区（Partition）后发送给Slave进行任务的执行，每个Slave单独执行自己分配到的任务，在完成后通知Master并由Master反馈给数据整合开发人员。

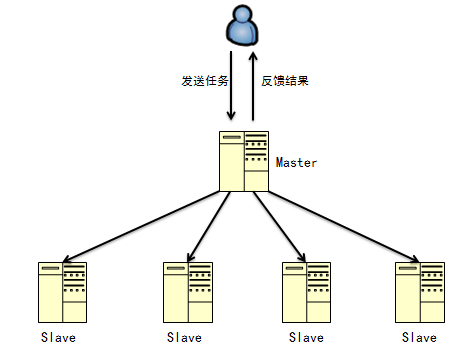


图4-11 ETL前置机集群架构

Fig 4-11 Architecture of the agent cluster

ETL前置机管理模块中的关键功能点为数据整合引擎自动部署和数据整合任务集群分配，其类图如图4-12所示：

* 数据整合引擎自动部署：该功能通过管理环境与前置机之间的SSH通信，数据整合引擎发送到前置机集群上，并指定一个端口运行整合引擎实例，接受数据整合任务。
* 数据整合任务集群分配：该功能为数据整合任务添加集群信息，然后向前置机集群中的Master发送的数据整合任务，之后Master根据Slaver列表，，将任务分交到各个Slave手中。

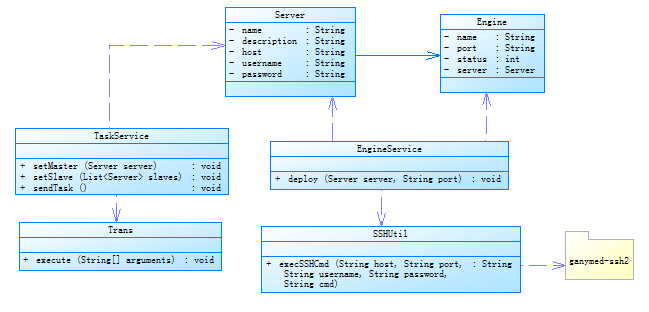


图4-12 ETL前置机管理模块类图

Fig 4-12 Class diagram of agent management module

1）Server类：Server类描述前置机服务器，包含以下属性：name为服务器名称，description为服务器描述，host为服务器主机名，username为SSH登录服务器的用户名，password为SSH登录服务器的密码。

2）Engine类：Engine类描述数据整合引擎实例，包含以下属性：name为引擎实例名称，port为引擎运行端口，server为运行引擎实例的服务器，status为引擎当前运行状态。

3）EngineService类：EngineService类提供数据整合引擎部署功能，其关键方法为：

* deploy方法：deploy方法根据参数设置的前置机服务器以及端口号，通过SSH访问服务器，传输数据整合引擎相关资源，并开启相应的端口，运行整合引擎。

4）TaskService类：TaskService类负责为数据整合任务添加集群执行信息，其关键方法为：

* setMaster方法：setMaster方法为该任务指定集群执行的Master。
* setSlave方法：setSlave方法为该任务指定集群执行的一系列Slave。
* sendTask方法：sendTask方法将数据整合任务发送给数据整合引擎进行执行。

5）SSHUtil类：SSHUtil类是远程执行SSH命令的辅助工具类，该类使用SSH2的Java实现ganymed-ssh2包实现SSH命令的远程执行。该类包含以下方法：

* execSSHCmd方法：execSSHCmd方法远程执行SSH命令，其依赖ganymed-ssh2包。该方法接收的参数有host、port、username、password和cmd，分别为远程机器的ip地址、端口号、登录用户名、密码和需要执行的命令，并返回值执行命令后的输出

前置机管理模块数据整合引擎自动部署的执行流程顺序图如图4-13所示。

客户端首先向EngineService发送部署引擎的请求，该类通过deploy方法，调用辅助工具类SSHUtil中的execSSHCmd方法，并得到执行结果的输出，返回给用户。

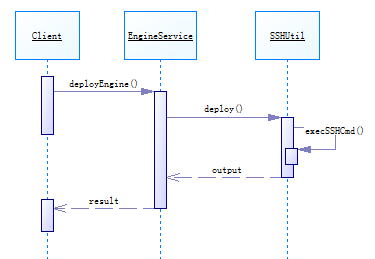


图4-13 数据整合引擎部署顺序图

Fig 4-13 Sequence diagram of data integration engine deployment

## 本章小结

本章重点介绍了元数据管理系统核心功能模块的设计与实现，包括技术元数据获取模块、业务元数据管理模块、元数据映射模块、字段匹配模块及ETL前置机管理模块。

# 系统测试及应用

本章对交通物流元数据管理系统进行系统测试及应用。首先对系统进行功能测试和性能测试，然后以某省交通物流公共服务平台中综合执法数据及货运市场信用数据管理与整合为例，对系统进行可行性及有效性分析。



## 系统测试

### 功能测试

本次测试将元数据管理系统分为两部分进行测试：元数据管理工具和数据整合工具。

1）元数据管理工具

元数据管理工具的测试用例及测试结果如表5-1所示：

表5-1 元数据管理工具的测试用例及测试结果

Table 5-1 Test cases and results for the metadata management tool

|  |  |  |
| --- | --- | --- |
| 功能模块 | 测试用例 | 测试结果 |
| 技术元数据获取模块 | 设计技术元数据获取任务、执行任务后得到企业数据库的技术元数据 | 得到企业技术元数据并存储到一体化元数据库中  测试通过 |
| 业务元数据管理模块 | 构建业务元数据，并对其进行查询、修改与删除等操作 | 能够正确得到业务元数据，并且查询、修改与删除等操作可以正常执行  测试通过 |
| 技术元数据管理模块 | 对技术元数据进行查询、修改与删除等操作 | 查询、修改与删除等操作可以正常执行  测试通过 |
| 元数据映射 | 对业务元数据与技术元数据建立映射，并对映射关系进行维护 | 生成元数据映射，并可以进行维护  测试通过 |

表5-1 元数据管理工具的测试用例及测试结果(续)

Table 5-1 Test cases and results for the metadata management tool (Cont.)

|  |  |  |
| --- | --- | --- |
| 功能模块 | 测试用例 | 测试结果 |
| 前置机管理 | 自动部署前置机、启停数据整合引擎、配置数据整合任务集群 | 能够自动部署前置机并启动数据整合引擎实例，能够正确为数据整合任务添加集群执行信息  测试通过 |
| 系统管理 | 管理一体化元数据库中系统用户等信息 | 能够正常管理系统  测试通过 |

2）数据整合工具

数据整合工具的测试用例及测试结果如表5-2所示：

表5-2 数据整合工具的测试用例及测试结果

Table 5-2 Test cases and results for the data integration tool

|  |  |  |
| --- | --- | --- |
| 功能模块 | 测试用例 | 测试结果 |
| 数据整合任务开发 | 使用数据整合客户端设计数据整合任务并存储于一体化元数据库中，同时可以对整合任务进行管理 | 生成可正确执行的数据整合任务  测试通过 |
| 数据整合任务执行 | 执行设计好的数据整合任务对多个异构数据源进行数据整合，在执行时通过元数据映射进行字段自动匹配 | 任务成功执行且目标数据库得到正确的数据  测试通过 |

### 性能测试

本节对元数据管理系统中数据整合工具进行性能测试，结果包含两部分，添加元数据支持后的数据整合工具的性能测试结果，以及其与改进前的ETL工具测试结果的对比。

1）测试环境

本次测试中，ETL前置机集群共3台前置机，其配置都为CPU频率2.27GHz，内存8G；数据源与数据目标都使用Oracle Database 11g数据库；前置机集群与数据源通过100M局域网相连，并通过20M电信宽带与数据目标相连。

2）测试结果及分析

表5-3为使用添加元数据支持后的数据整合工具执行数据整合任务所得到的测试结果。结果数据表明，随着数据量增长，执行任务的耗时呈线性增长，而ETL前置机的CPU资源消耗率处于稳定状态，其中作为master的前置机由于仅进行任务分发，因此几乎不消耗CPU资源。

表5-3 数据整合工具性能测试结果

Table 5-3 Performance test results for the data integration tool

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 测试项  数据量  （万行） | 耗时（秒） | 前置机CPU资源消耗率（百分比） | | |
| 2 | 24.8 | <1 | 13.1 | 12.9 |
| 5 | 64.3 | <1 | 13.0 | 13.1 |
| 20 | 251.3 | <1 | 13.3 | 13.1 |
| 100 | 1258.7 | <1 | 13.3 | 13.0 |

图5-1为改进后的数据整合工具与改进前的测试结果对比图，结果表明，在数据整合任务的执行环节，改进后的工具与改进前在性能方面几乎没有差别。然而在数据整合任务开发环节上，由于改进后的工具提供了源字段与目标字段的自动匹配，因此，提高了数据整合任务的开发效率。

图5-1 性能测试结果

Fig 5-1 The results of performance test

## 系统应用

### 运行实例

综合执法系统中存储关于交通违规案件的记录，货运市场诚信系统中存储关于企业信用数据。交通物流云平台数据中心需要整合这些数据，并提供给物流企业进行相关业务分析。本文以综合执法数据和企业信用数据的管理和整合为例，介绍元数据管理系统中，元数据管理工具和数据整合工具的应用。

1）元数据管理工具的应用

* 技术元数据获取

图5-2为获取技术元数据的步骤，首先检查当前综合执法系统数据库和货运市场诚信系统数据库的连接情况，然后分别从这两个数据库中获取元数据并存储到数据中心一体化元数据库中。该任务采用动态更新的策略，以天为周期循环执行，捕获当天发生过变动的元数据并更新到一体化元数据库中。

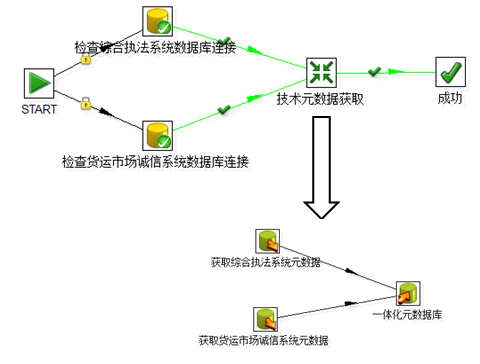


图5-2 技术元数据获取步骤设计

Fig 5-2 Steps of technical metadata harvesting

图5-3为元数据管理工具的Web管理界面中展示的综合执法系统和货运市场诚信系统技术元数据，这些技术元数据以表-字段的树形结构组织。

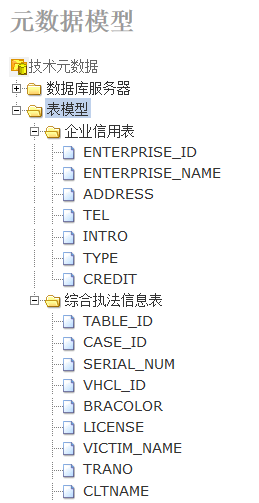


图5-3 技术元数据

Fig 5-3 The technical metadata

* 创建业务元数据

元数据管理人员可通过元数据管理工具的Web管理界面建立业务元数据。业务元数据根据元数据模型分为机构、类别、概念、编码、编码列表及数据结构定义等，元数据管理人员通过工具可以创建交通物流领域相关的业务元数据。一套完整的业务元数据创建步骤如下：

1）创建机构：创建“数据中心”、“执法机关”、“信用服务提供商”机构表示元数据的提供组织，所有其它业务元数据都需要指定维护它们的机构；

2）创建类别：根据综合执法和货运市场诚信的业务领域创建“综合执法”及“货运市场诚信”类别，多个类别可组成类别视图；

3）创建概念：概念代表业务术语，在综合执法和货运市场诚信业务领域中，概念有“案件编号”、“车辆颜色”、“车牌号码”、“当事人”、“企业”、“企业信用度”等；

4）创建编码列表：若存在特定取值的概念，则可以创建编码和编码列表来指定概念的枚举值；

5）创建数据结构定义：创建综合执法信息的数据结构定义，指定概念的类型（维度、属性或度量）并绑定相应的编码列表。在数据结构定义中，可以创建切片表示观察的特点维度。

图5-4为元数据管理人员根据综合执法和货运市场诚信业务创建的概念。其中界面左侧为树形组织的元数据模型元素列表及对应的元数据，右侧为对左侧元素的具体管理内容。

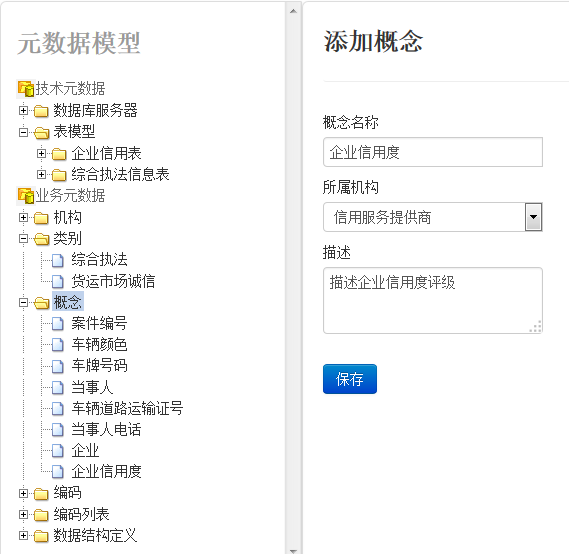


图5-4 创建概念

Fig 5-4 Creation of concepts

完成创建概念、编码列表等业务元数据元素后，元数据管理人员可以此创建数据结构定义。图5-5为元数据管理人员创建了综合执法信息数据结构定义后，绑定数据结构中的概念和编码列表，同时为概念指定类型，包括维度、属性和度量。如维度有案件编号、车牌号码、当事人、车辆道路运输证号等，属性有车辆颜色等。



图5-5 创建数据结构定义

Fig 5-5 Creation of data structure definition

* 建立元数据映射

在得到技术元数据并且创建完业务元数据之后，可以按照两者之间的关联添加技术元数据与业务元数据的映射。技术元数据的一个表模型中包含相应的字段，图5-6中列出字段名称和字段的数据类型以及映射的业务元数据。



图5-6 建立元数据映射

Fig 5-6 Metadata mapping establishing

2）数据整合工具的应用

数据中心需要整合综合执法系统中的综合执法数据，提供给物流企业进行相关业务分析，本次数据整合中，源数据表为综合执法系统中的ZF\_CASES表和货运市场诚信系统中的ENTERPRISE\_CREDIT表，目标数据表为CREDIT\_INFO表，数据表、数据字段和它们映射的业务元数据如表5-4所示。

表5-4 数据整合字段

Table 5-4 Data integration fields

|  |  |  |
| --- | --- | --- |
| 源数据表：ZF\_CASES join ENTERPRISE\_CREDIT  目标数据表：CREDIT\_INFO | | |
| 源字段名 | 目标字段名 | 映射业务元数据 |
| ZF\_CASES .SERIAL\_NUM | SERIAL\_NUM | 案件编号 |
| ZF\_CASES .LICENSE | LICENSE | 车牌号码 |
| ZF\_CASES .VICTIM\_NAME | VICTIM | 当事人 |
| ZF\_CASES .BRACOLOR | BRACOLOR | 车辆颜色 |
| ZF\_CASES .TRANO | TRA\_NO | 车辆道路运输证号 |
| ZF\_CASES .PHONE | PHONE\_NUM | 当事人联系电话 |
| ZF\_CASES .PSNCTCTYPE | PSNCTCTYPE | 证件类别 |
| ZF\_CASES .ID\_NUM | ID\_NUM | 当事人有效证件 |
| ZF\_CASES .CASE\_TIME | CASE\_TIME | 违法时间 |
| ENTERPRISE\_CREDIT.ENTERPRISE\_NAME | UNIT\_NAME | 当事人单位名称 |
| ENTERPRISE\_CREDIT.CREDIT | UNIT\_CREDIT | 当事人单位信用度 |

数据整合任务的开发步骤流程如图5-7所示。由于在综合执法系统和货运市场诚信中，存在关于执法案件以及企业信用的完整信息，如果部分数据并非包含在而数据中心整合需求中，则可以对这些数据进行级联Join操作和选择性过滤，最后输出到数据中心的数据仓库中。这些数据后续可以通过数据中心为各应用开发商或平台物流企业提供服务。

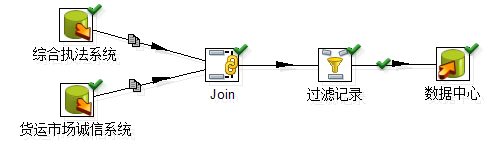


图5-7 数据整合步骤

Fig 5-7 Steps of data integration

在传统的数据整合任务开发过程中，需要数据整合开发人员参照数据字典等工具为源字段和目标字段建立匹配。然而，本系统中的数据整合工具添加了对元数据的支持，根据已有的元数据映射关系，可以快速、自动建立字段之间的匹配。图5-8为自动匹配后的结果。

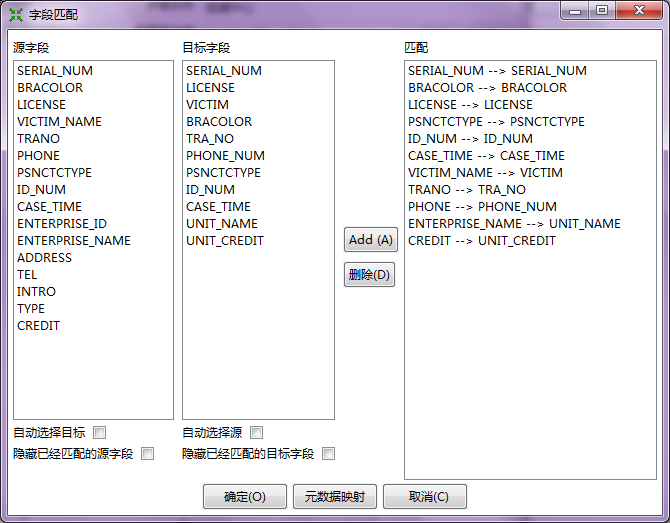


图5-8 匹配字段

Fig 5-8 Field mapping

将开发完成的数据整合任务提交给数据整合引擎执行后，数据成功从综合执法系统和货运市场诚信系统整合到数据中心。任务的执行结果如图5-9所示。

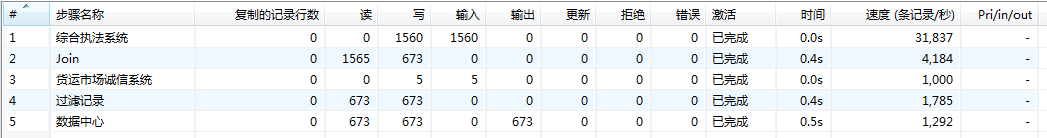


图5-9 整合任务执行结果

Fig 5-9 The execution result of the data integration task

同时，通过数据整合任务的Web管理界面，可以查询到综合执法数据整合任务相关任务信息，如图5-10所示。



图5-10 数据整合任务信息

Fig 5-10 Information of the data integration task

### 应用效果及分析

本系统通过在某省交通物流云平台试运行，为平台数据中心统一管理和整合数据提供服务，并形成符合平台业务和商业需要的主题数据库，目前涉及到的数据如表5-5所示。同时，交通物流行业中的物流企业、应用开发商、政府部门、行业协会等通过本系统提供的功能从数据中心进一步获取数据用于业务分析，比如提供给决策分析、KPI考核、运价指数的计算等。

表5-5 元数据管理系统的应用

Table 5-5 Application of the metadata management system

|  |  |  |
| --- | --- | --- |
| 主题数据库 | 数据管理/整合项 | 数据来源 |
| 平台物流基础信息主题数据库 | 企业信息、车辆信息、司机信息 | 运输政务系统、物流企业生产信息系统、货运枢纽节点信息系统、城市及短途运输信息系统、车辆维修与救援信息系统 |
| 交通基础设施主题数据库 | 货运站场信息、物流线路信息、码头信息、航线信息、高速服务区、高速公路、全省道路、港口信息 | 交通物流基础设施运行信息系统、港口物流信息系统 |
| 交通出行信息主题数据库 | 交通气象信息、实时路况信息、客运线路信息 | 交通出行信息系统 |
| 货运市场诚信主题数据库 | 业户诚信信息、从业人员诚信信息、企业诚信信息 | 货运市场诚信系统、陆海运输信息交易系统、综合执法系统 |
| 订单综合信息主题数据库 | 订单综合信息 | 物流企业生产信息系统 |
| 应急物流信息主题数据库 | 应急车辆信息、应急设备信息、应急仓储信息、应急物资信息 | 应急物流信息系统 |
| 货运枢纽节点信息主题数据库 | 枢纽档口信息、停车位信息 | 货运枢纽节点信息系统 |
| 交易信息主题数据库 | 货源信息、车源信息 | 陆海运输信息交易系统 |

本系统为数据管理人员提供了一套企业元数据一体化管理的方案及管理工具，并为数据整合开发人员提供了更加便捷的数据整合任务开发流程。目前，在整个交通物流云平台中，每日发生的数据整合次数约为35000次（包含增量更新），整体数据量在TB级别以上，系统的应用效果如下：

* 通过元数据管理系统提供的元数据管理工具，构建交通物流业务元数据，使得异构企业技术元数据的管理更加快速便捷，数据中心整体数据管理能力得到提高；
* 通过元数据管理系统提供的数据整合工具，在数据整合开发中，将元数据与数据整合结合，简化了任务的开发过程。如图5-11所示，基于元数据的数据整合将传统数据整合中高开销的字段匹配步骤自动化，提高了开发效率；

图5-11 数据整合开发效率

Fig 5-11 Data integration development efficiency

* 通过设置前置机集群，元数据管理系统对元数据管理和数据整合请求响应迅速，并能够有效跨越企业防火墙，安全地获取企业私有数据。

结果表明，本系统能够有效支持交通物流大数据处理任务，增强了数据管理能力并提高了数据整合效率。因此，本系统是可行且有效的，达到了设计目标。

## 本章小结

本章首先对元数据管理系统进行功能测试和性能测试。然后以综合执法数据和货运市场诚信数据管理与整合的应用场景为例，分析了本系统的应用效果，并验证了本系统的可行性及有效性。

# 总结与展望



## 工作总结

本文以某省交通物流云平台项目为背景，该项目包含一系列交通物流数据处理需求，如对物流企业、应用开发商及平台数据中心的数据进行数据整合及统一管理。本文针对以上需求，设计并实现一个可有效支持交通物流数据处理任务的元数据管理系统。

本文工作主要包括以下几个方面：

1）深入研究元数据、SDMX信息模型、RDF数据立方词汇表及ETL等相关技术。

2）结合SDMX信息模型及RDF数据立方词汇表，提出了一个面向交通物流领域的业务元数据模型。本文基于SDMX信息模型及RDF数据立方词汇表，对交通物流领域的术语、业务实体进行建模，并构建业务元数据；同时建立并维护业务元数据与技术元数据的映射关系。

3）提出了基于元数据的交通物流数据整合方法，将业务元数据应用到交通物流领域的数据整合。在用户开发数据整合任务时，利用业务元数据与技术元数据的映射，自动建立整合字段源和目标的匹配，简化了开发过程。

4）设计并实现了元数据管理系统，该系统包括元数据管理工具与数据整合工具两部分。元数据管理工具提供技术元数据获取、元数据管理、元数据映射、前置机管理及系统管理功能；数据整合工具提供数据整合任务开发与数据整合任务执行功能。

5）将该系统应用于某省交通物流云平台，为数据中心及平台物流企业等提供元数据管理和数据整合服务，并通过试运行验证系统的可行性及有效性。

## 下一步工作

本课题后续研究工作包括以下几个方面：

1）为数据管理人员提供通用的交通物流业务元数据库，减少其工作量。

2）进一步研究业务元数据模型，使其适用范围更广，更易于扩展。

3）为常用的业务场景定制数据整合任务模板，使数据整合开发人员能够更快速高效地进行数据整合开发。

# 参 考 文 献

[1] Maurizio Lenzerini (2002). "Data integration: A theoretical perspective"[C], PODS 2002. pp. 233–246.

[2] Wikipedia. ETL[OL]. http://en.wikipedia.org/wiki/Extract,\_transform,\_load.

[3] Oracle. "Oracle Data Integrator: An Executive Overview"[Z], January 2007.

[4] P. Vassiliadis. "A survey of extract-transform-load technology"[C], International Journal of Data Warehousing and Mining (IJDWM), 5(3):1-27, 2009.

[5] Xin Luna Dong, Barna Saha, Divesh Srivastava. "Less is More: Selecting Sources Wisely for Integration"[C], VLDB, 2012.

[6] Yagiz Kargin, Milena Ivanova, Ying Zhang et al. "Lazy ETL in Action ETL Technology Dates Scientific Data"[C], Proceedings of the VLDB Endowment, Vol. 6, No.12, 2013.

[7] U. Dayal et al. "Data integration ows for business intelligence"[C]. In EDBT, pages 1-11. ACM, 2009.

[8] N. Swartz. "Gartner warns rms of 'dirty data' "[J]. Information Management Journal, 41(3), 2007.

[9] Amr Ebaid, Ahmed Elmagarmid et al. "NADEEF: A Generalized Data Cleaning System"[C], VLDB, 2013.

[10] Alexander Albrecht. "METL: Managing and Integrating ETL Processes"[C], VLDB, 2009.

[11] Xiufeng Liu, Christian Thomsen, Torben Bach Pedersen. "MapReduce­based Dimensional ETL Made Easy"[C], VLDB, 2012.

[12] Martin Hentschel, Laura Haas, Renee J. Miller. "Just-in-time Data Integration in Action"[C], VLDB, 2010.

[13] Xin Luna Dong, Divesh Srivastava. "Big Data Integration"[C], VLDB, 2013.

[14] National Information Standards Organization, Rebecca Guenther and Jaqueline Radebaugh (2004). Understanding Metadata[M]. Bethesda, MD: NISO Press. [ISBN](http://en.wikipedia.org/wiki/International_Standard_Book_Number) [1-880124-62-9](http://en.wikipedia.org/wiki/Special:BookSources/1-880124-62-9). Retrieved 2 April 2014.

[15] Kamlesh Mhashilkar, Jaideep Sarkar. "Information Integration: Metadata Management Landscape"[Z]. White Paper, April 2009.

[16] 孙安健. 通用ETL工具的研究与设计[D]. 复旦大学, 2012.

[17] 张晓钧. 基于CWM的元数据管理系统在电子政务中的设计和实现[D]. 上海交通大学, 2012.

[18] SDMX[OL]. Available at: http://sdmx.org/

[19] Ranjeet Devarakonda , Giriprakash Palanisamy , Bruce E. Wilson et al. "Mercury: reusable metadata management, data discovery and access system"[J], Earth Sci Inform(2010)3:87-94.

[20] "Mercury (2008) Distributed metadata management, data discovery and access system"[OL]. Oak Ridge national laboratory. Available at: http://mercury.ornl.gov, Accessed August 2009.

[21] Juan Wang, Dan Feng, Fang Wang et al. "MHS: A distributed metadata management strategy"[C], The Journal of Systems and Software 82 (2009) 2004–2011.

[22] Mauricio A. Hernandez et al. "Data Exchange with Data-Metadata Translations"[C], VLDB, 2008.

[23] Antoine Isaac, Ed Summers. SKOS Simple Knowledge Organization System Primer. 18 August 2009[OL]. Available: http://www.w3.org/TR/skos-primer

[24] The Statistical Core Vocabulary[OL]. http://sw.joanneum.at/scovo/schema.html

[25] Dublin Core metadata initiative. [Dublin Core metadata element set, version 1.1](http://dublincore.org/documents/dcmi-terms/)[OL]. July 1999. Available: http://dublincore.org/documents/dcmi-terms/

[26] Keith Alexander, Richard Cyganiak, Michael Hausenblas et al. [Describing Linked Datasets with the VoID Vocabulary](http://www.w3.org/TR/void/)[OL]. 3 March 2011.

[27] Dan Brickley, Libby Miller. [FOAF Vocabulary Specification 0.98[OL].](http://xmlns.com/foaf/spec/) 9 August 2010. Available: http://xmlns.com/foaf/spec/

[28] Dave Reynolds. [The Organzation Ontology](http://www.w3.org/TR/vocab-org/)[OL]. 25 June 2013. Available: http://www.w3.org/TR/vocab-org/

[29] RDF[OL]. Available: http://www.w3.org/RDF/

[30] RDF Data Cube Vocabulary[OL]. http://www.w3.org/TR/vocab-data-cube/

[31] Steven Harris, Andy Seaborne. [SPARQL 1.1 Query Language](http://www.w3.org/TR/sparql11-query/)[OL]. 21 March 2013. http://www.w3.org/TR/sparql11-query/

[32] Pascal Hitzler; Markus Krötzsch; Bijan Parsia; Peter Patel-Schneider; Sebastian Rudolph. [OWL 2 Web Ontology Language Primer (Second Edition)](http://www.w3.org/TR/owl2-primer/)[OL]. 11 December 2012. http://www.w3.org/TR/owl2-primer/

[33] SDMX Contnent Oriented Guidelines[OL]. <http://sdmx.org/?page_id=11>

[34] ["Cybertec releases OLAP cubes for PostgreSQL"](http://www.postgresql.org/about/news.653)[OL]. PostgreSQL. 2006-10-02. Retrieved 2008-03-05. http://www.postgresql.org/about/news.653

[35] "OLAP and OLAP Server Definitions"[OL]. The OLAP Council. 1995. Retrieved. http://www.olapcouncil.org/research/glossaryly.htm

[36] Pentaho Data Integrator[OL]. Available: http://www.pentaho.com/

[37] ISO 8601:2004, "Data elements and interchange formats - Information interchange - Representation of dates and times"[S]. Available: http://www.iso.org/

[38] John Poole, Dan Chang, Douglas Tolbert, and David Mellor (2002). The Common Warehouse Metamodel: An Introduction to the standard for Data Warehouse Integration[M]. OMG Press (John Wiley & Sons), 2002 ISBN 0-471-20052-2.

# 致 谢

本论文的撰写即将结束，在此我要向所有在我两年半的研究生期间给予过我关心和帮助的老师、同学和朋友表示真挚的感谢。

首先，我要感谢我的导师饶若楠副教授，两年半的时间虽然不长，但他给予我的教诲却是能让我受用一生的。饶老师给予了我许多参加大型项目研究和开发的机会，并且在项目工作中对我细心指导。在撰写论文的过程中，饶老师从选题、立意、写作及系统开发等各方面都给予了我许多宝贵的建议，并要求我精益求精，培养了我严谨踏实的学习工作作风，使得我的论文能够顺利完成。同时在生活上，饶老师给予了我无微不至的关心和帮助。

感谢鼓励过我、帮助过我的所有同学，尤其是分布式计算实验室的兄弟们，在这两年半的学习生活中，我们互相探讨、取长补短，共同取得进步。在此我希望你们能够在接下来的学习研究中取得更好的成绩。

感谢我的家人，你们一直在学习和生活上支持我、关怀我、鼓励我，希望你们能够永远健康快乐。

# 攻读硕士学位期间已发表的学术论文

[1] H. Ge, R.N. Rao. A business metadata modeling approach for data integration based on SDMX[C]. 2014 International Conference on Network Security and Communication Engineering(NSCE2014), Hong Kong, China.