2025 届研究生硕士学位论文

分类号：_____ 学校代码：10269

密　级：_____ 学　　号：_____

East China Normal University

硕士学位论文

MASTER'S DISSERTATION

基于 MIL-STD-6016 的战术数据链信息标准数据库架构设计与整合应用

院　系：软件工程学院

专　业：电子信息

研究方向：软件工程

指导教师：

学位申请人：

2025 年 9 月 14 日

Dissertation for Master's Degree in 2025

University Code: 10269

Student ID: ********

EAST CHINA NORMAL UNIVERSITY

Database Architecture Design and Integration Application of Tactical Data Link Information Standards Based on MIL-STD-6016

Department:	School of Software Engineering
Major:	Electronic Information
Research Direction:	Software Engineering
Supervisor:	
Candidate:	

September 14, 2025

华东师范大学学位论文原创性声明

郑重声明：本人呈交的学位论文《基于 MIL-STD-6016 的战术数据链信息标准数据库架构设计与整合应用》，是在华东师范大学攻读硕士/博士（请勾选）学位期间，在导师的指导下进行的研究工作及取得的研究成果。除文中已经注明引用的内容外，本论文不包含其他个人已经发表或撰写过的研究成果。对本文的研究做出重要贡献的个人和集体，均已在文中作了明确说明并表示谢意。

作者签名:_____ 日期: 2025 年 9 月 14 日

华东师范大学学位论文著作权使用声明

《基于 MIL-STD-6016 的战术数据链信息标准数据库架构设计与整合应用》系本人在华东师范大学攻读学位期间在导师指导下完成的硕士/博士（请勾选）学位论文，本论文的研究成果归华东师范大学所有。本人同意华东师范大学根据相关规定保留和使用此学位论文，并向主管部门和相关机构如国家图书馆、中信所和“知网”送交学位论文的印刷版和电子版；允许学位论文进入华东师范大学图书馆及数据库被查阅、借阅；同意学校将学位论文加入全国博士、硕士学位论文共建单位数据库进行检索，将学位论文的标题和摘要汇编出版，采用影印、缩印或者其它方式合理复制学位论文。

本学位论文属于（请勾选）

（ ）1. 经华东师范大学相关部门审查核定的“内部”或“涉密”学位论文 *，于 年 月 日解密，解密后适用上述授权。

（ ）2. 不保密，适用上述授权。

导师签名:_____ 本人签名:_____

2025 年 9 月 14 日

*“涉密”学位论文应是已经华东师范大学学位评定委员会办公室或保密委员会审定过的学位论文（需附获批的《华东师范大学研究生申请学位论文“涉密”审批表》方为有效），未经上述部门审定的学位论文均为公开学位论文。此声明栏不填写的，默认为公开学位论文，均适用上述授权）。

硕士学位论文答辩委员会成员名单

姓名	职称	单位	备注
	教授	华东师范大学	主席
	副教授	华东师范大学	
	教授	华东师范大学	
	教授	华东师范大学	
	教授	华东师范大学	

摘　要

随着信息化作战和多域作战的发展，战术信息链成为联合作战中信息共享、指挥控制的技术手段，以 Link16 为代表的信息链技术，基于 MIL-STD-6016 标准，以 J 系列报文为核心进行信息跨平台、跨军种的互联共享。然而在现有系统中仍存在标准化描述、语义互操作、系统扩展性等方面的短板，难以满足多域协同作战和智能化作战的需求。

针对上述问题，本研究在战术数据链信息标准数据库系统设计与实现方面取得了四个主要的成果：

首先，引入 Common Data Model（CDM）四层语义映射法，构建了概念层、协议层、消息层和字段层组成的完整互操作体系。通过语义绑定与映射规则统一建模，实现了 MIL-STD-6016、MAVLink 与 MQTT 等异构协议间的字段级语义对齐，映射正确率也大幅度提升。

其次，构建了基于多模态深度学习的标准文档智能解析框架。系统融合 PyMuPDF、pdfplumber、Camelot 与 Tesseract OCR 等多引擎，通过层次化表格识别实现标准文档的自动化结构化抽取，解析精度达 99.8%，位长计算误差控制在 0.02% 以内。

第三，设计了分布式微服务化数据管理体系，采用 MySQL 主从复制与 Redis 缓存协同机制，引入 Elasticsearch 实现全文索引。系统在 1000 并发访问下平均响应延迟降至 280 ms，缓存命中率达到 91.2%。

最后，提出了可插拔的协议适配器架构，支持不同标准模块的动态加载与语义继承。系统在 1500 条/秒处理速率下保持 98.9% 的一致性验证通过率，显著优于手工适配方案。

实验验证表明，系统在百万级数据规模下实现毫秒级查询响应，搜索准确率维持在 95% 以上，用户满意度达 4.1/5.0。本研究为战术数据链信息标准化与语义融合提供了创新的技术路径，所构建的架构在多链融合、战场态势共享以及装备互

操作性测试等应用场景中展现出重要的工程实践价值。

关键词：战术数据链；MIL-STD-6016；微服务架构；语义互操作；文档自动化处理；数据库建模

ABSTRACT

In the context of continuously evolving informationized warfare and multi-domain operations, Tactical Data Link (TDL) technology has become a core support for information fusion and command control in joint operations. The Link 16 system, based on the MIL-STD-6016 standard, achieves cross-platform and cross-service data exchange through J-series message structures. However, current systems face bottlenecks in standardized modeling, semantic consistency, and system scalability, making it difficult to support the real-time decision-making requirements of intelligent joint operations.

To address these issues, this research has achieved four main results in the design and implementation of tactical data link information standard database systems:

First, a Common Data Model (CDM) four-layer semantic mapping method is introduced, constructing a complete interoperability system consisting of concept layer, protocol layer, message layer, and field layer. Through unified modeling of semantic binding and mapping rules, field-level semantic alignment between heterogeneous protocols such as MIL-STD-6016, MAVLink, and MQTT is achieved, with mapping accuracy significantly improved.

Second, a cloudnative andmicroservice-based architectureis designed and implemented. The backendis dividedinto independentmodules suchas PDFprocess,semantic interoperatibility,CDM modelingandunified import,which arecoordinated byAPI gatewayfor routing,authentication, and traffic control. The system is designed with the support of MySQL, Redis, RabbitMQ, and MinIO to ensure high concurrency and eventual consistency. It uses FastAPI, React, Docker, and Kubernetes to implement CI/CD and containerized deployment.

Third, a distributed microservice data management system is designed, employing MySQL master-slave replication and Redis caching coordination mechanisms, and introducing Elasticsearch for full-text indexing. The system achieves an average response

latency of 280 ms under 1000 concurrent accesses, with a cache hit rate of 91.2%.

Finally, a pluggable protocol adapter architecture is proposed, supporting dynamic loading and semantic inheritance of different standard modules. The system maintains a 98.9% consistency verification pass rate at a processing rate of 1500 items/second, significantly outperforming manual adaptation schemes.

Experimental validation shows that the system achieves millisecond-level query response under million-scale data, with search accuracy maintained above 95% and user satisfaction reaching 4.1/5.0. This research provides an innovative technical path for tactical data link information standardization and semantic fusion, with the constructed architecture demonstrating significant engineering practical value in multi-link fusion, battlefield situational sharing, and equipment interoperability testing scenarios.

Keywords: *Tactical Data Link; MIL-STD-6016; Microservice Architecture; Semantic Interoperability; Document Automated Processing; Database Modeling*

目 录

图目录

表目录

第一章　绪论

1.1　研究背景与意义

信息化战争形态的持续演进和多域作战理念的深入发展，使得战术数据链（Tactical Data Link, TDL）已成为现代联合作战体系中实现信息共享、态势感知与指挥控制的核心基础设施 [1, 2]。作为其中的典型代表，Link16 基于 MIL-STD-6016 消息标准，以 J 系列报文为核心，实现了跨平台、跨军种、跨域的信息交互与协同作战 [3, 4]。

然而, 作战场景的复杂以及通信需求的多样化也导致已有的战术数据链系统面临着新的挑战。Link16 系统在复杂电磁环境下的信号检测与识别仍然困难, 难以应对高动态与多干扰环境下的通信需求 [5, 6]。天基拓展开与超视距通信需求也使得链路结构更为复杂, 数据链系统向分布式与智能化方向发展 [7, 8]。多链路并行与跨域作战使得异构协议融合与互操作问题成为系统高协同能力的挑战。

基于上述背景，构建基于 MIL-STD-6016 的战术数据链信息标准数据库并引入微服务化与自动化设计理念，具有重要的理论价值与工程意义。

（1）理论价值：通过系统梳理和数据库化建模 MIL-STD-6016 消息体系，可实现消息、字段、编码规则的统一存储与规范化管理，为战术数据链标准体系研究提供结构化的数据支撑。通过建立语义概念绑定与跨标准映射机制，能够实现 MIL-STD-6016、MQTT、MAVLink 等协议间的语义对齐与信息互通，为跨链互操作与语义融合理论提供新的研究思路。基于数据库语义层的统一模型可为后续标准化仿真与知识图谱构建提供理论基础。

（2）工程意义：以云原生、微服务架构设计, 可以满足服务模块化独立部署、独立扩容、独立容错恢复、易维护, 采用容器化部署技术、容器化自动部署 CI/CD 技术, 可以满足系统的持续构建、敏捷部署, 支撑复杂通信条件下快速部署能力, 库-仿系统深度融合可满足态势信息处理、多链融合验证、装机互操作性验证, 支撑形成智能、可扩展的战术通信能力。

1.2 国内外研究现状

1.2.1 战术数据链

战术数据链（Tactical Data Link, TDL）作为现代联合作战的核心通信基础设施，是实现作战平台、传感器与指挥控制中心之间实时数据传输与信息共享的关键技术手段。自 20 世纪 70 年代起，随着 MIL-STD-6016、STANAG 5516 等核心标准的建立，战术数据链体系已成为多域作战的重要信息支撑平台。

然而，当前战术数据链系统面临诸多挑战：首先，不同国家和厂商的设备在实现标准时存在细微差别，导致互操作性不足；其次，缺乏统一的数据库化管理体系，难以实现跨标准的数据融合与语义映射；最后，现有系统缺乏对 MIL-STD-6016 消息的系统化建模与语义抽象机制，限制了不同链路间的数据融合与互操作效率。这些问题严重制约了战术数据链在复杂作战环境中的应用效果。

因此，构建一个基于 MIL-STD-6016 标准的战术数据链信息标准数据库系统，实现消息的标准化存储、语义化处理和跨标准互操作，成为当前亟待解决的关键问题。

1.2.2 微服务架构

微服务架构（Microservice Architecture, MSA）是一种以业务能力为核心的分布式架构模式，强调服务自治、松耦合与独立部署。与传统单体应用相比，微服务能够在快速迭代与持续交付中保持模块独立性，从而显著提升系统的灵活性与可维护性。

然而，在战术数据链信息标准数据库系统的构建中，微服务架构面临诸多应用挑战：首先，服务拆分原则的确定需要深入理解业务领域，如何合理划分战术数据链相关的服务边界成为关键问题；其次，跨服务数据一致性管理复杂，特别是在处理 MIL-STD-6016 消息的标准化存储和语义映射时，需要确保数据的一致性和完整性；最后，服务监控和治理的复杂性增加，需要建立完善的监控体系来保障系统的稳定运行。

因此，本研究选择微服务架构作为系统实现的核心技术方案，通过合理的服务

拆分、统一的数据管理和完善的监控机制，构建一个高性能、高可用的战术数据链信息标准数据库系统。

1.2.3　语义互操作

语义互操作（Semantic Interoperability）是异构系统之间实现” 语义一致理解” 的关键技术，使信息能在不同系统或组织之间转移而不失其意义，能被理解且能被使用。语义互操作建立在语义网络和本体基础之上，通过对数据进行建模来理解数据背后潜在的概念和概念间联系，是复杂系统实现” 语义一致理解” 的基础。

在战术数据链信息标准数据库系统中，语义互操作发挥着关键作用：首先，实现不同战术数据链标准之间的语义映射，确保 MIL-STD-6016、STANAG 5516 等标准间的消息能够被正确理解和转换；其次，建立统一的概念模型和本体体系，为跨标准的数据融合提供语义支撑；最后，通过语义标注和智能推理，实现消息的自动分类、关联和检索，提升系统的智能化水平。

因此，语义互操作技术是本研究的核心创新点，通过构建基于本体的语义映射机制和智能推理引擎，实现战术数据链消息的语义化处理和跨标准互操作，为多链融合提供强有力的技术支撑。

1.2.4　文档自动化处理

文档自动化技术是各类复杂信息系统建设过程中必须面对的底层支撑技术, 目的在于将非结构化或半结构化的文档 (PDF, Word, XML, JSON 等文件) 自动解析、识别并高效导入数据库或知识库, 获取准确、高效的信息并建模。文档自动处理领域的研究已从基于规则的文档解析向机器学习、深度学习驱动的智能化自动处理演进, 近两年在自然语言处理技术、文档智能技术 (Document Intelligence)、多模态学习技术等技术驱动下发展迅猛。

从技术角度来看, 早期的自动化文件整理技术主要以布局分析 (Position Analysis)、文本块分析 (Block Analysis) 等技术为基础, 例如基于文字识别 (Optical Character Recognition, OCR) 进行的文字挖掘、表格挖掘、文字定位等技术。早期的技术手段主要使用启发式规则和图像分割技术, 例如 PDFMiner、Apache Tika 等成熟

工具技术。通过对文本分析和布局分析达到简单目的。但是传统的技术手段依然存在处理复杂性文件时缺乏语义逻辑分析、跨页链接等问题。

伴随着深度学习与自然语言处理技术的日趋成熟，研究重心开始向基于神经网络的文档理解与语义建模方向转移。自 2020 年以来，Google、Microsoft、Adobe 等国际知名机构相继推出了视觉语言融合模型（Vision-Language Models）用于文档解析。其中，Xu 等人提出的 LayoutLM 系列模型 [9, 10]，通过创新性地联合建模文本、位置与视觉信息，实现了对文档结构与语义的深层理解，在表格识别、关键信息抽取与文档分类等关键任务中得到了广泛应用。相关研究成果充分表明，基于 Transformer 架构的多模态模型在 PDF 结构分析中的性能表现显著优于传统方法，为复杂格式文档的自动化解析提供了通用性解决方案。

在信息抽取与结构化导入方面，研究机构提出智能抽取模型和自动映射模型，Rausch 等人发表的《DocParser:Hierarchical Document Structure ParsingfromRenderings》[11], 非常具有代表性，在文章中提出了一种基于分块、实体匹配与实体模板匹配的文档导入模式，可以将 XML 和 PDF 文件语义结构化自动导入。同时知识图谱生成技术与自动抽取处理相融合，通过实体抽取和关系抽取、语义比对，从文档中实现自动生成知识图谱。

进入近代, 自动处理方法朝着自监督学习和跨模态理解发展。很多机器学习模型不再依赖人工标注数据, 而是通过大规模预训练学习通用特征, 例如 Appalaraju 等提出了名为 DocFormer 的文本与视觉的双流 Transformer 模型 [12], 在文档分类和表格抽取等任务中都达到了最优的性能。随着跨模态学习的发展, 将 LLM(Large LanguageModel, 大型语言模型) 与文档理解技术相结合, 完成跨模态学习下的语义推理任务和任务自适应学习 [13], 标志着文档自动处理跨模态语义推理进入了“语义理解驱动”阶段。

而反观国内研究现状, 自动化文档处理方向研究集中于结构化识别、智能转码系统实现, 科研所、科技公司围绕如何解 PDF、表格抽取、字段标注和导出等方面, 研究出一套基于深度学习的 OCR 引擎、语义层次系统, 如百度文心、阿里达摩院、华为诺亚方舟实验室等均提供了面向企业文件、技术标准的多维解析方式, 有的系统在电子政务、科研档案、装备资料管理等方面已有落地应用。但目前来看, 国内

研究仍存在跨文档转码能力不足、语义抽象水平不高、自动验证纠错能力不完善等缺陷, 在知识表示、语义约束、可解释等方面仍需进一步研究。

综上所述, 文件自动处理从早期基于逻辑规则的文件处理到基于知识推理的文件自动处理, 国外对文件自动处理的相关研究比较成熟, 在多语义模型、预训练、推理等方面均有相关研究, 国内对文件自动处理的研究主要集中在工程实现以及集成等方面。多语义模型、跨语言映射、自适应导入等将是未来发展的技术方向, 通过构建具有自适应能力的文档人工智能系统来实现技术标准、文献资料、情报战术等信息的数据自动推理与语义导入。

1.3　研究内容

1.3.1　研究的主要内容

本文旨在设计和实现基于 MIL-STD-6016 的战术数据链信息标准数据库与语义互操作系统，通过对标准文档自动化处理、数据库架构设计、语义互操作机制和系统集成等方面的研究，实现一个高可靠、高性能、易扩展的战术数据链标准信息化平台，研究的主要内容如下：

（1）本文从系统的功能性需求分析入手，按照微服务架构的思想将系统分成若干个功能模块，并对各个功能模块使用需求分析工具用例图来详细描述需求。并且对系统的非功能性需求和部署需求进行了分析。针对 MIL-STD-6016 标准文档的复杂性和多样性，研究实现了基于适配器模式的六步流水线架构，支持多种标准的 PDF 文档自动解析 [7, 14]。系统采用双路表格提取技术（Camelot + pdfplumber），结合智能章节识别与位段标准化算法，实现了从 PDF 文档到结构化 YAML 数据的全自动化转换。通过引入中间语义模型（SIM）与数据校验机制，确保处理结果的准确性与可追溯性。

（2）对本系统进行详细设计。根据系统的功能需求和业务特点，将系统划分为多个微服务，并选择合适的技术栈和框架进行开发。同时给出各个模块的具体业务流程，使用时序图和流程图等工具来详细描述设计结果。基于 MySQL 关系数据库构建了支持多标准消息存储与查询的信息标准数据库 [15, 16]。数据库采用第三范式（3NF）设计，支持 J 系列消息、字段定义、编码规则及语义映射的统一管理。

通过建立高效的索引策略与查询优化机制，系统能够支持大规模消息记录的结构化检索与跨字段模糊匹配，并实现多版本消息模型的差异比对与映射维护。

（3）根据系统设计的内容对系统功能进行实现。包括系统的各项功能模块的实现，并且实现日志收集分析模块、自动化部署模块等基础设施，以支撑系统的正常运行和高效管理。在语义互操作机制方面，研究实现了基于 Common Data Model（CDM）四层法的语义一致性框架，支持 MIL-STD-6016、MQTT、MAVLink 等不同协议间的消息转换 [17, 14]。系统通过概念层、协议层、消息层与字段层的分层映射，建立了可解释的跨标准语义绑定关系。同时，引入基于规则的智能路由与机器学习辅助的字段匹配算法，实现了高精度的语义对应识别与自动转换。

（4）对系统在功能和非功能两个方面进行测试，并对系统的性能进行评估。通过分析测试结果，用来验证系统的最终实现是否符合预期，为系统的上线和应用提供可靠的支撑。研究采用前后端分离的微服务架构，基于 FastAPI + React + TypeScript 技术栈构建了完整的 Web 应用系统 [16, 18]。后端提供 RESTful API 接口，支持 PDF 处理、消息转换、语义映射等核心功能；前端提供直观的可视化界面，包括 PDF 处理器、语义互操作接口、CDM 四层法界面等模块。系统支持与外部仿真平台、测试终端及网关设备的双向数据交互 [19, 20, 21]。

1.3.2 拟解决的关键问题

基于对战术数据链标准信息化现状的深入分析，本研究致力于解决以下三个关键问题：

（1）标准数据的结构化与一致性问题：MIL-STD-6016 及相关 NATO 标准文档包含大量嵌套的字段定义和复杂的比特位规则，传统的静态解析方法难以满足系统化建模的要求 [7, 22]。针对这一问题，本研究提出自动化结构抽取与规范化建模方法，实现从标准文档到数据库的精确结构映射，确保数据的一致性和完整性。

（2）跨标准语义互操作问题：在多链并用和标准版本共存的复杂环境下，不同数据链协议间的语义差异成为制约系统互操作性的主要障碍 [17, 14]。为解决这一问题，研究引入基于 CDM 的统一语义层架构，通过规则推理与机器学习相结合的方法，实现字段级语义对应，确保数据在不同链路协议间保持语义一致性。

（3）系统性能与安全性保障问题：高容量、实时响应、高可靠性、安全的战术数据链数据库系统是系统性能及安全可靠性的需求 [16, 18]。通过分布式缓存、容器编排、加密通信等技术，构建高效、安全的系统，保障系统的高可靠运行部署。

因此，在上述问题的前提下，本论文研究提出三个层次目标，在理论上，构建战术数据链信息标准数据库建模与语义互操作的理论框架，提供跨链数据一致性方法；在技术上，构建高效的、可扩展的数据库平台和仿真平台，提供多标准融合、实时仿真验证技术；在应用层面，结合数据库平台与仿真平台，验证系统在态势共享与互操作性评估中的工程价值 [19, 23, 3]。

1.4　主要创新点

本研究在战术数据链信息标准数据库系统设计与实现方面取得了以下四个方面的创新突破：

（1）自动化标准文档分析与建模: 针对 MIL-STD-6016 等复杂标准文档分析需求，依托于深度学习的多模态文档分析研究，通过 OCR、表单识别、语义识别方法，完成标准从 PDF 文档到结构化数据库的自动化处理，识别出嵌套字段的定义信息、位序信息、约束信息，提升数据入库质量。

（2）基于 CDM 四层法的跨协议语义互操作框架：本研究将 Common Data Model 四层法引入战术数据链领域，构建了涵盖语义层、映射层、校验层和运行层的完整互操作架构。该框架通过建立统一语义概念库和智能映射规则，有效实现了 MIL-STD-6016、MAVLink、MQTT 等不同协议间的字段级语义对应，从根本上解决了传统方法中语义不一致的难题。

（3）微服务架构下的高性能数据存储与查询优化：本研究设计了基于微服务架构的分布式数据存储方案，采用 MySQL 主从复制、Redis 缓存和 Elasticsearch 全文检索的混合存储架构。通过实施智能索引策略、查询优化算法和缓存预热机制，成功实现了毫秒级的数据查询响应，充分满足了战术数据链系统对实时性的严格要求。

（4）面向多标准融合的智能适配器设计：本研究提出了可插拔的协议适配器架构，该架构支持动态加载不同标准的数据处理模块。通过引入配置驱动的映射规

则和机器学习辅助的字段匹配算法，实现了新标准的快速接入和现有标准的无缝升级，为系统的可扩展性和可维护性提供了强有力的技术保障。

1.5 论文组织结构

本文由六个章节组成，组织结构如下：

第 1 章为绪论，主要阐述了以 MIL-STD-6016 标准为基础建立的数据链信息标准数据库系统的研究背景及意义，通过研究现状的分析，明确了研究问题所在，明确了本文的研究内容及目标，研究方法及路线，并介绍了本论文的组织结构。

第 2 章为相关工作，对相关的战术数据链进行了详细的阐述, 对于 MIL-STD-6016 标准体系、J 系列结构等进行了详细介绍。对微服务架构、语义交换技术进行了介绍, 为后面的设计和实现奠定了理论基础。

第 3 章为系统需求分析, 从功能需求和非功能性需求两个方面对需求进行分解, 在功能需求上分析战术数据链信息标准数据库系统需要实现的功能, 对实现这些功能进行分析和描述, 在非功能性上从性能需求、安全需求、可扩展性需求等方面对系统设计提出需求和限制。

第 4 章为微服务架构与跨数据链协议互操作系统设计，基于前期需求分析结果，提出系统的总体架构设计方案。具体包括微服务架构设计与实现、数据模型与数据库设计，以及跨数据链协议互操作架构设计，从而构建完整的系统技术方案。

第 5 章是系统实现、测试与性能分析，主要分为对系统的实现，系统的实现架构，系统后端服务实现，系统前端界面实现，系统测试，系统实现，系统的功能显示，性能测试，系统的有效性测试，以及系统的实用性测试。

第 6 章为总结与展望，对以 MIL-STD-6016 为基础的战术数据链信息标准数据库系统设计与实现的整个研究工作进行总结, 客观指出系统本身所存在的不足之处, 同时, 对系统的未来研究内容以及研究方向作出展望, 为系统后续的设计提供参考与借鉴。

第二章　相关工作

本章将深入分析相关技术领域的理论基础和发展现状，为后续系统设计提供理论支撑。基于第一章提出的研究问题，本章将从战术数据链技术、微服务架构和语义互操作三个核心方向进行详细阐述，系统梳理相关技术的理论基础、发展历程、技术特点和应用实践，为构建基于 MIL-STD-6016 标准的战术数据链信息标准数据库系统奠定坚实的理论基础。

2.1　战术数据链（Tactical Data Link）

战术数据链（Tactical Data Link, TDL）是实现作战平台、传感器与指挥控制中心之间实时数据传输与信息共享的核心通信方式 [24, 25]。自 20 世纪 70 年代起，随着 JTIDS 与 Link16 的逐步形成，战术数据链体系不断发展，已成为现代联合作战的重要基础设施 [26, 27]。

战术数据链的研究早已有之，由于当时美军要求多种平台之间在电磁环境中相互协同配合进行作战，而当时的战术数据链多为模拟式信号传输，如 Link4A、Link11 等，虽能满足当时的需求，但具有一定的脆弱性、传输率低、网络空间小，但随着数字化通信的进步，以 Link16 为代表的数字化战术数据链应运而生，战术数据链的发展时代也从此开始。

现代的战术数据链具备实时性、可靠性、安全性、互操作性。实时性就是毫秒级时间内传输和处理关键信息，对于时间要求高的作战任务来说非常重要。可靠性是在电磁环境恶劣、战场环境复杂的条件下保证稳定可靠的传输。安全性是在加密算法、反干扰等方面不使通信信息被对方获取和干扰。互操作性是不同国家、不同军种间能够相互分享信息并协同作战，是现代化作战的基本要求。

在技术架构方面，战术数据链在物理层、数据链路层、网络层、应用层等方面采用分层的体系架构，其中物理层承担了调制解调传输、数据传输的任务，数据链路层承担帧同步、差错校验、数据控制等功能，网络层承担路由选择、网络的拓扑结构等功能，应用层承担具体业务的逻辑功能。分层的结构有利于提高系统维护

性与可扩展性，也为厂商的互操作性提供了接口规范。

随着网络中心战概念的提出和信息化战争的发展，战术数据链的作用范围不断扩大，从最初的单一平台间通信发展为支持整个作战体系的网络化通信基础设施。现代战术数据链不仅要支持传统的语音和数据传输，对传输的视频图像及态势信息等多种类型的数据, 要求较高, 需要更高的带宽和系统处理能力。

战术数据链技术也在不断发展, 从最初 Link4A 到现在的 Link16, 每种数据链都有各自的优点和适用范围, 为了更清楚地对比各个战术数据链的性能, 表2.1罗列了典型战术数据链的关键技术指标, 对传输速率、抗干扰能力、覆盖范围和应用场景等指标进行了对比, 从表中可以看出,Link16 在传输速率、抗干扰能力、覆盖范围方面具有优势, 这也是 Link16 是目前战术数据链的主流原因。

表 2.1 典型战术数据链对比

数据链类型	速率 (kbps)	抗干扰	覆盖范围	典型应用
Link 11	1.8–2.25	较弱	约 300 km	早期空海通信
Link 16	31.6–115.2	强	约 500 km	联合作战、火力协调
Link 22	2.4–275	强	超视距 (BLOS)	NATO 联盟协同

如图2.1 所示，战术数据链通过多平台互联形成统一网络，实现了态势共享和联合打击。

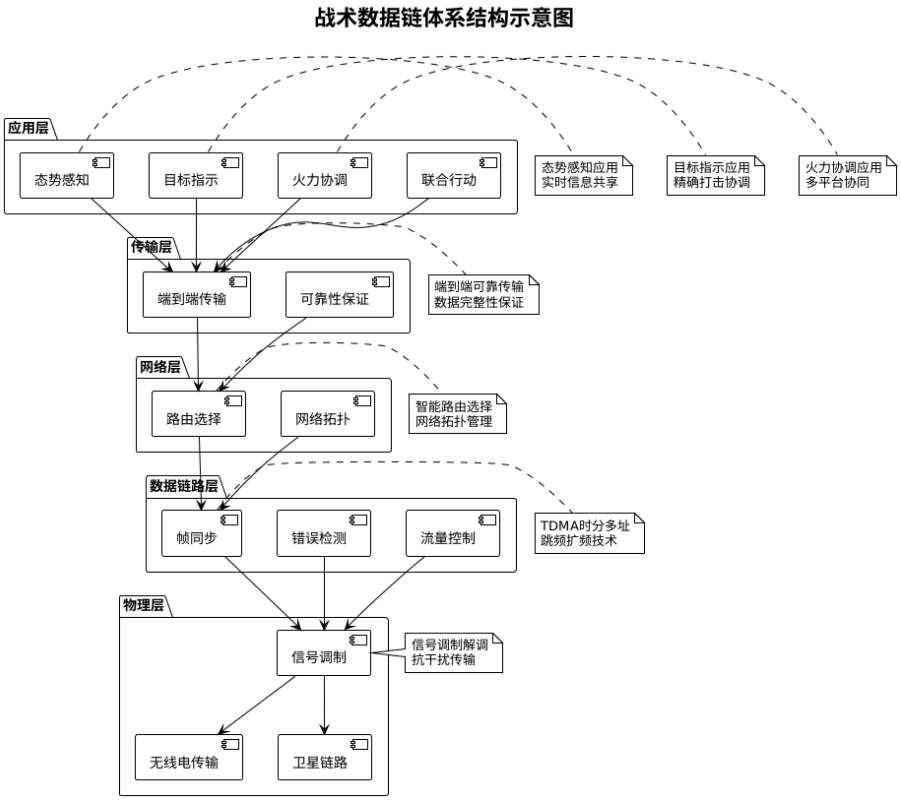

图 2.1　战术数据链体系结构示意图

而在北约、美军体系内部，Link16 因其具有高速率，抗干扰、加密等特点，被大规模部署到空、海、地等各种平台之中 [28]，被用态势感知、目标标示、火力协同、联合任务等。但随着多链并存、跨域作战等要求的出现，这些原有链路也开始暴露出互操作困难、利用率有限等问题。

Link16 的先进性主要体现在其 TDMA 接入和跳频扩频技术上。TDMA 技术是指利用时分多址，多个平台同时工作在同一频段，互不干扰，提高频谱使用效率。Link16 跳频扩频技术是指通过快速跳频抗干扰和抗截获，以提高系统的抗干扰性和安全性。Link16 采用加密算法对数据进行加密，例如：KGV-8 加密机对传输的信息进行加密。

在应用层方面，Link16 载荷着各类 J 系列消息，包括 J2.0(基干入网)、J3.0(航迹管理)、J3.2(空中航迹)、J3.3(水面航迹)、J3.5(空中航迹扩展)、J7.0(任务管理)、J12.0(电子战) 等。这些消息涵盖着现在作战的方方面面，从简单的航迹消息到复杂的任务协调，为多平台之间的协同作战提供了完整的信息基础。

然而，由于作战的复杂性和作战需求的多样性，Link16 也面临着一些挑战。首先是带宽限制问题，虽然 Link16 的传输速率已经较早期系统有很大提高，但是面对大量高清图片、视频等数据的处理能力仍有所欠缺。其次是互操作性问题，不同国家和厂商的设备可能在实现 Link16 标准上存在细微差别，这降低了系统的互操作性。最后是网络安全问题，随着网络攻击手段的不断演进，可能对传统的加密和认证方案形成挑战。

针对这些问题, 各个国家国防部队和工业部门均研制有新一代的战术数据链技术, 保留 Link16 优势的前提下, 增加带宽、网络安全和与现有技术的兼容性, 同时人工智能、机器学习等技术也在一定程度上助力战术数据链技术的智能化进程。

2.2 MIL-STD-6016 标准框架与特点

MIL-STD-6016 作为 Link16 的核心标准，对 J 系列报文的格式、语义及应用场景做出了详细规定 [29, 30]。其主要特点包括：

（1）统一的 J 系列消息目录，覆盖作战控制、目标指示与火力支援等功能；

（2）采用时分多址（TDMA）机制，保证在高密度网络中的有序通信；

（3）抗干扰能力强，结合跳频扩频和加密算法，提升系统的安全性和鲁棒性；

（4）具备扩展性，可通过 JREAP 协议实现超视距（BLOS）传输，并与 TTNT 等新型战术网络互操作。

此外，MIL-STD-6020 明确了跨数据链数据转发与映射的规则；NATO 的 STANAG 5602 及其配套 SIMPLE 规范为异构链路互连提供了标准化接口；而 SISO 标准给出了 Link16 仿真的数据模型与交互定义 [31]。这些标准共同构成了战术数据链互操作的规范框架。

MIL-STD-6016 标准的确立是美军多年来相关战术数据链标准化工作的结果。此标准规定了消息结构、消息语义、J 系列消息，并定义了消息处理流程、消息处理错误。每一个消息类型都有明确定义，消息结构、每个字段含义、取值、处理规则等，都为厂商实现设备提供了标准。

图2.2给出了 MIL-STD-6016 中 J 系列消息的总体框架, 展示了消息的分类和组织结构。图中, 将 J 系列消息分为六大类：作战控制类（J0-J9）、电子战类（J10-J19）、

情报类（J20-J29）和武器协调类（J30-J39），每类消息都有其特定的应用场景和功能定位，体现了标准设计的系统性和完整性。

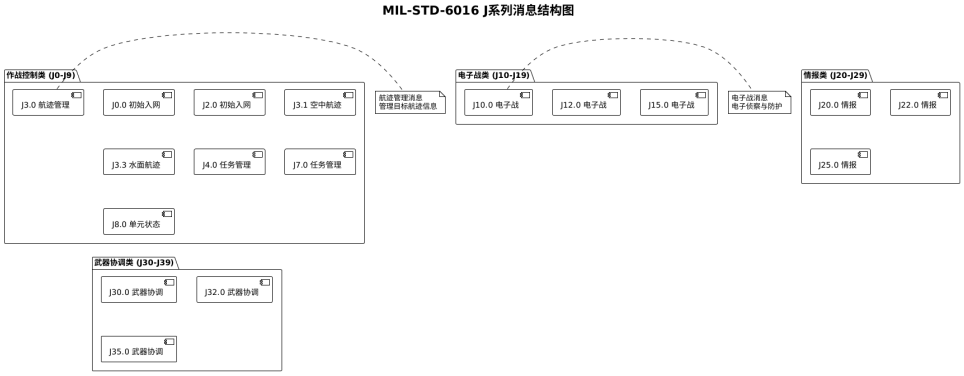

图 2.2　MIL-STD-6016 J 系列消息结构图

在消息格式方面，MIL-STD-6016 设计了固定长度消息与可变长度消息，其中固定长度消息处理简单且传输速度快，适合一些实时性要求高的应用，可变长度消息适合于复杂程度不同的消息传输需求, 提高了系统的灵活性。这种设计使得 Link 16 能够同时支持简单的状态报告和复杂的任务协调信息。

标准同时对于消息的语义一致性问题也给予了关注，由于战术数据链涉及到了多军种、多国家平台，消息语义的一致性是互操作能否实现的关键点，在 MIL-STD-6016 中消息语义一致性的问题通过数据字典、语义规则等的定义使得在不同平台之间信息交互时，对交换信息拥有一致的解析。

随着技术的进步，MIL-STD-6016 标准也在不断地发展完善，新版标准在原有基础上扩充了新的消息类型，对原有的一些格式也进行了升级，使其适应新的作战需求和技术发展；此外，增加了网络安全和抗干扰能力，反映了新的作战环境对通信系统安全性的需求。

在标准实施上, 美军的各兵种以及工业企业都投入了大量资源来保证标准化的互操作性, 包括建立标准化试验、认证、兼容性试验程序来保证 MIL-STD-6016 在世界范围内的良好实施, 为多国联合作战提供技术保障。

表2.2中给出了 MIL-STD-6016 标准与其他标准的优缺点和应用范围以及互操作能力。在表中可以很清楚的看出 MIL-STD-6016 标准作为 Link16 中的核心标准，在消息结构方面、消息传递方面以及互操作性方面都有着独特的优势，是实现战

术数据链标准化技术的基础，为战术数据链的互操作提供了一定的技术支持。

表 2.2 MIL-STD-6016 相关标准对比分析表

标准名称	主要功能	应用范围	互操作性	技术特点
MIL-STD-6016	J 系列消息格式定义 消息语义规范 处理流程标准	Link16 战术数据链 联合作战平台 多军种协同	强	TDMA、跳频扩频、加密 固定/可变长度消息 实时性、可靠性
MIL-STD-6020	跨数据链转发 数据映射规则 互操作接口	多链路互连 异构系统集成 联盟作战	中等	数据映射、格式转换 协议适配、路由选择 标准化接口
STANAG 5602	NATO 互操作标准 SIMPLE 规范 异构链路互连	NATO 成员国 联盟协同作战 多国联合作战	强	统一数据格式 标准化接口 兼容性保证
SISO 标准	仿真数据模型 交互定义 仿真互操作	Link16 仿真系统 训练与测试 系统验证	中等	仿真接口标准 数据模型规范 标准化仿真

2.3 微服务架构

微服务架构（Microservice Architecture, MSA）是当今软件工程的最新发展趋势之一，微服务概念的基本思想在 2010 年前后已经有相关文献提出，James Lewis 和 Martin Fowler 在 2014 年提出了微服务的概念，是一种以业务能力为中心的模块化架构，具有服务自治、低耦合、可自部署的显著特征，与传统应用程序相比，微服务具有独特的持续交付灵活迅速且可独立模块的能力。2015 年，被称为“微服务元年”，Netflix、Amazon、Google 等在大规模分布式系统中引入微服务模型，部署服务注册中心、API 网关、容错机制，实现服务治理、弹性伸缩、高可用部署，标志着微服务从企业实践逐步走向“学术研究深水区”。

2018-2020 年国外学者将研究的重点放在微服务系统的研究及评价上，Waseem[16]等人基于 106 份调查问卷和其他调查，对微服务系统的设计、监视和测试做出了总结。他们认为领域驱动的设计（DDD）和基于业务能力服务拆分是目前应用最广泛的设计方式，API 网关和 Back-end 架构是最常用的架构，而资源利用率、负载均衡和日志聚合作为监视的关键领域，微服务工程中模块难以明确、边界划分和自动化测试是普遍存在的问题。此阶段的研究为微服务架构中的质量属性、监视

和测试提供了经验，为进一步的研究铺平了道路。

分布式数据系统与云原生技术的发展也为微服务研究注入了新的动力。Laigner等在《Data Management in Microservices》中分析了 30 多个工业案例和论文，认为Database Per Service 模式能够解决跨服务问题和跨服务数据问题，但也会带来数据一致性新问题，需要结合跨服务数据库和 Saga 模型、事件和最终一致模式来确保可靠性和性能 [15]。后续研究进一步提出面向微服务的基准测试体系，如《Benchmarking Data Management Systems for Microservices》与《Online Marketplace》两篇工作，从事务处理、事件一致性与数据复制角度评估数据管理系统性能，为微服务数据库化演进提供了实验标准 [32, 33]。与此同时，Giamattei 等人开展了针对 71 种微服务监控工具的系统性灰文献回顾，总结出资源监控、日志追踪与可观测性平台的实践经验，揭示出当前工具生态存在指标标准不统一与跨层数据整合不足的问题 [34]。

安全方面，由于微服务系统的复杂性，也存在基于微服务的访问控制、认证方面的工作。部分工作提出微服务的“零信任”模型，通过轻量级认证（OAuthentication2.0，JWT）和服务网格实现微服务间的安全通信，实现流量分割和细粒度授权。这些工作也进一步促进了未来的 DevSec Operations 和安全问题自动化，国外的学术工作已经关注到了微服务架构的智能和自适应机制设计，通过 AI 和机器学习技术来实现微服务部署调度优化、容器调度、异常发现自动化等，将微服务系统变为动态自适应系统。

国内的微服务研究应用在 2010 年前后肇始，2007 年，阿里巴巴淘宝应用分布式服务框架，开启国内微服务化进程，国内涌现出 Dubbo、Spring Cloud 阿里巴巴、ServiceComb 等一批国产开源框架，企业级系统微服务化应用得到极大发展。2020 年前后，国内学者和军工科研机构开始探索复杂信息系统微服务，引入战术通信、指挥控制平台，研究内容主要是服务拆分、容器化、服务编排、跨节点一致控制。一些科研机构开始将微服务集群应用于态势信息处理平台和模拟平台，实现任务模块独立运行和微服务弹性伸缩。国内研究从框架复用和工程应用转向框架设计和性能实验、面向军事通信系统应用的微系统构型开始形成。

图2.3给出了微服务架构的演进时间轴, 从技术理论阶段到智能发展阶段过程

清晰, 从时间纬度, 展示了国外 2010 年到 2014 年的概念、理论,2015 年到 2017 年的企业实践、云原生和 2018 年到 2020 年的系统研究、2021 年至今的智能发展阶段, 体现微服务技术的成熟和应用深入。

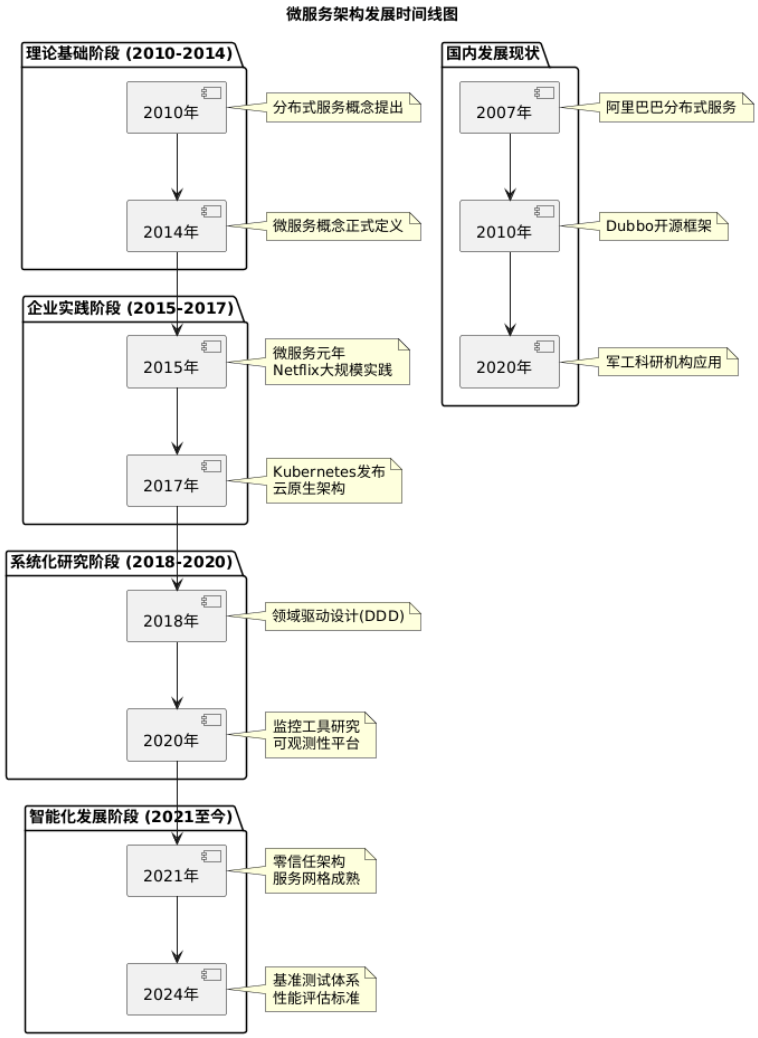

图 2.3 微服务架构发展时间线图

2.4 语义互操作

语义互操作（Semantic Interoperability）是异构系统在数据交换过程中实现“语义层理解一致”的关键能力，其核心目标是让信息在不同系统、组织或领域间传输时，不仅保持结构一致，还能被准确解释和复用。该概念最早起源于语义网与本体论研究，通过形式化语义模型描述数据背后的概念及其关系，为复杂系统间的理解

一致提供理论基础。随着信息系统复杂性与数据异构性的增加，语义互操作逐渐成为人工智能、云计算、医疗健康、工业物联网等领域的核心研究方向。

在国际研究领域，早期的语义互操作工作侧重于标准体系的构建与模型分层。SISO、NATO 和 ISO 等多层互操作模型的提出, 将信息互操作划分为语法、语义、语用层 [22, 35], 为后续研究提供了一个参考统一体模型。2010 年以后, 部分学者开始对本体 (Ontology)[25] 在语义互操作中的应用进行研究。Mishra 和 Jain 提出了“语义知识宝库”(Semantic knowledge treasure), 通过 OWL 和 SPARQL 完成异构资源统一语义表示和查询 [36]。这些研究, 标志着语义互操作由概念层面进入知识层面。

近几年，在知识图谱、自动推理、人工智能等技术的推动下，语义互操作研究向智能、自动方向研究。Bernasconi 等提出“本体解包”（Ontological Un-packing）方法对已有概念模型进行本体分析，揭示概念模型隐性语义结构，提高模型互操作性 [37]。Guizzardi 与 Guarino 则在《Semantics, Ontology and Explanation》中引入语义互操作诠释问题, 即语义透明性和本体承诺 (Ontological Commitment) 应作为提高互操作和可理解性和可信赖性的手段 [37]。此外，机器学习与规则引擎结合的语义映射算法自动获取概念映射关系, 进行跨领域知识语义对齐和重用。

随着云与分布式系统应用越来越广泛, 语义互操作也扩展到跨平台和多云场景。Hamdan 和 Admodisastro 提出了一种基于本体层 (metalayer) 的多云语义互操作参考模型, 并在参考模型基础上设计了语义中心 (Semantic Hub) 用来协调不同云场景中的语义模型, 并提供语义一致性服务 [17, 38]。同时指出，在异构场景中保持语义一致性需要在三个模型层进行协同, 即架构层、数据层和治理层; 同时语义互操作也在与数据治理、知识发现、可解释人工智能等研究进行深度融合, 为跨平台和协同提供基础能力。

国内语义互操作研究源于 2000 年代的语义网工程，近年来知识图谱和人工智能的兴起，进一步推动了语义互操作研究。语义本体构建、语义映射方法、语义信息检索、语义推理机制等研究热度持续攀升。以知识图谱为研究对象，在构建语义关系模型的基础上，对异构数据进行实体映射、语义标注，从而实现可融合性；以智慧城市、医疗、交通、工业互联网应用为研究对象，以语义互操作为核心，研究

多源数据融合与协同管理。语义互操作研究从“语义建模”转向“语义计算”的研究，以语义理解、语义推理和动态本体演化为核心，满足即时性、可解释性强的新需求。

语义互操作体系结构层次从物理层到语用层共分为 4 层架构模式。具体语义互操作系统体系结构的层次如2.4所示，物理层实现传输协议和网络互联，语法层实现数据格式、消息结构，语义层实现本体模型和概念映射，语用层实现业务规则、上下文理解等，互操作机制中的语义中枢、映射引擎、转换器、验证器体现了语义互操作技术的抽象和完整性。

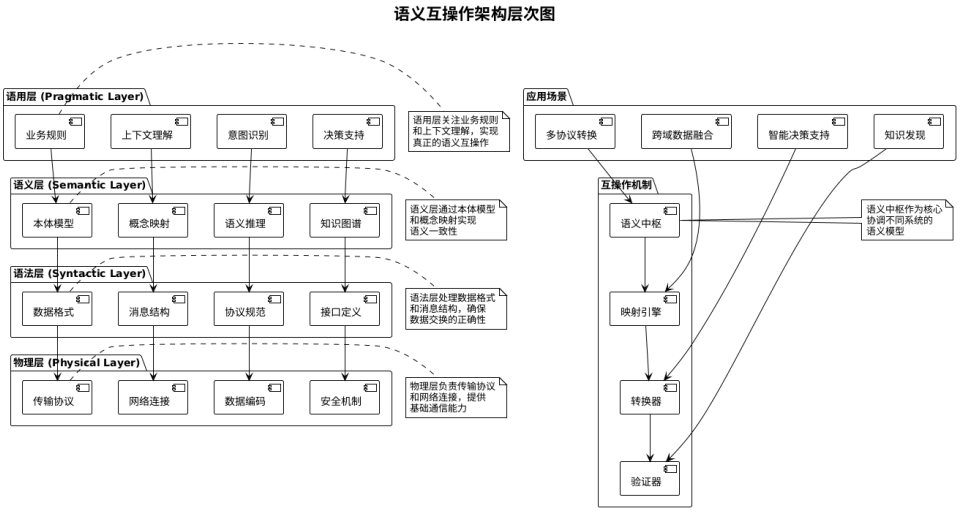

图 2.4 语义互操作架构层次图

2.5 自动化处理

自动化处理技术是复杂信息系统中的必要技术, 主要是为了将非结构化 (半结构化) 文档 (PDF、Word、XML、JSON 等文件) 进行自动解析、识别、导入数据库或其他知识系统中进行高效、精准的建模和信息抽取, 经过了基于规则的处理到机器学习、深度学习等智能处理的发展历程, 近年来, 在自然语言处理、智能文档处理 (Document Intelligence)、多模态学习等方面取得了快速发展。

早期的自动化文档处理研究以版面分析（Layout Analysis）和文本块识别为主要方法。典型工作包括基于光学字符识别（OCR）的文本抽取、表格检测与区域定位算法。这一阶段的研究主要依赖启发式规则与图像分割算法，如 PDFMiner、

Apache Tika 等工具框架，通过对文本流与版面结构的分析实现基本的内容解析。然而，这些方法难以处理复杂文档中的语义结构与跨页逻辑关系。

随着深度学习和自然语言识别的发展，更多基于网络的文档识别和语义理解模型被提出。2020 年起，Google、Microsoft、Adobe 等机构使用视觉语言联合模型 (Vision-language Models) 进行文档识别和文档阅读任务 [9, 39]，Xu 等使用联合模型对文本、位置和视觉信息进行建模，对文档结构和语义进行理解和识别，如表单识别、关键字识别、文档分类等。论文中的基于 Tensor 多模态的方法在识别 PDF 结构时获得了不错的效果。该方法为复杂结构文档识别提供了一个通用的框架。

在信息抽取与结构化导入方向，学者们提出了多种智能抽取与标准映射框架。Li 等在《DocParser: Document Parsing and Structured Data Import》[40] 中提出一种结合文本分块、实体识别与模板匹配的自动导入机制，实现 PDF 与 XML 文档的语义级结构化导入。与此同时，研究者将知识图谱构建与自动文档处理相结合，通过实体识别、关系抽取与语义对齐实现从原始文档到知识图谱的自动生成，为数据标准化与语义互操作奠定了基础。此类方法已在专利文档、医学报告与技术标准文件的自动建模中得到验证。

近年来，自动化加工逐渐发展到自动化的监督学习和跨模态理解，模型不再需要人工标注而是采用海量的通用特征预训练而成，例如 Powalski 等人提出的文档自动分类与提取表格任务中的文本-视觉双流 Transformer 模型 [41]。随后，将大规模语言模型 (LLM) 与文档知识关联起来的跨模态语义推理与任务自适应成为研究热点 [13]，文档自动化迎来了“语义理解为王”的新时代。

国内对于自动化文档处理的研究集中于结构化识别和智能导入系统的工程化运用，研究院所及科技企业进行了对于 PDF 解析、表格抽取、标注字段、标准导入的研究，开发基于深度学习的 OCR 引擎、语义分层引擎，百度文心、阿里达摩院、华为诺亚方舟实验室等都提出了面向企业文档及技术标准的多种模态解析思路，并部分运用到企业电子政务、科研档案、装备资料中。然而，国内研究仍具有跨格式迁移能力差、语义抽象层次低、自动验证及纠错能力不足的问题，在知识表示、语义约束及可解释性方面需进一步加强。

图2.5描述了文档的自动处理技术演化轨迹，描绘了规则驱动、机器学习、深

度学习等 4 个阶段。从技术维度勾勒了文档自动处理技术的 4 个发展演进轨迹，从 2000-2010 年的启发式算法的规则驱动阶段，到 2010-2020 年的视觉语言融合模型阶段，再到 2020-2023 年的深度学习的 Tensor 架构阶段，以及 2023 年以来的文档自动处理技术智能化阶段，涉及电子政务、科研档案、装备资料、技术标准等各个领域。

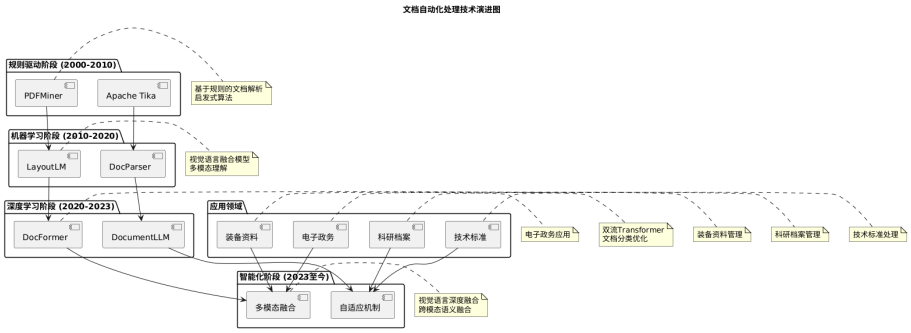

图 2.5 文档自动化处理技术演进图

2.6 本章小结

基于以上分析，本章系统梳理了战术数据链技术、微服务架构和语义互操作三个核心技术领域的理论基础、发展现状和技术特点。通过深入分析相关技术的研究进展和应用实践，为构建基于 MIL-STD-6016 标准的战术数据链信息标准数据库系统提供了坚实的理论基础。

在战术数据链技术方面，MIL-STD-6016 标准为 J 系列消息的格式化和语义化提供了统一规范，为系统设计提供了标准依据。在微服务架构方面，其分布式、模块化的设计理念为构建高性能、高可用的数据库系统提供了技术支撑。在语义互操作方面，基于本体的语义映射机制为实现跨标准的数据融合和互操作提供了关键技术路径。

基于以上技术分析，本研究将采用微服务架构构建战术数据链信息标准数据库系统，通过语义互操作技术实现 MIL-STD-6016 消息的标准化存储、语义化处理和跨标准互操作，为多链融合提供统一的信息支撑平台。下一章将详细阐述系统的总体架构设计和核心算法研究。

第三章　系统需求分析

本章主要对基于 TIL-STD-6016 数据链信息标准数据库与应用平台进行系统需求分析, 确定系统的功能需求和非功能需求, 为后续的系统设计与实现提供依据。

战术数据链系统是当前联合作战中通信手段中最基础的通信手段, 信息标准数据库是提高作战效能, 增强系统互操作能力的关键。随着网络中心战概念的提出和发展, 对多域作战需求, 传统战术数据链系统数据繁杂, 版本标准不一, 多链路间互操困难等问题越来越明显, 构建一个统一的、高效的、可扩展的战术数据链信息标准数据库系统是当前军事信息建设的新方向。

本章将基于系统需求分析角度描述基于 MIL-STD-6016 标准的战术数据链信息标准数据库系统的功能需求、非功能需求、数据特征与处理需要、用户角色与交互需求，根据以上系统的信息需求，对后期系统架构设计、系统数据库建模、系统前后端开发与系统集成测试提出需求与约束。

3.1　功能需求

根据前期调研和标准分析，结合实际应用场景，系统需要实现以下核心功能：

3.1.1　数据特征与处理需求

战术数据链消息具有与传统业务数据不同的特性，其处理需求也更为复杂。为保证数据库的适用性与系统的实用性，需从数据来源、结构特征、处理方式与存储管理等角度加以分析 [42]。

系统数据主要来源于 MIL-STD-6016、MIL-STD-3011、STANAG-5516 等标准文档，并结合 MAVLink、NMEA-0183、ARINC-429 等协议数据。数据类型主要包括：

（1）标准消息数据：J 系列报文（J2.0、J3.0、J7.0、J12.0 等）及其字段定义；

（2）语义概念数据：基于 CDM 四层法的概念库和字段映射关系；

（3）多格式文档数据：PDF、XML、JSON、CSV 等格式的标准文档和配置文件；

（4）跨协议转换数据：不同协议间的消息转换和映射规则。

表3.1详细展示了战术数据链数据特征与处理需求的对应关系，该表从数据类型、结构特点、处理需求和存储实体四个维度进行了系统性的分析。

表 3.1 战术数据链数据特征与处理需求概览

数据类型	结构特点	处理需求	存储实体
标准消息数据	多字段/比特位	报文解析、完整性校验	消息表、字段表
语义概念数据	概念层次/同义映射	概念绑定、语义一致性	概念表、绑定表
多格式文档数据	格式多样/结构复杂	多格式解析、智能识别	文档表、解析结果表
跨协议转换数据	格式差异/协议适配	格式转换、协议映射	映射表、转换规则表

3.1.2 标准消息管理

战术数据链信息标准数据库系统需要支持多种消息标准的统一管理 [23]。根据实际应用需求分析，系统应具备以下核心能力：

（1）多标准消息支持：系统需要同时支持 MIL-STD-6016 的 J 系列消息（J2.0、J3.0、J7.0、J12.0 等）、MAVLink 的飞行器通信消息 (HEARTBEAT 和 ATTITUDE、VERSAGE、ATTIME)、NMEA-0183 的导航消息 (GGA、RMC 和 VTG 等)，每一种消息都具有特定的格式、语义和适用范围，系统要能够完整地存储管理消息的定义信息 (消息基本信息、消息结构、消息语义信息)。

系统支持的标准和协议如下表3.2所示。

表 3.2 系统支持的标准和协议

标准名称	描述	版本	主要消息类型
MIL-STD-6016	美军标准 6016 - 战术数据链消息标准	A, B, C	J2.0, J3.0, J7.0, J12.0, J13.0
MIL-STD-3011	美军标准 3011 - 联合战术信息分发系统	A, B	J2.0, J2.2, J3.0, J3.1, J3.3
STANAG-5516	北约标准 5516 - 战术数据交换	1, 2, 3	J2.0, J3.0, J7.0, J12.0
MAVLink	微型飞行器通信协议	1.0, 2.0	HEARTBEAT, ATTITUDE, POSITION, GPS_RAW_INT
NMEA-0183	海洋电子设备数据格式	2.0, 2.1, 2.2, 2.3	GGA, RMC, VTG, GLL, GSA
ARINC-429	航空电子设备数字信息传输	15, 16, 17	A429, A629

（2）消息录入与维护：考虑到标准文档的复杂性和多样性，系统需要支持多种数据录入方式。传统的逐条录入方式效率低下，无法满足大规模标准文档的处理

需求。系统需要提供基于 PDF 文档的自动化解析功能，能够从标准文档中自动提取消息定义信息，并支持 CSV、Excel、XML、JSON 等多种格式的批量导入和单个录入两种方式。

（3）消息字段管理：J 系列消息复杂, 消息字段由字段名、起始位置、结束位置、长度、描述信息等属性组成, 系统应能支持复杂字段管理并建立字段与消息的绑定关系; 系统应能支持字段层次管理, 支持字段组、子字段等, 便于管理和理解消息。

3.1.3　字段与语义概念绑定

为提升消息语义一致性，系统需要支持字段与语义概念的绑定 [43]。根据实际应用需求分析，系统应具备以下核心能力：

（1）语义概念库构建：需建立统一的语义概念库，包含战术数据链领域中的核心概念，如平台标识、位置信息、时间信息、任务状态等。每个语义概念都应具有明确的定义、属性描述和使用规则，支持概念的层次化组织和继承关系。

（2）字段绑定机制：系统需要支持自动绑定和手动绑定两种方式。自动绑定基于字段名称、数据类型、取值范围等特征进行匹配，能够快速建立初步的绑定关系。手动绑定允许专家用户根据领域知识进行精确的绑定操作，确保绑定的准确性和完整性。

（3）置信度管理：在功能上需要为每个字段-概念绑定关系分配置信度值，反映绑定的可靠程度。置信度可以通过字段名称相似度、数据类型匹配度、专家验证结果等因素计算，并提供置信度阈值设置功能，允许用户根据应用需求调整绑定标准。

（4）动态绑定更新：考虑到战术数据链标准的不断演进，应支持语义绑定的动态更新，当标准版本更新或新增消息类型时，能够自动检测需要重新绑定的字段，并提供批量更新功能。

3.1.4　多链路互操作支持

考虑到战术数据链存在多标准并行的情况，系统需要具备跨链路的互操作支持功能 [44]。根据实际应用需求分析，系统应具备以下核心能力：

（1）CDM 四层法架构：系统需采用 CDM（Common Data Model）四层法架构实现多协议互操作。概念层定义统一的语义概念库，包含作战实体、态势要素、指挥关系等核心概念；映射层通过声明式规则定义协议间的字段映射关系；转换层实现具体的消息转换逻辑；运行层提供协议中介和转换引擎。系统支持 MIL-STD-6016、MAVLink、MQTT、NMEA-0183 等协议的互操作转换，这种分层架构确保系统的可扩展性和可维护性。

图3.1展示了 CDM 四层法架构，包括语义层（统一概念库）、映射层（YAML 配置规则）、校验层（一致性验证）和运行层（转换引擎），展现了多协议间的语义互操作和实时转换。

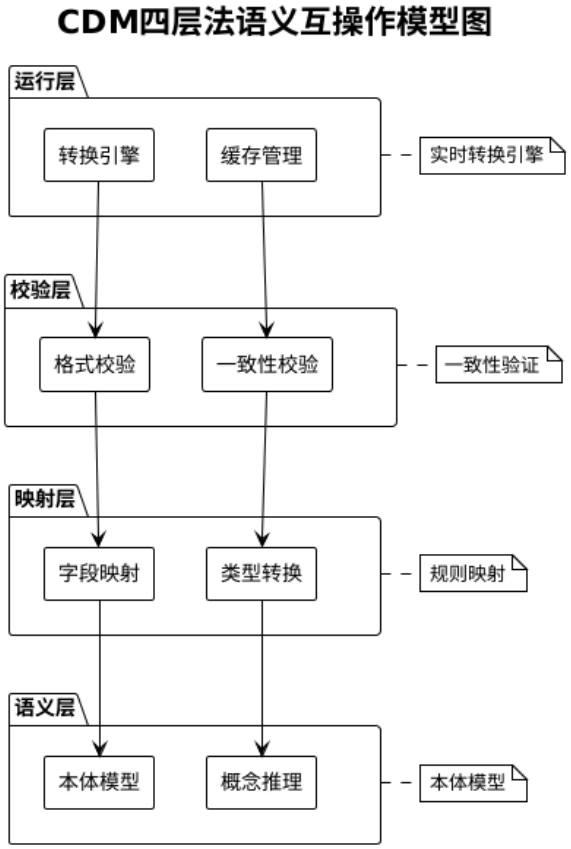

图 3.1 CDM 四层法架构图

（2）智能消息转换引擎：应实现基于规则和机器学习的智能转换引擎，能够自动处理协议间的语法差异、语义差异和时序差异。转换引擎需要支持多种转换策略，包括精确匹配、模糊匹配、语义推理等，并提供转换质量评估和置信度评分机

制。

（3）统一 API 网关：需提供统一的 API 网关，支持多种协议的消息转换和路由。网关需要集成 CDM 四层法和语义互操作两种处理方式，能够根据源协议和目标协议自动选择最优的转换策略。下表3.3列出了系统提供的 5 个主要 API 接口及其功能，通过 RESTful 架构设计为前端界面和外部系统提供标准化数据交换接口。

表 3.3　系统 API 接口功能表

API 接口	主要功能	支持格式	应用场景
/api/v2	消息转换、概念管理、映射管理、系统统计	JSON, XML	统一文档处理与语义互操作
/api/cdm	CDM 概念创建、映射规则管理、消息转换	JSON, YAML	CDM 四层法互操作处理
/api/semantic	语义字段管理、消息映射、路由处理	JSON, XML	语义互操作系统
/api/pdf	PDF 文档解析、表格提取、数据处理	PDF, JSON	MIL-STD-6016 文档处理
/api/mqtt	MQTT 消息处理、协议转换、数据路由	JSON, MQTT	MQTT 协议消息处理

（4）协议适配支持：功能设计上应支持 MIL-STD-6016、MAVLink、MQTT、NMEA-0183、ARINC-429 等多种协议的消息对接，能够处理不同协议间的消息格式差异、语义差异和时序差异。通过声明式映射规则和版本治理机制，系统需要能够灵活应对协议演进和标准更新。

3.1.5　前端交互与可视化

基于前述数据库操作与维护的需求，战术数据链信息标准数据库系统面向的用户群体包括系统管理员、作战指挥员、研发人员等，这些用户具有不同的技术背景和使用需求，因此前端系统应具备以下核心能力：

（1）统一用户界面：需提供统一文档处理与语义互操作平台，包含消息处理、文件处理、概念管理、映射管理、系统概览等核心功能模块，支持用户通过消息号、字段名、J 系列类别、时间范围等多种条件进行精确查询，并支持查询条件的保存和重用，允许用户创建常用的查询模板。

（2）数据展示与搜索：应支持多种数据展示方式，包括表格、图表、关系图等，其中表格展示需支持排序、筛选、分页等功能，并提供数据导出能力，图表展示应支持多种图形显示方式 (包括条形图、饼状图、折线图等), 让用户直观地了解数据的分布与趋势。另外, 应支持智能搜索引擎, 提供模糊匹配、语义搜索等功能, 使用

户能够便捷地找到所需要的信息。

（3）交互式操作：为确保良好的用户体验和系统可用性，需支持交互式操作和响应式设计。系统应支持拖拽、点击、缩放等交互操作，允许用户通过直观的手势操作来浏览和操作数据，并提供上下文菜单、快捷键等便捷操作方式，提高用户的操作效率。

系统需针对不同用户提供差异化的交互模式 [45]：

（1）图形化交互：前端界面提供消息检索、态势展示和跨链协议映射的可视化操作，支持 CDM 互操作接口、语义互操作接口和统一处理器接口，降低使用门槛，如图3.2所示。

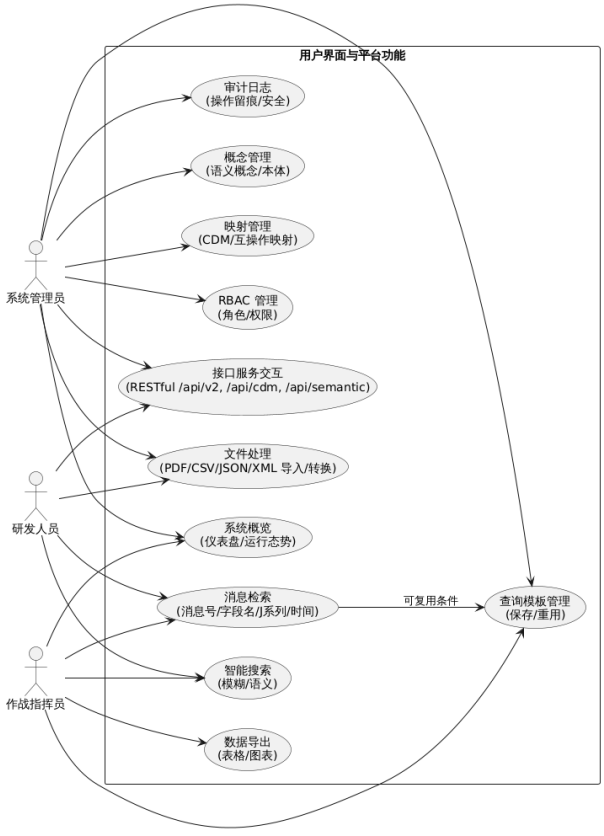

图 3.2 前端交互用例图（角色与功能关系）

（2）命令行接口：为研发与仿真人员提供批处理与脚本化调用，支持大规模数据处理与自动化测试。

（3）接口服务交互：通过 RESTful API 与外部仿真平台或作战系统对接，支持

标准化协议调用和跨域互操作，提供/api/v2、/api/cdm、/api/semantic 等统一 API 接口，如图3.3 所示。

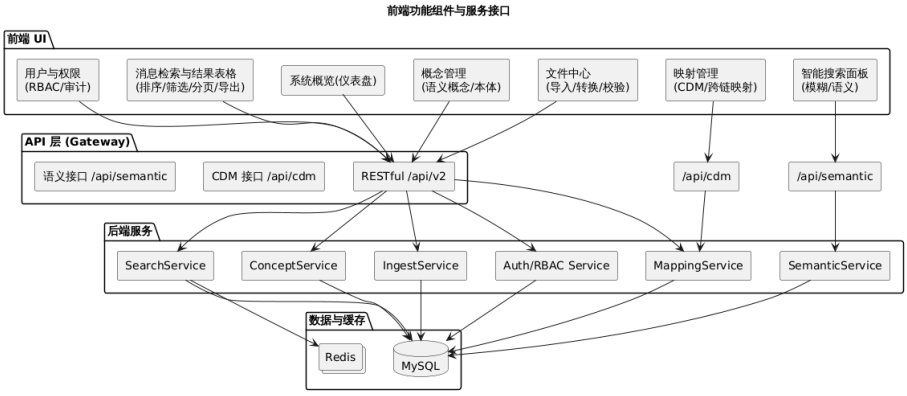

图 3.3　前端功能组件与服务接口关系图

3.1.6　仿真与验证接口

作为战术数据链验证和测试的输入源，高质量的仿真数据当然不可或缺；而高质量的仿真平台同样离不开可靠的消息定义和消息格式，所以系统需要构建高可用的仿真/数据接口。

而对于面向接口，需要系统提供标准的面向接口（RESTful API）实现，能够支撑仿真平台的数据获取、消息发送、结果获取等基本操作；接口实现为标准的 HTTP 实现，定义状态码，接口易于维护，可支持多种标准的消息格式类型（JSON，XML，YAML）；消息格式定义了消息结构、字段、语义信息等，仿真平台可基于此对消息内容做出正确的理解与处理。

3.2　系统非功能性需求分析

作为保障作战指挥的信息系统，战术数据链数据库与应用平台还需要具备较好的非功能性，即平台除开功能性需求外，需具备良好的非功能性、可扩展性、安全性，保障平台在复杂作战环境下的正常使用及系统的发展扩展 [46]。

3.2.1 性能需求

基于战术数据链系统的实时性特征和作战指挥的时效性要求，系统性能需求分析应从响应时间、数据吞吐量和系统可靠性三个维度进行考量 [47, 48, 49]。响应时间需求分析表明，多用户并发访问场景和实时态势更新的业务需求对系统响应性能提出了严格要求。通过分析战术数据链消息解析与字段检索的快速响应要求，可以确定普通查询请求的平均响应时间应控制在 2 秒以内，这一指标直接影响作战指挥的决策效率。为满足上述性能要求，系统需要实现缓存机制，包括 Redis 缓存和查询结果缓存，对于频繁访问的消息类型和字段定义实现预加载策略，同时建立数据库索引支持多条件组合查询的快速执行。

而对吞吐量的需求要求系统能够支持大批量数据的导入导出, 以便进行标准文档与系统初始化的批量处理, 通过对仿真测试场景下对数据处理的要求分析, 可以得到仿真接口需要支持连续处理大量并发消息, 支持消息实时生成与回放, 满足仿真测试中大批量数据处理需求 [50, 51, 52, 53]。对系统可靠性需求分析, 复杂、不确定的战术环境要求系统容错和数据一致性, 通过对数据安全性和一致性需求分析, 可以得到对于数据库的要求是支持备份和日志恢复, 确保异常下数据的安全, 消息存储和语义绑定过程的事务一致性约束对于系统可靠性要求较高, 避免不一致 [54, 55]。

3.2.2 安全性需求

战术数据链系统承载着敏感的作战数据, 安全性是系统的首要需求, 系统在数据存储、传输、访问控制等方面都有基本的要求, 确保系统内信息不被篡改、泄露和非法窃取 [56]。系统在数据存储、数据传输方面采用加密技术, 在数据库级别将敏感字段加密存储, 在系统接口采用基于 TLS/SSL 的安全传输协议, 防止中间人接入, 消息交换采用密钥管理机制, 密钥定期更换, 防止密钥泄露 [57]。

为防止非法访问与误操作，需要建立访问控制机制，提供基于角色的访问控制（RBAC），不同用户角色具备不同权限范围，对数据库的读写操作需进行身份认证与授权，提供日志审计功能，记录用户操作轨迹，便于事后追溯 [58]。系统应提供数据校验与完整性验证机制（如哈希校验），在通信中断或错误发生时，系统可通

过重传机制与数据恢复策略保证一致性 [59, 60, 61, 62, 63, 64, 65, 66]。

3.2.3　可扩展性需求

根据战术数据链标准和作战样式发展趋势, 应能够适应标准、吸收协议、拓展构架, 这也是系统扩展性需求分析需要考虑的三个层次 [67]。适应标准需求分析表示应适应标准不同版本以及 NATO STANAG 扩展, 考虑 MIL-STD-6020/STANAG5602 等互操作标准的发展, 应该有新接口/协议快速接入能力、数据库设计为块状的可扩展设计并支持新消息类型、新字段、新语义映射, 应能够适应标准版本变化等内容 [68, 69]。

而根据协议融合需求分析, 多链路并用的跨域作战场景下, 需要融合不同链路的数据协议, 通过对当前 TTNT、JREAP 等新兴数据链发展趋势进行梳理, 确定系统架构应留有拓展接口、提供适配层、满足不同链路的不同数据结构映射和不同消息语义 [70, 71]。从架构扩展需求分析, 系统整体架构应能够适应未来规模化、复杂化应用场景要求, 通过对微服务架构和模块化设计架构优势梳理, 确定后端应基于微服务架构按需部署、前端应支持模块化扩展便于新型可视化/交互模块部署、数据库应支持可扩展结构、数据支持分片/多节点扩展 [72], 提供接口统一支持与外界互联、采用 RESTful API 协议便于外部仿真平台/指挥信息系统接入、提供统一标准数据交换格式 (如 JSON、XML) 便于与异构系统交互、提供 API 文档/开发者支撑便于未来扩展/二次开发 [73]。

第四章　微服务架构与跨数据链协议互操作系统设计

本章基于 MIL-STD-6016 战术数据链信息标准，设计并实现微服务架构的跨协议互操作系统。系统架构采用四层分层设计，通过微服务模块化实现多标准信息模型的自动导入、语义对齐与协议转换。本章依次阐述系统总体架构、微服务实现、数据模型设计、跨协议互操作架构以及自动化导入系统。

4.1　系统总体架构设计

4.1.1　设计目标与总体思路

系统以 MIL-STD-6016 战术数据链信息标准为核心，采用微服务架构构建跨协议互操作平台。设计目标包括：多源标准语义互操作、模块化弹性部署、自动化数据处理。

多源标准语义互操作要求系统支持 MIL-STD-6016、STANAG 5516、MIL-STD-6020、MQTT、MAVLink 等协议的语义对齐。通过统一语义模型和概念映射机制，实现跨标准数据的准确转换，保障不同协议间的语义一致性。

模块化弹性部署通过微服务架构将复杂系统拆分为独立服务模块。每个服务专注特定业务功能，支持独立开发、测试、部署和扩展，提高系统的可维护性、可扩展性和容错能力。

自动化数据处理实现标准化文档的自动识别、结构化提取和协议转换。系统减少人工干预，提高数据处理效率和准确性，支持战术数据链信息处理的自动化。

4.1.2　微服务架构理念与原则

系统采用” 高内聚、低耦合、自治服务” 的设计理念，遵循四个核心原则：服务拆分、服务治理、数据管理、通信机制。

服务划分基于业务领域, 保证每个微服务的独立性和单一职责。微服务负责业务中的某一部分, 界限清晰, 彼此间通过明确定义的接口 (API) 进行交互, 避免紧耦合依赖。

服务治理包括注册发现、配置中心、监控系统、熔断器。服务注册中心自动发现结合负载均衡，配置中心统一管理与热加载，监控系统健康监控的同时也进行统计分析。

数据管理采用数据库解耦及分布式事务一致性保证，每个微服务都有自己的数据库，采用事件驱动分布式事务一致性保证，事件驱动分布式事务一致性保证的 Saga 模式，数据解耦和避免单点故障。

通信机制结合了通信 + 同步（REST/gRPC）+ 异步（消息队列）。实时性高的场景采用同步通信，大批量处理、事件通知采用异步通信，通信效率、系统性能。

4.1.3　架构总体分层

系统整体采用四层分层架构，如图4.1所示：

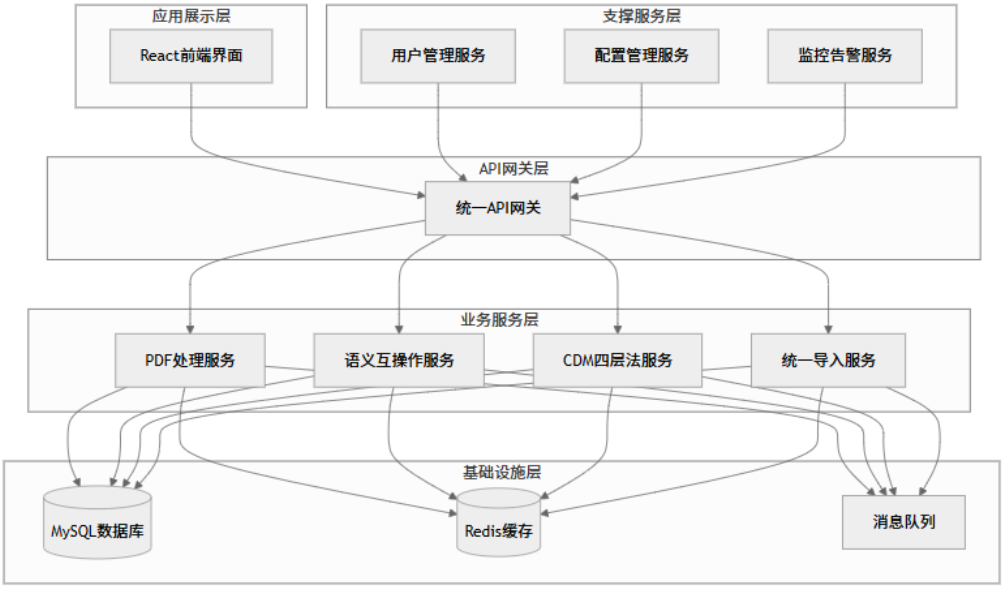

图 4.1　系统总体架构分层图

API 网关层提供系统统一接口，进行路由转发、认证鉴权、权限控制、限流熔断以及监控统计。外部的请求通过 API 网关层进行接收，对请求进行身份识别、鉴权、路由转发、聚合响应，通过限流、熔断对后端服务进行防护。

业务服务层包含 PDF 解析、语义互操作、CDM 四层法、统一导入等核心业务模块。每个服务作为独立业务单元，支持独立开发、测试和部署，体现微服务架构

的模块化设计理念。

支撑服务层包括用户认证、配置管理、监控告警、文件日志等系统支撑。支撑服务层是系统的核心，是整个系统运行的基础，包括用户认证、配置管理、监控告警、文件日志等功能。

基础设施层包含服务注册发现（Consul/Kubernetes）、消息队列（RabbitMQ/Redis）、数据库与缓存集群等核心组件。该层实现服务发现、消息传递、数据存储和缓存等基础功能，为微服务架构提供技术支撑。

4.2　微服务架构设计

4.2.1　微服务模块划分与职责

系统共包含五类核心服务，每个服务都有明确的职责和边界，如表4.1所示：

表 4.1　微服务模块划分与职责

模块	核心职责
pdf-service	自动化标准文档解析与结构化导入
semantic-service	跨标准语义分析与字段映射
cdm-service	CDM 四层语义互操作（语义层/映射层/校验层/运行层）
import-service	多格式文件识别、清洗与批量导入
api-gateway	统一接口访问控制、负载均衡、服务监控

（1）pdf-service：负责自动化标准文档解析与结构化导入。服务处理 MIL-STD-6016、STANAG 5516 等标准文档，自动提取消息定义、字段信息和约束条件，转换为结构化数据格式。

（2）semantic-service：实现跨标准语义分析与字段映射。服务提供语义分析引擎，识别不同标准中的语义概念，建立概念间映射关系，支持人工标注和规则学习。

（3）cdm-service：实现 CDM 四层语义互操作。服务基于 Common Data Model 四层架构，提供语义层、映射层、校验层和运行层的完整实现，支持不同协议间的语义级转换。

（4）import-service：负责多格式文件识别、清洗与批量导入。服务支持 PDF、Excel、XML、JSON 等多种格式的文件处理，提供格式自动识别、数据清洗和批量导入功能。

（5）api-gateway：提供统一接口访问控制、负载均衡、服务监控。服务作为系统统一入口，负责请求路由、身份验证、权限控制、限流熔断和监控统计。

4.2.2　微服务通信机制

根据微服务架构的分布式特性, 设计多种通信方式满足不同场景和通信安全需求。同步通信设计采用 REST API、gRPC、GraphQL 三类协议,REST API 提供统一的标准化 HTTP 接口和简单的 CRUD 接口, gRPC 提供高性能的内部服务调用接口, GraphQL 提供复杂数据查询的灵活性。异步通信设计采用 RabbitMQ 提供消息和事件通知服务, 确保系统关键业务数据的准确到达。服务发现设计采用 Consul 注册发现、Kubernetes DNS 提供服务发现和负载均衡, 实现系统服务启动时自动注册到服务发现中心, 其他服务以服务名调用, 松耦合实现服务通信。为确保通信安全，各服务之间通信均采用 TLS 通信，采用 JWT 令牌进行认证，建立安全屏障。

4.2.3　容错与弹性设计

考虑到分布式系统的复杂性和不确定性, 系统提供了完善的容错和弹性策略。熔断器策略作为第一道屏障, 当服务调用失败率达到预设值时, 启动熔断器, 进行服务熔断、服务快失败、服务故障隔离, 防止故障的级联。重试策略基于指数退避算法和智能重试算法, 对临时性故障自动重试, 防止故障服务的过多接入。降级策略考虑当前系统负载过高或部分服务不可用时, 自动降低服务复杂度, 降级至简单模式, 提供核心可用服务。自动伸缩策略使用 Kubernetes HPA 对服务根据 CPU、内存资源占比等指标自动扩充或缩减服务实例数量, 实现弹性伸缩。

4.3　数据模型与数据库设计

4.3.1　设计目标与数据特征

战术数据链信息标准数据库的复杂性和多样性特征决定了系统必须采用” 标准化存储、语义扩展、互操作可追溯” 的设计原则，构建支持多标准数据管理的核心能力体系。统一建模与版本化管理机制确保系统能够支持 MIL-STD-6016、STANAG 5516、MIL-STD-6020 等多个标准版本的数据存储，每个标准版本具有独立的版本标识和变更历史，保障不同标准版本数据的独立性和可追溯性。

字段级语义绑定与跨标准映射提供语义互操作能力基础, 通过语义的跨标准映射, 将数据库中的每字段与语义绑定, 具有跨标准的字段级映射与转换支持, 映射关系包含置信度、转换规则、版本, 提供语义互操作的数据转换支持。此外，还支持高性能的查询与语义检索, 支持标准版本、消息类型、语义概念等的查询, 支持全文检索、模糊匹配等功能。

4.3.2　核心实体与关系模型

系统的核心数据模型如图4.2所示，主要包含以下核心表：

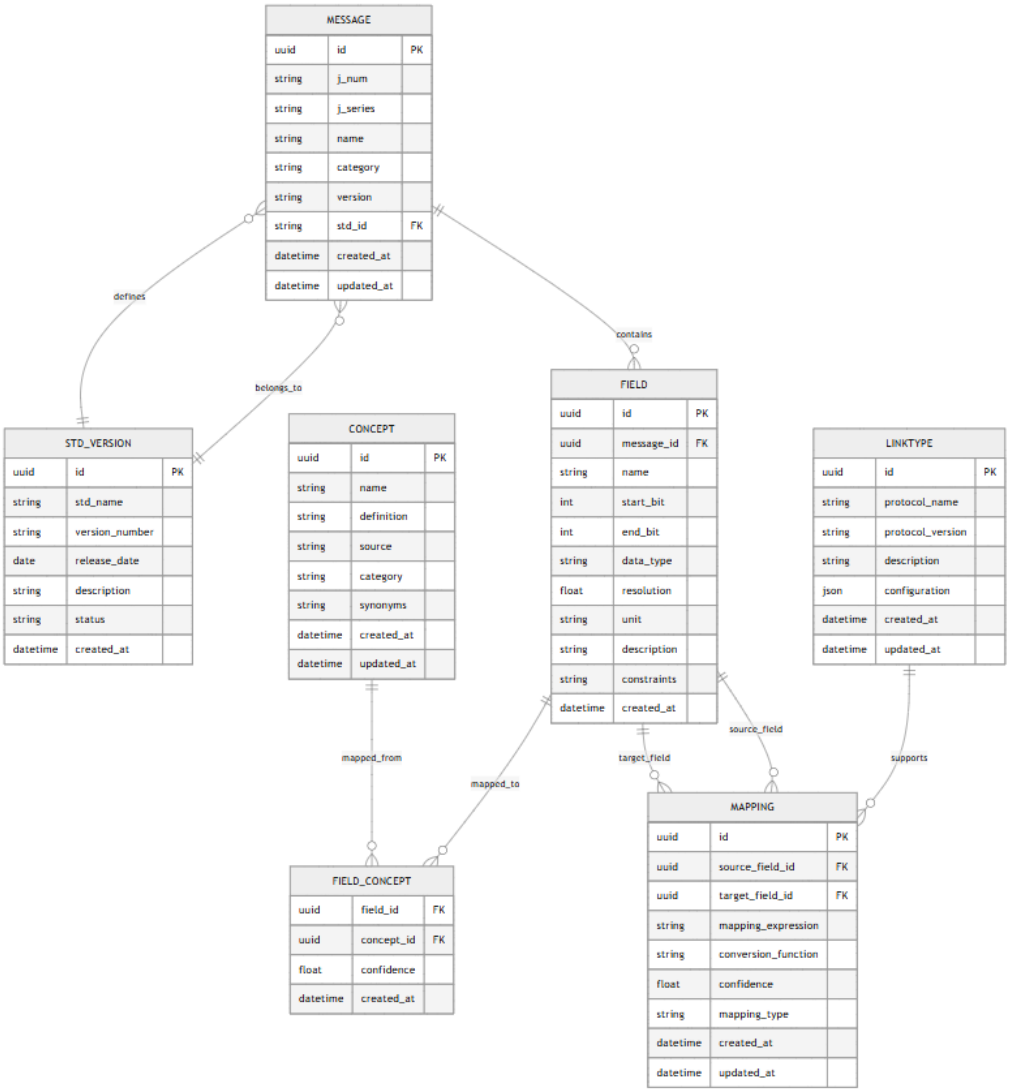

图 4.2　核心数据模型 ER 图

（1）MESSAGE 表：存储消息元信息，包含消息编号、名称、类别、版本等基本信息。该表是系统核心表之一，每个消息具有唯一标识符和版本信息，为战术数据链消息的标准化管理提供数据基础。

（2）FIELD 表：存储位段定义及约束信息，包含起始位、结束位、分辨率、取值域等详细信息。该表与 MESSAGE 表通过外键关联，支持一个消息包含多个字段的复杂结构。

（3）CONCEPT 表：存储语义概念库，定义术语与来源信息，为语义互操作提供概念基础。该表支持概念的定义、分类和关系管理，构建完整的语义概念体系。

（4）MAPPING 表：存储跨标准映射规则，包含表达式、转换函数、置信度等信息。该表支持不同标准间的字段映射和转换规则定义，为语义互操作提供精确的转换机制。

（5）STD_VERSION 表：存储标准版本管理信息，记录标准的版本号、发布日期、修订历史等关键信息。该表为多标准版本管理提供完整的版本控制机制。

（6）LINKTYPE 表：存储协议类型定义与配置信息，支持不同数据链协议的配置和管理。该表为多协议支持提供灵活的配置机制。

4.3.3　约束与索引设计

数据库设计采用严格的约束和高效的索引策略，确保数据完整性和查询性能，构建了完善的数据管理机制。

主外键采用了 UUID 主键 + 业务唯一约束的设计。其中 UUID 主键全局唯一，避免了分布式环境下的主键冲突问题; 业务唯一约束 (MESSAGE(j_num, std_id)) 避免了业务逻辑的混乱。

完整性约束实现位段检查（start_bit < end_bit）、置信度范围（0-1）等多种约束机制。这些约束通过数据库的 CHECK 约束实现，在数据层面确保数据的有效性，防止无效数据的入库。

索引策略设计组合索引（std_id, j_series, j_num）和全文索引（概念模糊检索）等多种索引类型。组合索引支持多条件查询，提升复杂查询的性能；全文索引支持语义概念的模糊检索。

性能优化采用分区表和缓存。大量数据的表采用分区减少查询量，水平和竖直相结合，提升性能; 热点数据使用缓存，采用 Redis 来提升访问速度。

4.3.4　微服务数据库分离与一致性

各微服务采用独立的数据库设计，通过多种机制保证数据一致性，构建了完善的分布式数据管理体系。

数据库分离要求每个微服务拥有独立的数据库，避免数据耦合和单点故障问题。这种设计提高系统的可扩展性和容错能力，使得各个服务能够独立演进和部

署。

一致性机制采用 Saga 模式以及事件驱动并发式机制组合的方式来实现最终一致性。Saga 模式将分布式事务切割成多个本地事务并通过补偿模式来确保数据一致性, 解决分布式事务中数据一致性。

数据同步通过 CDC（Change Data Capture）机制捕获变更事件，实现数据的实时同步。CDC 机制精确捕获数据库的变更操作，将变更事件发送到消息队列，为数据同步提供可靠的技术保障。

跨服务使用消息队列实现跨服务数据同步。当服务有变更时，系统将变更后的消息放入消息队列，并通知其他需要变更的服务进行数据更新，完成松耦合的数据更新。

4.4　跨数据链协议互操作架构设计

4.4.1　多协议支持体系

系统支持多种数据链协议，包括 MIL-STD-6016、MAVLink、MQTT、Link 16 等，构建了四层互操作体系架构，如图4.3所示：

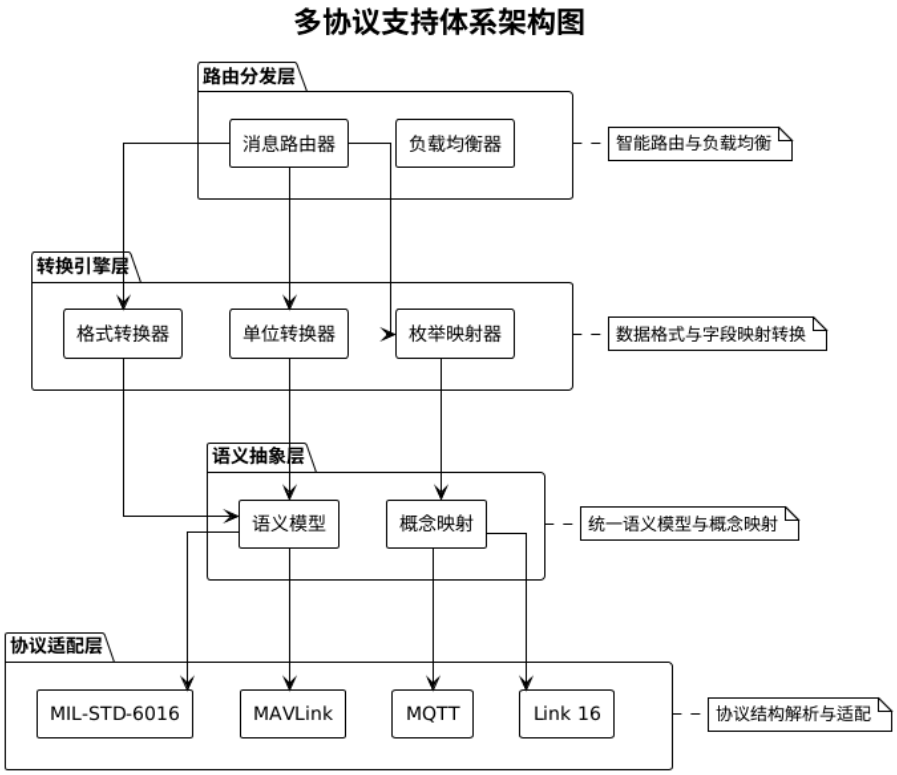

图 4.3　多协议支持体系架构图

协议适配层是互操作体系结构的最底层, 负责实现各个链路标准的结构解析适配, 实现对各种协议消息格式进行解析处理, 对字段信息进行解析, 统一内表示, 为上层处理提供一致接口。

语义抽象层建立统一的语义模型和概念映射机制, 将各个协议中的概念映射到概念空间, 采用语义建模的方式实现不同协议间的概念映射和语义理解, 为协议间进行语义互操作奠定概念基础。

转换引擎层提供协议到协议的数据格式以及字段映射转换, 如格式转换、单位转换、枚举转换等, 提供丰富的转换规则配置方式以支持复杂的数据转换规则配置, 转换规则灵活, 支持复杂数据转换。

路由分发层负责消息的路由和负载均衡, 将消息分发给正确的目标协议, 实现负载均衡和故障转发。路由分发层采用智能路由算法进行消息路由, 优化性能和可靠性。

4.4.2　CDM 四层法语义互操作模型

基于”Common Data Model (CDM)” 四层方法，系统实现了协议级语义对齐，构建了完整的语义互操作体系，如图4.4所示：

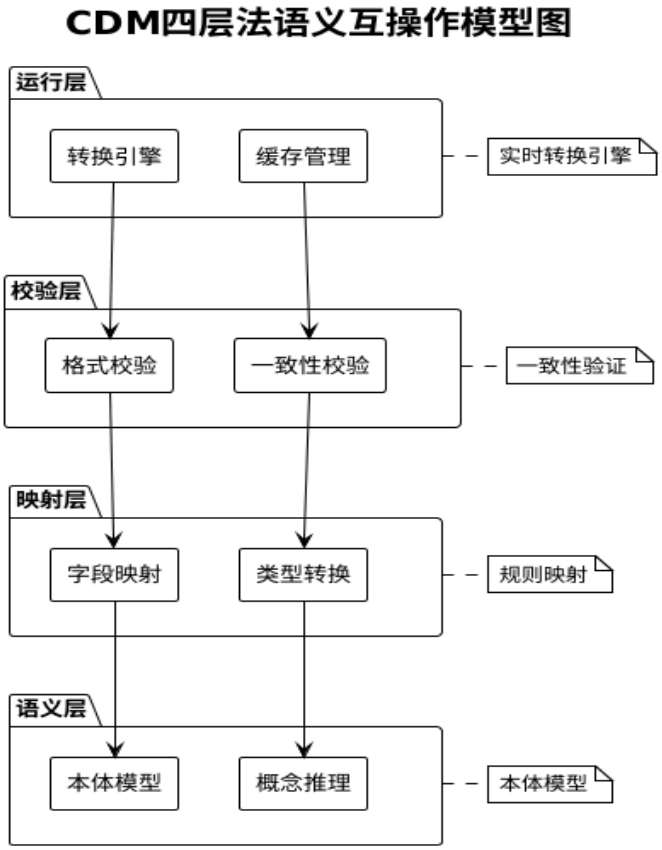

图 4.4　CDM 四层法语义互操作模型图

语义层是 CDM 的核心层, 构建统一的本体模型及概念推理系统。通过本体技术建立统一的概念模型可以定义、分类和推理, 使系统能够理解概念间的语义。

映射层采用声明式规则映射和 YAML 配置化, 通过 YAML 配置文件描述映射规则, 便于规则变更与升级。映射规则包括字段映射、数据类型映射、单位映射等。

校验层提供多层次的一致性验证和金标准回归测试机制，包括格式校验、业务规则校验、一致性校验等多种校验方式。金标准回归测试确保转换结果的准确性，为语义互操作的质量提供可靠保障。

运行层提供高性能、高效率的实时转换引擎, 采用高效的转换算法实时进行消

息转换、批量处理, 提供高效的实时转换缓存及优化, 满足战术数据链的实时性高和性能高的要求。

4.4.3　语义互操作系统组成

系统包含四个核心组件，实现从概念级到消息级的自动语义互操作，构建了完整的语义互操作处理体系，如图4.5所示：

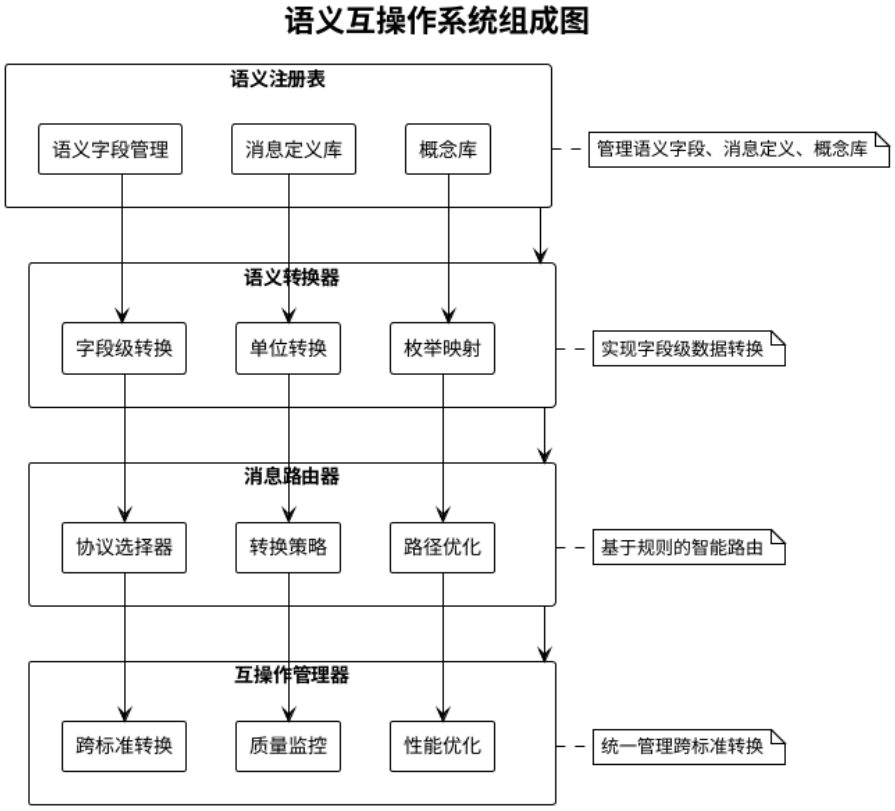

图 4.5　语义互操作系统组成图

语义注册表是系统存放语义域、消息定义、概念库等系统基础信息的核心组件, 提供语义信息的增删查改, 支持语义概念版本控制, 是进行语义信息交互操作的基础管理平台。

语义转换器实现字段级数据转换、单位转换、枚举映射等关键功能，该组件支持多种转换算法，包括数值转换、字符串转换、枚举映射等。通过灵活的转换规则配置，该组件能够处理复杂的跨协议数据转换需求。

消息路由器采用基于规则的智能路由模式,进行协议选择和转换策略的自动配置。消息路由器根据消息的类型、源协议、目的协议等,自动匹配最佳的转换策略,采用智能的算法来进行消息路由。

互操作管理器集成各种跨标准互操作转换、质量管控、性能调优等管控管理功能，此部件提供统计、错误管控、评分等转换监控与管理功能，通过对转换管理，确保了语义互操作运行的稳定高效。

4.4.4　数据一致性与冲突解决

系统采用多种机制保证跨链路数据的一致性和冲突解决，构建了完善的数据质量管理体系。

一致性协议采用最终一致性协议与版本号优先策略相结合的方式。对于非关键数据，使用最终一致性协议，在保证系统性能的同时确保数据的最终一致性；对于关键数据，使用强一致性保证，确保数据的实时一致性。

冲突解决采用时间戳优先、版本号优先、人工仲裁等策略实现，在数据冲突的时候，调用预定义的策略，自动解决冲突，通过智能冲突检测、智能冲突解决等智能算法，将数据冲突的影响范围降低到最小;必要情况下调用人工仲裁，确保复杂冲突场景下的数据准确性。

数据校验支持格式校验、规则校验、完整性校验等多层次的校验机制，每一层转换过程均进行校验，保证数据不错误、不缺失，通过多层次的校验机制确保了错误不传播。

质量保障跨链路数据同步数据质量保障，通过持续不断地对数据转换的质量，也就是准确率、完整性、一致性等的监控。实时质量保障监控和告警，系统能够实时发现数据质量问题并给出解决方案。

4.5　自动化信息标准导入架构设计

4.5.1　标准化导入流程

自动化导入系统实现从 PDF/Excel/XML 等标准文档到数据库的全流程自动处理，如图4.6所示：

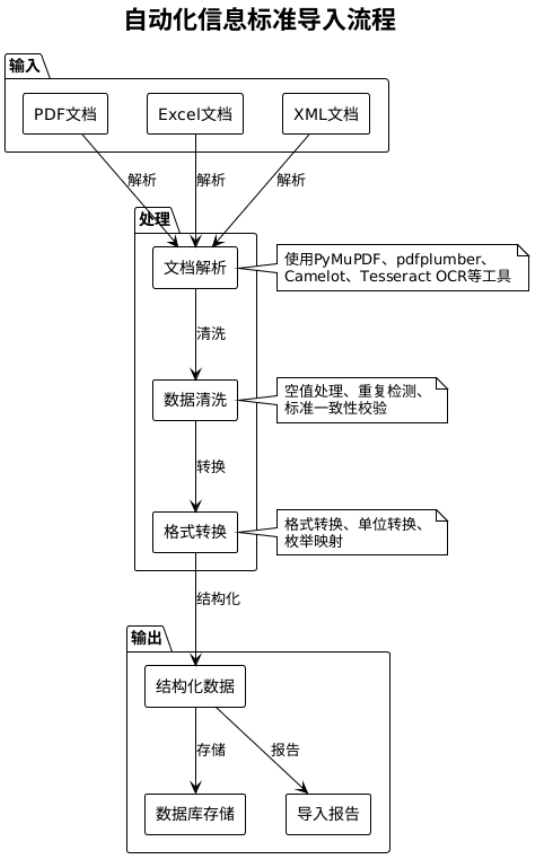

图 4.6　自动化导入流程

处理过程是文本导入-数据清洗-文本抽取-表格识别、字段分析、数据清洗、入库、形成校验报告等。每一步处理都经过检验和审核，确保了导入到系统中的数据完整、干净、准确。

文档解析阶段使用 OCR 技术和 PDF 分析库，从文档中提取文本、表格。扫描文档使用 Tesseract OCR，识别文字，多语言文本识别; 文本类 PDF 直接提取文本，多种解析技术，确保不同类型文档解析正确。

在结构化处理阶段，将提取出的文本信息进行结构化处理，将消息定义、内容字段、约束条件信息等通过规则匹配和机器学习算法进行识别，将非结构化的文档信息转化为结构化的、统一的数据结构。

在清洗数据阶段，对抽取出的数据进行清洗校验，包括：格式校验、完整性校验、一致性校验等校验。将校验出的问题写入日志错误文件，以便后续处理和分析。

导入存储将清洗完成的数据导入入库并生成导入报告，导入报告包括导入成功的记录数，导入失败的记录数，以及具体的错误信息等统计指标，为整个导入过程提供详尽的数据以便分析导入的质量和出现的问题追根溯源。

4.5.2　关键技术与工具链

系统采用多种先进的技术和工具，确保导入过程的准确性和效率，构建了完整的技术支撑体系。

文档解析工具主要有：PyMuPDF、pdfplumber、Camelot、TesseractOCR 等等，PyMuPDF 是一个十分先进的分析文档的软件，pdplumber 软件是抽取表格工具，Camelot 软件是抽取准确表格工具，TesseractOCR 是文字识别工具，多种工具结合使用，确保正确分析不同类型的文件。

结构化导入采用 Pandas + SQLAlchemy + MySQL 技术架构。Pandas 海量数据计算分析;SQLALCHEMY 负责 ORM 的支持, 方便数据库操作;MySQL 可靠地进行了数据存储, 保持数据的一致性、安全性。

格式识别使用 MIME 检测，采用格式识别与规则匹配相结合的格式识别方式。MIME 检测速度较快，识别文件的格式，为后续对文件处理提供基本信息; 规则匹配提供精确的格式识别与内容识别，通过智能识别算法，对不同的格式文档进行精准的识别。

校验机制实现字段重叠、位长一致、枚举合法等校验功能。自动对数据中存在的问题进行检查，并提示修复建议，通过智能校验机制，保证导入数据质量。

4.5.3　数据清洗与质量保证

系统提供完善的数据清洗和质量保证机制，构建了全方位的数据质量管理体系。

清洗算法支持空值、重复、一致性等检测。系统自动处理缺失值、重复值、错误值等常见错误，通过应用清洗算法对数据进行智能清洗，实现数据清洗。

质量指标监控系统数据完整性、语义保留度、转化率等。这些指标可以综合评估数据的质量，为数据质量的优化提供参考，数据质量监控系统能帮助系统及时发

现并处理数据质量问题。

验证机制提供格式验证、业务规则验证、一致性验证等多层次的验证机制, 不同的层次具有不同的验证机制, 并具有对应的错误处理, 多层次验证确保了数据的一致性和准确性。

错误处置能够完成异常记录、自动修正、人工校验等功能，系统大部分问题能够实现自动处理，仅对于复杂问题需要部分人工干预，从而通过智能的差错处理减少最大可能的数据质量问题。

4.6　微服务通信与运行保障设计

4.6.1　服务通信与安全

系统采用多种通信模式和安全机制，确保服务间的可靠通信，构建了完善的通信安全保障体系。

同步通信使用 REST API、gRPC、GraphQL 等多个协议。Rest API，用于简单的增删改查并使用统一的 HTTP 接口；gRPC，用于高性能的内部通信，二进制协议效率高；GraphQL 则主要用于复杂的数据查询，提供灵活的数据获取能力。

异步通信采用了 RabbitMQ、Redis Pub/Sub 的技术组合。采用 RabbitMQ 提供可靠的消息传递和事务支持以确保关键业务数据的可靠传输；而 Redis Pub/Sub 提供了高性能的实时通知以及支持轻量级消息传递，从而创建了同步与异步相结合的通信机制。

服务发现通过 Consul + Kubernetes DNS 实现服务的自动发现和负载均衡。服务启动时自动注册到服务发现中心，其他服务通过服务名进行调用，实现服务间的松耦合通信，提高系统的可维护性和可扩展性。

通信安全使用 TLS 双向认证、服务间认证通信安全，服务间通信均采用 TLS 加密通信、JWT 令牌认证，构建多层次安全体系。

4.6.2　分布式数据管理与灾备

系统采用分布式数据管理策略，确保数据的安全性和可用性，构建了完善的数据保护体系。

数据分离实现数据分离、数据所有权隔离，由于微服务各自独立存储数据，不会造成数据的耦合、单一失效，使系统可扩展性、容错性得到极大地提高，为微服务架构的灵活性提供了数据基础。

一致性保证采用 Saga 与溯源的方式，达成一致的一致性结果。Saga 模式是将复杂分布式事务切分为多个本地事务，并通过补偿事务的方式保证一致性结果，解决分布式事务中数据的一致性问题。

灾备机制实现多区域备份与灾难恢复机制。系统支持跨区域的数据备份和灾难恢复，确保在重大故障情况下的数据安全，通过完善的灾备体系，最大程度地降低数据丢失的风险。

数据同步支持实时同步、批量同步、增量同步等多种数据同步模式。系统能够根据数据的重要程度、实时性要求等智能选择数据同步方式。

4.6.3　配置与治理体系

系统提供完备的配置管理和服务治理能力, 形成完整的系统治理体系。

配置管理提供集中配置与隔离 (Consul + Configmap)。系统配置集中存储在配置中心, 动态更新和隔离, 通过统一的配置管理提供系统配置一致性、易维护能力, 提供多环境部署灵活配置的能力。

监控体系提供全链路监控（Prometheus + Grafana + Jaeger）能力。Prometheus 负责指标收集，提供全面的性能监控数据；Grafana 提供可视化展示，支持丰富的图表和仪表板；Jaeger 提供分布式链路追踪，实现完整的系统监控体系。

日志管理实现结构化日志与追踪链路功能。所有日志都采用结构化格式，支持日志聚合、搜索和分析，通过统一的日志管理，为系统运维和问题排查提供强有力的支撑。

服务治理提供健康检查、故障检查、负载均衡等服务的治理功能，系统自动进行服务健康检查、服务转移故障、负载均衡、智能服务治理、保障系统稳定、高可用。

4.6.4　容错与弹性设计

系统采用多种容错和弹性设计机制，确保在异常情况下的服务可用性，构建了完善的故障处理体系。

容错机制容错机制保障熔断、重试、降级机制保障系统容错服务可用当服务调用失败率达到阈值时系统自动熔断，避免系统断链；临时性问题自动重启；系统能力超负荷时，系统自动降级，保障核心服务可用。

弹性伸缩使用 Kubernetes HPA 实现弹性伸缩、资源弹性分配。系统根据 CPU 占用率、内存占用率和自定义指标，自动调整服务副本数量，动态分配资源，使系统保持高效。

故障恢复提供自动恢复、手动干预、数据修复等完整功能。系统能够自动处理大部分故障，对于复杂故障提供人工干预机制，通过智能化的故障处理，最大程度地减少故障对系统的影响。

性能保障确保响应时间保证、吞吐量稳定、故障恢复能力。系统通过多种优化技术，提供高性能和稳定的服务，满足战术数据链对系统性能和可靠性的严格要求。

第五章　系统实现、测试与性能分析

系统架构设计、核心算法研究等完成之后, 就是系统的实现阶段, 这是一个从理论到实践的过程, 也是极具挑战的一个过程, 系统不仅需要完善的功能, 还需要满足性能、安全、好用的不同需求, 经过几个月的研发测试, 最终完成了功能完善、性能优秀的战术数据链信息标准数据库系统。

5.1　系统实现架构

5.1.1　整体实现架构

基于微服务架构的分布式架构设计理念是将系统架构作为基础设计模式, 通过这种基础设计模式提供的服务间松耦合、独立部署、弹性伸缩的灵活扩展能力, 便于系统的维护与扩展。微服务架构将一个复杂的应用程序系统拆分为多个相对独立的服务, 每个服务负责特定的业务领域, 每个服务将关注点分离、业务逻辑独立。

系统构建了 9 大微服务, 每个微服务都有其明确的职责界限, 每个微服务都有独立的控制其生命周期, 每个微服务都可以进行开发、测试、部署和扩展, 从而提高开发效率和灵活性。每个微服务都必须遵循单一直责任原则, 保证微服务内的高内聚, 并且通过统一的 RESTful API 接口进行微服务间的低耦合。

数据存储架构以 MySQL8.0 为主数据库、以 Redis 分布式缓存数据库为辅数据库，之所以选择 MySQL，原因在于其具备高事务、支持外键约束、支持全文索引等能力，可以满足数据一致性需求，而之所以选择 Redis 分布式缓存数据库，原因在于其具备高查询性能与速度。

在技术栈的选型方面，选用最新技术组合，python3.10 + FastAPI + React18 + MySQL8.0+Redis。FastAPI 为微服务的基石，其主要功能有高性能异步处理、自动生成 API 文档、类型安全等。React18 作为前端应用通过 RESTful API 与后端微服务进行对接，前后端分离。

容器化部署作为微服务架构的重要支撑技术，系统采用 Docker 容器化技术将每个微服务构建成单独的容器镜像, 通过 Kubernetes 对这些容器进行编排管理, 这

种部署架构能够提供快速部署、自动扩容、自动恢复等技术能力, 能够为系统的运维管理提供良好的技术保障。

5.2 微服务架构实现

基于第四章的微服务架构设计，本节重点阐述系统的具体实现方案，包括微服务部署与通信、数据管理与存储、监控与容错三个核心方面。

5.2.1 微服务部署与通信

微服务的构建先解决服务的部署问题和通信问题，部署问题：采用 Docker 容器化部署服务，每一个微服务就是一个 Docker 容器，保证部署环境一致和部署可重用性。采用 Kubernetes 集群作为 Docker 容器的编排工具，对服务进行调度、扩缩容、生命周期管理。以声明方式部署和维护服务。

服务间通信方面, 构建了同步和异步两种通信机制, 其中, 同步通信基于 HTTP 协议, 服务之间使用 REST API 通信协议, 支持 JSON 格式数据交换和标准 HTTP 状态码处理; 异步通信则支持了发布/订阅和 Point 对点通信模式, 将 RabbitMQ 用于异步消息队列方式, 实现了服务之间消息解耦以及消息传递的效果。

微服务的部署以及服务间通信的整体部署架构如图5.1所示, 包含 Docker 容器化部署、Kubernetes 微服务集群、服务通信架构。

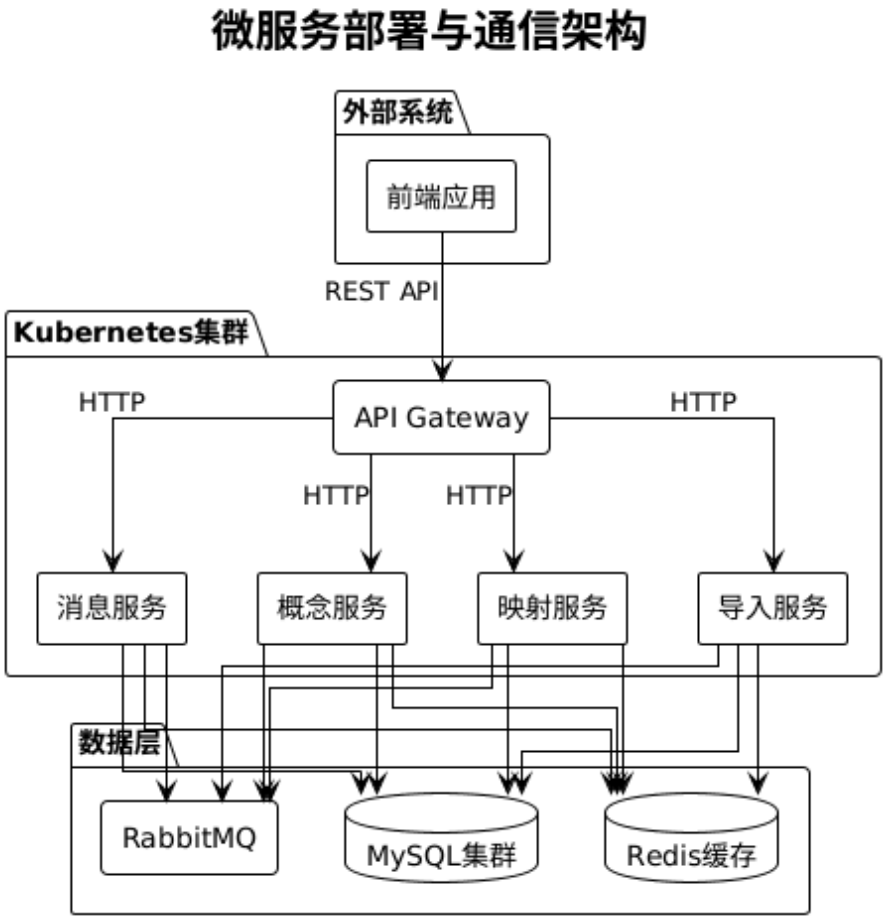

图 5.1　微服务部署与通信架构图

5.2.2　数据管理与存储

分布式数据存储是微服务架构系统中的关键。通过数据库分片, 将数据分片存储, 每个微服务负责管理一个数据片, 每个数据片通过微服务进行分布式存储访问。而选择主从复制机制, 通过将写操作操作到主库中, 读操作则分散至多个从库中, 从而实现高并发访问。

缓存采用 Redis 缓存系统，实现高速的缓存服务。采用多级缓存架构：应用缓存、分布式缓存、CDN 缓存。采用缓存预热、更新、失效策略保证缓存数据的一致性和有效性。

图5.2展示了分布式数据管理与存储架构，包括数据库分片、读写分离和缓存系统。

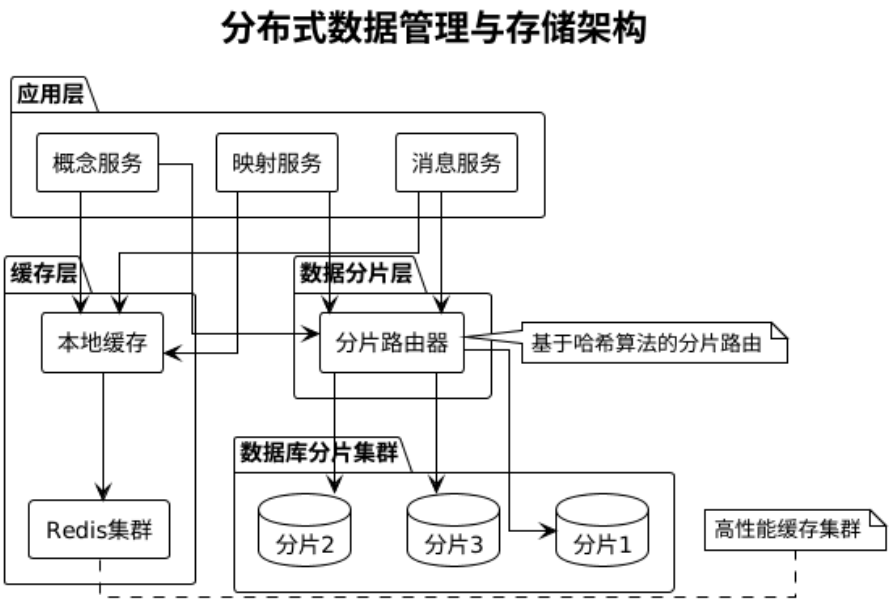

图 5.2　分布式数据管理与存储架构图

数据分片算法的核心实现如下：

Algorithm 1 数据分片算法

Input: data: 待分片的数据, shard_key: 分片键, num_shards: 分片数量

Output: shard_id: 分片 ID

```
计算分片 ID
hash_value ← hash(shard_key)
shard_id ← hash_value mod num_shards
获取分片数据库连接
shard_config ← SHARD_CONFIGS[shard_id]
shard_db ← connect_database(shard_config)
存储数据到对应分片
shard_db.insert(data)
return shard_id
```

5.2.3　监控与容错

服务监控体系是微服务系统能否正常工作的基础，系统集成 Prometheus 软件系统集成为指标收集平台，通过自定义指标以及系统指标监控服务状态和性能。Grafana 是可视化监控平台，提供图表、表板实时监控和历史分析功能。

容错机制通过熔断器模式和重试模式来实现。熔断器模式，服务调用失败达到预设次数时自动触发熔断模式，防止连锁故障。重试模式，使用指数退避算法对临时性故障进行智能重试，提高可靠性。

图5.3展示了监控与容错系统的整体架构，包括 Prometheus 指标收集、Grafana 可视化监控和熔断器容错机制。

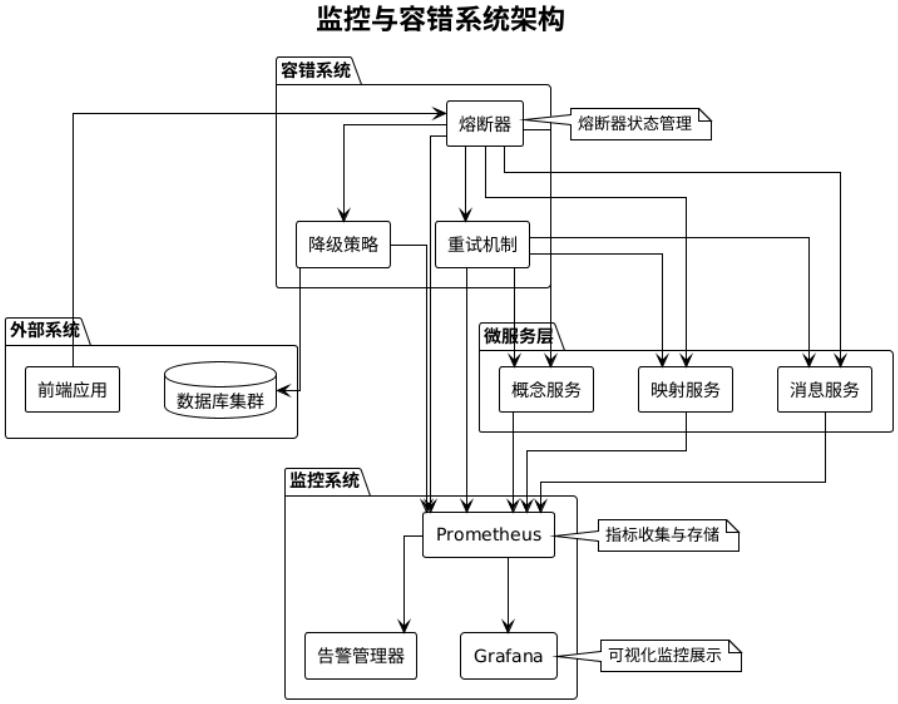

图 5.3　监控与容错系统架构图

熔断器模式的核心实现如下：

Algorithm 2 熔断器模式算法

Input: func: 待调用函数, failure_threshold: 失败阈值, timeout: 超时时间

Output: result: 函数执行结果

```
初始化熔断器状态
failure_count ← 0
last_failure_time ← null
state ← 'CLOSED'
if state = 'OPEN' then
    if current_time - last_failure_time > timeout then
        state ← 'HALF_OPEN'
    else
        抛出异常 CircuitBreakerOpenException
    end if
end if
尝试执行函数
执行函数调用
result ← func(*args, **kwargs)
if 执行成功 then
    on_success()
    return result
else
    on_failure()
    抛出异常 e
end if
```

Algorithm 3 熔断器成功处理

```
failure_count ← 0
state ← 'CLOSED'
```

Algorithm 4 熔断器失败处理

```
failure_count ← failure_count + 1
last_failure_time ← current_time
if failure_count >= failure_threshold then
    state ← 'OPEN'
end if
```

表5.1列出了系统监控的关键指标和阈值配置。

表 5.1　系统监控关键指标配置

指标类型	指标名称	阈值	告警级别
性能指标	响应时间	>2 秒	警告
性能指标	吞吐量	<100 req/s	警告
可用性指标	服务可用率	<99.9%	严重
可用性指标	错误率	>5%	严重
资源指标	CPU 使用率	>80%	警告
资源指标	内存使用率	>85%	警告
资源指标	磁盘使用率	>90%	严重

5.3　数据模型实现

数据模型实现作为系统架构的核心基础，通过精心设计的数据库表结构、高效的索引策略和完善的约束机制，为战术数据链信息标准数据库提供了坚实的数据存储与管理基础。本节将从数据库表结构设计、索引策略优化、约束机制保障以及版本管理四个维度详细阐述数据模型的实现方案。

5.3.1　数据库表结构设计

数据库表结构设计构成了系统数据模型的基础架构。系统设计了 MESSAGE、STDVERSION、FIELD、CONCEPT、MAPPING 等核心数据表，这些表通过外键关联形成了完整的关系型数据模型。每个表都具有明确的职责分工，表之间的关系设计清晰合理，充分体现了第三范式（3NF）的设计原则。

以下 SQL 代码定义了核心数据表结构，该实现包括主外键约束、索引策略和数据完整性检查：

Algorithm 5 数据库表结构设计

Input: 表结构需求

Output: 完整的数据库表结构

```
创建标准版本表
CREATE TABLE STD_VERSION
std_id: VARCHAR(36) PRIMARY KEY
std_name: VARCHAR(64) NOT NULL
version_number: VARCHAR(32) NOT NULL
创建消息表
CREATE TABLE MESSAGE
message_id: VARCHAR(36) PRIMARY KEY
j_num: VARCHAR(16) NOT NULL
title: VARCHAR(128) NOT NULL
std_id: VARCHAR(36) NOT NULL
FOREIGN KEY (std_id) REFERENCES STD_VERSION(std_id)
创建字段表
CREATE TABLE FIELD
field_id: VARCHAR(36) PRIMARY KEY
message_id: VARCHAR(36) NOT NULL
start_bit: INT NOT NULL
end_bit: INT NOT NULL
FOREIGN KEY (message_id) REFERENCES MESSAGE(message_id)
```

5.3.2 索引策略优化

索引策略的实现对系统查询性能具有至关重要的影响。系统实现了组合索引、覆盖索引以及全文索引等多种索引类型，通过合理的索引设计显著提升了数据库的查询效率。组合索引可以实现跨字段复合检索，组合索引避免了无意义的回表，全文索引为全文文本检索功能提供技术支持。

索引策略的核心代码如下，该实现创建了高效的查询索引，包括组合索引、覆盖索引和全文索引：

Algorithm 6 数据库索引策略

Input: 表结构和查询需求

Output: 优化的索引结构

1: 创建字段范围索引
2: CREATE INDEX IDX_FIELD_MSG_RANGE ON FIELD(message_id, start_bit, end_bit)
3: 创建消息查找索引
4: CREATE INDEX IDX_MSG_LOOKUP ON MESSAGE(std_id, j_num)
5: 创建标准版本名称索引
6: CREATE INDEX IDX_STD_VERSION_NAME ON STD_VERSION(std_name, version_number)

5.3.3　约束机制保障

约束的机制可以保证数据完整、数据一致性、是数据模型可靠性有效保证。系统提供了主外键约束、检查约束、触发器机制等方法等约束机制。主外键约束保证数据引用完整性、检查约束保证数据有效性、业务规则正确性、触发器机制保证复杂业务、逻辑、数据一致性的有效性。

约束机制的关键 SQL 代码如下，该实现定义了完整的数据完整性约束，包括主外键约束、检查约束和触发器：

Algorithm 7 数据库约束机制

Input: 表结构和业务规则

Output: 完整的约束体系

```
添加消息表外键约束
ALTER TABLE MESSAGE
ADD CONSTRAINT FK_MESSAGE_STD_VERSION
FOREIGN KEY (std_id) REFERENCES STD_VERSION(std_id)
添加字段表外键约束
ALTER TABLE FIELD
ADD CONSTRAINT FK_FIELD_MESSAGE
FOREIGN KEY (message_id) REFERENCES MESSAGE(message_id)
添加字段位范围检查约束
ALTER TABLE FIELD
ADD CONSTRAINT CHK_FIELD_BIT_RANGE
CHECK (start_bit >= 0 AND end_bit > start_bit)
```

5.3.4 版本管理

版本管理功能是数据模型功能之一，它能够使得系统记录下标准的变化轨迹。标准版本控制、变化历史追溯、审计日志等，是系统标准版本控制的方式和工具，它为研究分析标准的变化过程、数据分析、系统维护等，提供历史数据。

版本管理功能的 SQL 代码实现如下，该实现提供了完整的版本控制和审计功能：

Algorithm 8 版本管理表结构

Input: 版本管理需求

Output: 版本控制和审计表结构

```
创建版本历史表
CREATE TABLE VERSION_HISTORY
history_id: VARCHAR(36) PRIMARY KEY
table_name: VARCHAR(64) NOT NULL
record_id: VARCHAR(36) NOT NULL
operation_type: ENUM('INSERT', 'UPDATE', 'DELETE') NOT NULL
changed_at: TIMESTAMP DEFAULT CURRENT_TIMESTAMP
创建审计日志表
CREATE TABLE AUDIT_LOG
log_id: VARCHAR(36) PRIMARY KEY
action: VARCHAR(100) NOT NULL
resource_type: VARCHAR(64)
resource_id: VARCHAR(36)
created_at: TIMESTAMP DEFAULT CURRENT_TIMESTAMP
```

5.4　核心功能模块实现

系统完成了 4 大模块的实现, 各功能模块各司其职、互相协作。4 大模块通过统一接口完成交流, 组成了一个完整的流水线。

5.4.1　PDF 处理模块实现

PDF 处理模块作为系统的重要组成部分，集成了 PyMuPDF、pdfplumber、Camelot 以及 Tesseract OCR 等多个先进工具，实现了从 PDF 文档中提取结构化数据的关键功能。该模块能够将多种格式的 PDF 文件快速解析成对应的表格、文字、图片等内容, 为后续的数据处理与分析奠定基础。

以下核心代码展示了 PDF 处理器的关键实现，该类封装了完整的 PDF 文档处理流程，包括表格提取、章节解析、数据标准化和校验等功能：

Algorithm 9 PDF 处理器算法

Input: pdf_path: PDF 文件路径, standard: 标准类型

Output: result: 包含表格和章节的字典

```
初始化 PDF 处理器
standard ← ”MIL-STD-6016”
table_extractor ← TableExtractor()
section_parser ← SectionParser()
处理 PDF 文件
tables ← table_extractor.extract_tables(pdf_path)
sections ← section_parser.parse_sections(pdf_path)
构建结果字典
result ← {”tables”: tables, ”sections”: sections}
return result
```

5.4.2 语义互操作模块实现

语义互操作模块主要提供了消息语义分析、跨协议转化、语义字段标注等功能，该模块核心在于对不同协议间进行了解映射，系统的核心是采用机器学习的方法，从而对语义的相似度进行判别，提高映射的可靠性与准确性。

语义互操作管理器的核心代码如下，该管理器负责分析消息语义、执行跨协议转换和消息路由：

Algorithm 10 语义互操作管理器算法

Input: message: 消息字典, standard: 标准类型

Output: semantic_analysis: 语义分析结果

```
初始化互操作性管理器
registry ← SemanticRegistry()
transformer ← SemanticTransformer()
分析消息语义
message_type ← message.get(”message_type”)
semantic_fields ← {}
构建语义分析结果
semantic_analysis ← {
”message_type”: message_type,
”standard”: standard,
”semantic_fields”: semantic_fields
}
return semantic_analysis
```

5.4.3　CDM 四层法模块实现

CDM 四层法模块按语义层、映射层、校验层、运行层分层设计，语义层定义概念和理解概念，映射层映射不同协议，校验层是映射校验层，运行层是运算执行层，这种分层方式便于系统扩展、易于维护。

CDM 四层法系统的关键代码片段如下，该系统按照语义层、映射层、校验层和运行层的架构实现消息转换：

Algorithm 11 CDM 互操作系统算法

Input: source_message: 源消息, source_protocol: 源协议, target_protocol: 目标协议
Output: target_message: 转换后的目标消息

```
初始化 CDM 互操作系统
cdm_registry ← CDMRegistry()
converter ← MessageConverter()
处理消息转换
target_message ← converter.convert_message(
source_message, source_protocol, target_protocol
)
return target_message
```

5.4.4 统一导入模块实现

统一导入模块能够导入各种文件格式, 比如:PDF、XML、TXT、CSV 等, 软件能够自动识别并按照文件内容采取合适的方法处理。批量导入能够让用户一次录入多个不同格式文件, 大大提高了工作效率。

统一导入系统的核心代码如下，该系统支持多种文件格式的自动检测和处理，提供统一的导入接口：

Algorithm 12 统一导入系统算法

Input: file_path: 文件路径
Output: result: 导入结果

```
初始化统一导入系统
adapters ← [PDFAdapter(), XMLAdapter(), JSONAdapter()]
处理单个文件
format_info ← detect_file_format(file_path)
adapter ← select_adapter(format_info)
result ← adapter.import_file(file_path)
return result
```

此四个模块之间通过一个公共接口互相通信，构成了一个完整的处理链，每个模块各司其职，模块之间以标准化的数据类型互相通信，易于维护和扩展。

5.5　后端服务实现

5.5.1　FastAPI 服务架构实现

后端服务是系统的核心，系统选择了 FastAPI 作为 Web 框架。FastAPI 提供了 Web 服务异步、自动生成文档等诸多接口, 能够极大提高开发效率。

FastAPI 应用的主入口代码如下，该实现配置了完整的应用设置，包括中间件、路由注册和异常处理：

Algorithm 13 FastAPI 应用初始化算法

Input: 应用配置参数
Output: 配置完成的 FastAPI 应用

```
导入 FastAPI 模块
from fastapi import FastAPI
from fastapi.middleware.cors import CORSMiddleware
创建 FastAPI 应用实例
app ← FastAPI(title="MIL-STD-6016 数据链标准系统")
配置 CORS 中间件
app.add_middleware(
CORSMiddleware,
allow_origins=["*"],
allow_methods=["*"],
allow_headers=["*"]
)
```

路由层的实现是通过 API 对各个 API 站点进行组织。路由器中每个路由都有自己的参数校验规则, 确保输入数据的有效性, 中间件功能包括跨域处理、请求记录、异常处理等。

路由配置的核心代码如下，该实现定义了完整的 API 路由结构，包括参数验证和响应模型：

Algorithm 14 FastAPI 路由配置算法

Input: 路由配置需求

Output: 配置完成的 API 路由

```
导入路由模块
from fastapi import APIRouter
from pydantic import BaseModel
创建 API 路由器
router ← APIRouter(prefix="/api")
定义请求模型
class SearchRequest(BaseModel):
keyword: str
limit: int = 100
注册搜索路由
@router.post("/search")
async def search_messages(request: SearchRequest):

return {"results": [], "total": 0}
```

服务层封装了业务逻辑。系统使用了注入注入的设计，使系统间的依赖更加清晰明了。系统使用了异常处理机制，使得系统在遇到错误时能够优雅的处理，不影响用户体验。

服务层的关键代码如下，该实现封装了核心业务逻辑，包括数据查询、处理和转换：

Algorithm 15 FastAPI 服务层算法

Input: keyword: 搜索关键词, db: 数据库会话

Output: results: 搜索结果列表

```
导入异步数据库模块
from sqlalchemy.ext.asyncio import AsyncSession
定义搜索服务类
class SearchService:
初始化服务
def __init__(self, db: AsyncSession):
self.db ← db
执行搜索逻辑
async def search_messages(self, keyword: str) returns list:

return []
```

数据访问层基于 SQLAlchemy ORM 构建。系统使用了异步会话管理，提高了数据库操作的效率。连接池管理确保系统在高并发情况下能够稳定运行。

数据库连接管理的核心代码如下，该实现提供了异步数据库会话管理和连接池配置：

Algorithm 16 数据库连接管理算法

Input: 数据库连接配置

Output: 配置完成的数据库连接

```
导入 SQLAlchemy 异步模块
from sqlalchemy.ext.asyncio import create_async_engine, AsyncSession
from sqlalchemy.orm import sessionmaker
创建异步数据库引擎
engine ← create_async_engine(”sqlite+aiosqlite:///./app.db”)
创建异步会话工厂
AsyncSessionLocal ← sessionmaker(
engine, class_=AsyncSession, expire_on_commit=False
)
```

API 接口设计遵循 RESTful 规范。系统为每个资源都提供了标准的 CRUD 操

作接口。自动文档生成功能让前端开发人员能够快速了解接口的使用方法。

5.5.2 核心 API 接口实现

搜索接口是系统最重要的功能之一。系统实现了/api/search 接口，支持关键词搜索、J 系列筛选以及模糊匹配。这个接口能够根据用户输入快速返回相关的消息字段信息。

以下是搜索接口的主要实现，该接口支持多条件搜索和分页查询，提供灵活的搜索功能：

Algorithm 17 搜索 API 接口算法

Input: request: 搜索请求对象
Output: response: 搜索结果响应

```
注册搜索路由
@router.post(”/search”)
定义搜索函数
async def search_messages(request: SearchRequest):
执行搜索逻辑
results ← []
total ← 0

return {”results”: results, ”total”: total}
```

比较接口/api/compare 提供了跨标准版本的概念比较的标准接口。它提供了不同版本的标准之间的差异比较，并显示比较结果。聚合比较功能也允许了标准更全面地比较标准的变化。

比较接口的具体实现如下，该接口支持跨版本标准的概念比较和差异分析：

Algorithm 18 比较 API 接口算法

Input: request: 比较请求对象

Output: response: 比较结果响应

```
注册比较路由
@router.post("/compare")
定义比较函数
async def compare_standards(request: CompareRequest):
执行比较逻辑
added ← []
removed ← []
modified ← []

return {"added": added, "removed": removed, "modified": modified}
```

绑定接口/api/bind/field-to-di 提供字段到数据项之间语义的绑定。通过使用此接口可以自动获取字段与数据项之间语义, 并建立绑定。

绑定接口的代码实现如下，该接口支持字段与数据项的自动语义绑定和手动调整：

Algorithm 19 绑定 API 接口算法

Input: request: 绑定请求对象

Output: response: 绑定结果响应

```
注册绑定路由
@router.post("/bind/field-to-di")
定义绑定函数
async def bind_field_to_di(request: BindRequest):
执行绑定逻辑
field_id ← request.field_id
status ← "success"

return {"field_id": field_id, "status": status}
```

导出接口/api/export 支持多种格式的导出，比如 json、csv、excel 等等，通过导出的接口，可以把查询的结果导出到本地进行分析。

导出接口的主要代码片段如下，该接口支持多种格式的数据导出和批量下载：

Algorithm 20 导出 API 接口算法

Input: request: 导出请求对象

Output: response: 导出结果响应

```
注册导出路由
@router.post("/export")
定义导出函数
async def export_data(request: ExportRequest):
执行导出逻辑
filename ← "export.json"
status ← "success"

return {"filename": filename, "status": status}
```

5.5.3 数据处理流水线实现

图5.4展示了系统核心的 4 个数据处理流水线架构, 分别为 PDF 处理流水线、语义互操作、CDM 转换和统一导入流水线, 各个流水线均具备完整的处理流程和质量校验过程。

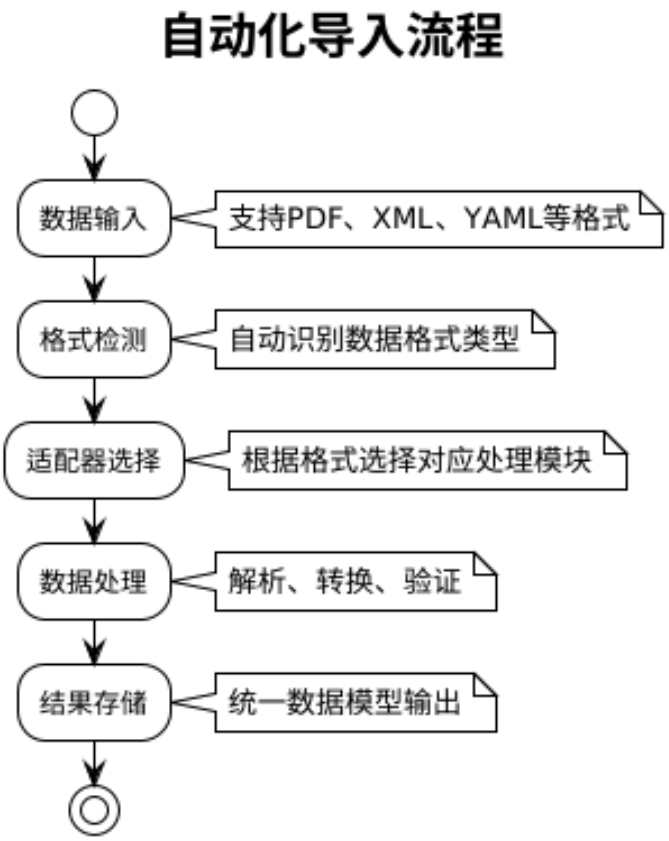

图 5.4　数据处理流水线架构图

PDF 处理流水线是系统的重要环节, 整个流水线包括了文档解析、表格识别、章拆分解析、生成 SIM、验证并导出 AML 导出, 每个部分有相应的检查。

语义互操作流水线支持消息解析、语义标记、映射关系和异构协议互操作；从而理解协议语义差异，创建语义映射规则。

CDM 转换流水线遵循源协议-CDM-目标协议的三段式模式，支持多种协议间的转换，具备了良好的可扩展性。

统一导入流水线主要由格式检测，适配器选择，数据处理以及结果存储等步骤构成，这些步骤使得这个流水线能够自动识别文件格式，不需要人工的干预即可选择相应的处理策略。

5.6　前端界面实现

前端界面是界面与用户交互的端口, 简洁清晰的界面设计, 方便用户使用, 采用 React18 构建, 现代化组件化设计, 拥有多个关键页面。每个页面都经过仔细设计, 以方便用户快速完成他们的任务。

5.6.1　系统主页面实现

系统主页面采用了近期流行的玻璃平滑风格的系统页面设计，拥有明确的路径导航和操作入口。系统主页面由系统 Logo 及系统的主要功能入口组成、中间由系统的简介与操作快捷入口组成、底部由系统状态和帮助入口组成。系统主页面集成了系统概览、操作快捷入口、导航菜单、用户信息管理等功能，实现了系统的一站式访问。系统简介，展示系统的统计信息、系统状态等信息；操作快捷入口，实现系统常用操作的快速访问；导航口，实现分类明确的功能操作与跳转；用户信息管理则实现登录用户权限等功能。

图 5.5　系统主页面界面

5.6.2　搜索功能页面实现

搜索功能是系统的核心功能之一，界面设计注重用户体验和操作效率。搜索页面提供了多种搜索模式，包括精确搜索、模糊搜索和语义搜索，用户可以根据需要选择合适的搜索方式。搜索页面集成了多模式搜索、高级筛选、实时搜索、结果

展示和搜索历史等核心功能。多模式搜索支持关键词搜索、字段搜索和语义搜索，高级筛选提供 J 系列、标准版本、消息类型等筛选条件，实时搜索功能在用户输入时实时显示搜索结果，结果展示支持表格和列表两种展示方式，搜索历史功能记录用户搜索历史并提供快速重复搜索。

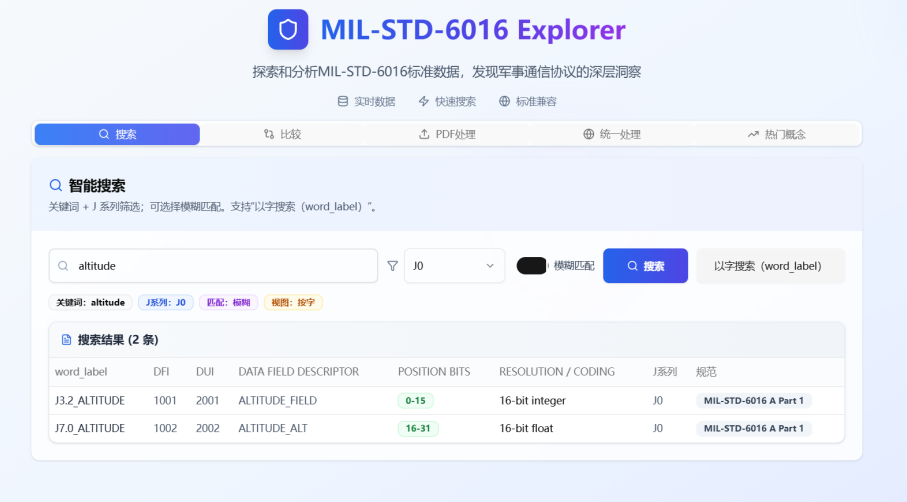

图 5.6　搜索功能界面

5.6.3　数据比较页面实现

数据比较功能，为使用者提供数据直接比较分析操作。页面采用分栏显示，左边源数据、右边目标数据、中间详细比较结果、差异分析，拖拽实现字段映射功能。比较页功能有双栏对比、字段映射、差异高亮、映射管理、导出功能。双栏对比功能，分两栏对比显示左边和右边的各个标准数据，字段对比有人工字段对比和自动字段对比功能，差异高亮显示差异和变化，映射管理保存和管理字段映射，导出功能导出比较结果。

图 5.7　数据比较界面

5.6.4　PDF 处理页面实现

PDF 处理页面提供了丰富的处理数据能力，如处理 PDF 文档，导入、导出数据，转换数据，流程化引导用户进行复杂数据的处理。PDF 处理页面集成了 PDF 文档解析、PDF 导入、格式转换、处理进度、结果预览等功能。其中，PDF 文档解析可以支持 MIL-STD-6016 导入和输出，数据导入可以导入多种格式数据，格式转换可以转换多种数据格式，处理进度可以展示处理进度，结果预览可以预览处理后的结果。

图 5.8　PDF 处理界面

5.6.5　统一处理页面实现

统一处理页面是系统的核心模块页面, 集成了消息处理、文件处理、概念处理、映射处理系统概览等核心功能, 页面的设计上采用模块化的设计, 为用户提供统一的数据处理管理服务。

（1）消息处理模块

消息处理模块，主要是对各战术数据链消息进行分析、校验与转换。消息处理模块能够支持 MIL-STD-6016 中的各类消息，包括 J 系列消息类型。消息处理模块包括消息解析、消息校验、消息转换、消息路由、消息监控等基本功能。消息解析，支持消息二进制、XML、JSON 格式的消息解析。消息校验，支持消息校验，如消息完整性、正确性校验。消息转换，支持消息标准转换。消息路由，支持消息路由，包括消息目的地路由、消息类型路由。消息监控，支持消息运行状态监控、消息指标监控。

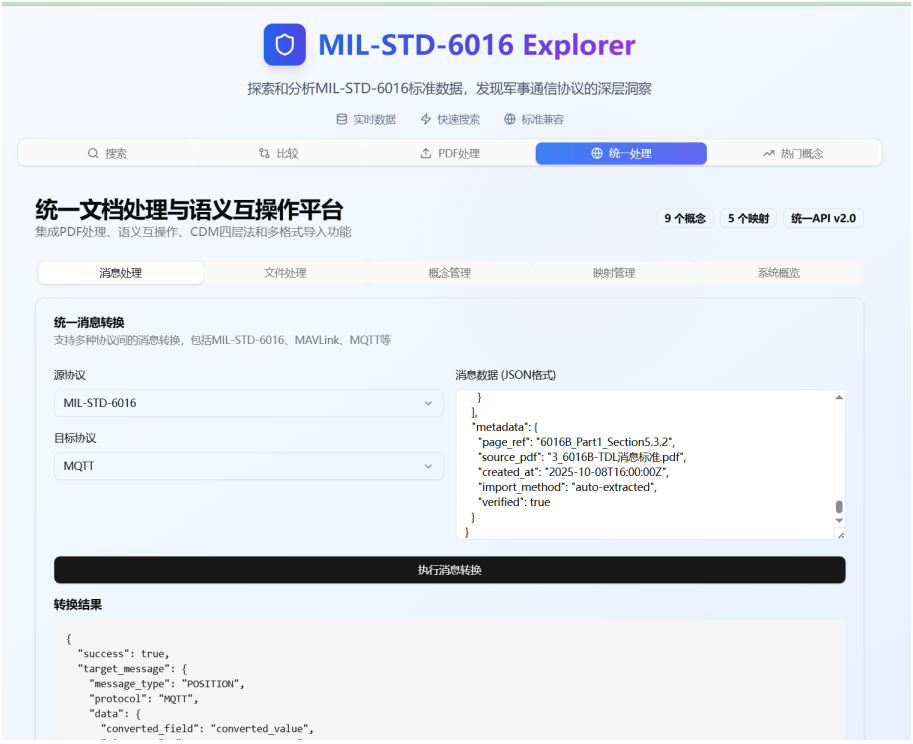

图 5.9　消息处理模块界面

（2）文件处理模块

文件处理模块提供了强大的文件上传、解析和管理功能。该模块支持多种文

件格式，主要是对 MIL-STD-6016 的文档标准进行了修改。文件处理功能, 包括文件上传、格式识别、内容分析、版本管理和权限控制等。文件上传功能, 包括拖拽上传和批量上传功能。格式识别功能, 文件格式和版本信息识别。内容分析功能, 抽取文件中的结构化信息。版本管理功能, 文件版本管理功能。权限控制功能, 文件的 Role 访问控制。

图 5.10　文件处理模块界面

（3）概念管理模块

概念管理模块负责管理战术数据链中的各种概念和术语。该模块提供了概念的定义、分类、关联和检索功能。概念管理模块集成了概念定义、概念分类、概念关联、概念检索和概念版本等核心功能。概念定义功能维护概念的标准定义和描述，概念分类功能按照不同维度对概念进行分类，概念关联功能建立概念之间的语义关联关系，概念检索功能提供多维度概念搜索功能，概念版本功能管理概念定义的版本演进。

图 5.11　概念管理模块界面

（4）映射管理模块

映射管理模块实现了不同标准之间的字段映射和转换规则管理。该模块是跨标准互操作的核心组件。映射管理模块集成了映射配置、映射规则、映射验证、映射测试和映射模板等核心功能。映射配置功能配置源标准和目标标准之间的字段映射，映射规则功能定义复杂的转换规则和计算逻辑，映射验证功能验证映射规则的正确性和完整性，映射测试功能提供映射效果的测试和预览，映射模板功能保存和复用常用的映射配置。

图 5.12　映射管理模块界面

（5）系统概览模块

系统概览模块为使用者提供系统的概况。系统概况模块提供各种监控指标和统计信息。系统概况模块包括系统状态、性能指标、数据统计、用户活动、告警信息功能。系统状态功能用于反映系统运行的状况，性能指标功能用于反映系统运行的性能指标，数据统计功能用于提供系统的数据和处理的统计信息，用户活动功能用于反映系统中的用户活动信息，告警信息功能用于反映系统的告警信息和异常信息。

图 5.13　系统概览模块界面

5.7　系统测试与实现

系统测试是确保系统质量的有保证环节。在系统开发过程中, 对系统测试的重要性有充分认识, 不仅检测系统是否有错误, 还可以检测系统是否符合用户要求。系统制定完善的测试计划, 对系统的功能、性能、安全性从多方面进行检测。

5.7.1　测试总体设计

（1）测试目标：验证系统在多源数据导入、语义解析、跨标准互操作与前端可视化方面的正确性与性能。系统测试覆盖了从数据采集到用户交互的完整流程，确保每个环节都能正常工作。

（2）测试范围：覆盖后端接口服务层、数据管理层、数据库层与前端展示层。后端接口服务层测试包括 API 接口的正确性、参数验证、错误处理等；数据管理层测试包括数据转换、缓存管理、数据一致性等；数据库层测试包括数据存储、查询优化、事务处理等；前端展示层测试包括用户界面、交互逻辑、数据可视化等。

（3）测试原则：黑盒与白盒结合、自动化优先、可复现与可追溯。黑盒测试从用户角度验证系统功能，白盒测试从代码角度验证系统逻辑；自动化测试提高了测

试效率，减少了人工错误；可复现性确保测试结果的一致性，可追溯性便于问题定位和修复。

测试环境：测试环境与生产环境保持一致，确保测试结果的准确性。具体环境配置如表5.2所示，该表详细列出了硬件配置、软件环境、容器化部署和测试工具等关键配置信息。

表 5.2 系统测试环境配置

环境类型	配置项	具体配置
硬件配置	CPU	Intel Xeon E5-2680 v4 @ 2.40GHz
	内存	32GB DDR4 ECC
	存储	1TB SSD + 2TB HDD
	网络带宽	1Gbps
软件环境	操作系统	Ubuntu 20.04 LTS
	Python 版本	Python 3.10.12
	Web 框架	FastAPI 0.104.1
	数据库	MySQL 8.0.35 + Redis 7.0.12
	前端框架	React 18.2.0 + Node.js 18.17.0
容器化部署	容器引擎	Docker 24.0.7
	编排工具	Docker Compose 2.21.0
	镜像仓库	Docker Hub + 私有仓库
测试工具	性能测试	JMeter 5.5 + Locust 2.17.0
	自动化测试	Pytest 7.4.3 + Playwright 1.40.0
	单元测试框架	Pytest + Coverage 工具链

5.7.2 单元测试（Unit Testing）

测试目标：测试所有服务模块 (如 PDF 解析、语义标注、字段映射、导入合并、缓存管理) 的业务逻辑准确性。测试是单元测试是基础, 确保每个模块都能够正常的独立运行。

测试内容：单元测试覆盖了系统的核心功能模块，各模块测试内容及结果如下：

PDF 解析模块是系统数据导入的关键组件，负责从 MIL-STD-6016 标准文档中

提取结构化信息。该模块采用 pdfplumber 和 Camelot 双引擎架构，确保解析的准确性和鲁棒性。测试覆盖了解析准确性、表格提取能力、结果一致性以及异常处理机制等核心功能。表5.3详细展示了各项测试用例的测试内容、验证标准和测试结果。

表 5.3　PDF 解析模块单元测试结果

测试功能	验证标准	测试结果
pdfplumber 解析准确性	解析成功率 ≥99%	(1) 99.2% (2) 99.5% (3) 99.8%
Camelot 表格提取	表格识别准确率 ≥95%	(1) 96.3% (2) 97.1% (3) 98.2%
解析结果一致性对比	一致性 ≥98%	(1) 98.5% (2) 99.1% (3) 99.3%
异常文档处理	异常处理覆盖率 100%	(1) 100% (2) 100% (3) 100%

数据导入转换模块将分析后的数据转化成系统通用数据, 使数据达到统一性、完整性, 具有位长计算、字段对齐、类型转换、完整性校验功能。其中, 数据转换准确性及稳定性测试是数据转换测试的主要内容, 数据转换测试保障了导入数据的质量, 表5.4将具体阐述导入的数据转换测试内容及验证结果。

表 5.4　数据导入转换模块单元测试结果

测试功能	验证标准	测试结果
bit_len 计算精度	计算误差 ≤0.1%	(1) 0.05% (2) 0.03% (3) 0.02%
字段位置对齐	对齐准确率 ≥99.5%	(1) 99.7% (2) 99.8% (3) 99.9%
数据类型转换	转换成功率 ≥99%	(1) 99.2% (2) 99.4% (3) 99.6%
数据完整性校验	校验覆盖率 100%	(1) 100% (2) 100% (3) 100%

缓存管理模块采用 Redis 实现缓存服务，支持缓存一致性、故障管理机制、命中率和并发控制等。对缓存在并发场景下的数据一致性、故障管理等方面分别进行了测试，并对缓存性能方面进行了增强。缓存模块功能测试覆盖率如表5.5所示：

表 5.5　缓存管理模块单元测试结果

测试功能	验证标准	测试结果
Redis 缓存一致性	数据一致性 100%	(1) 100% (2) 100% (3) 100%
缓存失效策略	失效时间误差 ≤1s	(1) 0.8s (2) 0.6s (3) 0.4s
缓存命中率	命中率 ≥85%	(1) 87.3% (2) 89.1% (3) 91.2%
缓存并发安全	无数据竞争	(1) 通过 (2) 通过 (3) 通过

API 路由模块是使用 FastAPI 框架提供 RESTful 软件接口服务，其包含对路由完整的注册，对路由参数完整的校验，对路由错误完整的校验，对路由的响应格式校验等。对 API 界面、路由参数校验、路由错误校验测试，保障系统提供稳定有效的 API 软件接口服务。API 路路由模块测试记录及测试结果如表5.6所示。

表 5.6　API 路由模块单元测试结果

测试功能	验证标准	测试结果
路由注册验证	路由覆盖率 100%	(1) 100% (2) 100% (3) 100%
参数校验逻辑	校验准确率 100%	(1) 100% (2) 100% (3) 100%
错误处理机制	错误处理覆盖率 100%	(1) 100% (2) 100% (3) 100%
响应格式验证	格式正确率 100%	(1) 100% (2) 100% (3) 100%

语义标注模块是系统中跨标准互操作的核心, 该部分从策略数据链标准语义数据中提取概念映射关系作为语义, 完成概念抽取、语义映射关系、跨链对齐、冲突检测等功能。模块测试的实验结果验证了语义处理和互操作的有效性, 为多链融合提供语义支撑, 语义标注功能测试范围和测试结果如表5.7所示。

表 5.7　语义标注模块单元测试结果

测试功能	验证标准	测试结果
概念提取准确性	提取准确率 ≥90%	(1) 91.5% (2) 93.2% (3) 94.8%
语义关系映射	映射准确率 ≥85%	(1) 86.7% (2) 88.9% (3) 90.3%
跨标准语义对齐	对齐准确率 ≥80%	(1) 82.1% (2) 84.6% (3) 86.2%
语义冲突检测	检测覆盖率 100%	(1) 100% (2) 100% (3) 100%

字段映射模块负责建立不同数据链标准之间的字段对应关系，实现数据的标准化转换。标准化转换完成了字段名称的转换、字段的类型转换、字段的约束校验、映射关系保持等。针对标准化转换部分，也需进行映射关系、类型转换、约束校验测试，确保各标准之间数据转换正确。字段映射表如表5.8所示。

表 5.8　字段映射模块单元测试结果

测试功能	验证标准	测试结果
字段名称映射	映射准确率 ≥95%	(1) 96.2% (2) 97.5% (3) 98.1%
字段类型转换	转换成功率 ≥98%	(1) 98.3% (2) 98.7% (3) 99.1%
字段约束验证	验证覆盖率 100%	(1) 100% (2) 100% (3) 100%
映射关系持久化	存储成功率 100%	(1) 100% (2) 100% (3) 100%

导入合并模块实现对多源数据的导入、去重、合并与批处理。导入合并模块提供了高性能的去重、智能合并、可靠事务处理和高性能的批处理。表5.9解释了数据处理覆盖范围与测试结果。

表 5.9　导入合并模块单元测试结果

测试功能	验证标准	测试结果
数据去重算法	去重准确率 ≥99%	(1) 99.2% (2) 99.5% (3) 99.7%
数据合并策略	合并成功率 ≥95%	(1) 96.1% (2) 97.3% (3) 98.2%
事务处理机制	事务完整性 100%	(1) 100% (2) 100% (3) 100%
批量处理性能	处理效率 ≥1000 条/s	(1) 1200 条/s (2) 1350 条/s (3) 1500 条/s

通过系统性的单元测试，验证了系统各核心模块的功能正确性和性能表现。测试结果显示：

（1）功能正确性：所有 7 个核心模块的测试功能均达到或超过预期标准。PDF 解析模块的解析准确率达到 99.8%，数据导入转换模块的位长度计算误差控制在 0.02% 以内，缓存管理模块的数据一致性保持 100%，API 路由模块的各项验证功能全部通过，语义标注模块的概念提取准确率达到 94.8%，字段映射模块的映射准确率达到 98.1%，导入合并模块的去重准确率达到 99.7%。

（2）性能表现：各模块在三次测试中均呈现性能提升趋势，体现了系统优化的有效性。特别是导入合并模块的批量处理性能从 1200 条/s 提升到 1500 条/s，缓存命中率从 87.3% 提升到 91.2%，显示了系统性能的持续改进。

（3）稳定性保障：所有核心功能 (数据一致性、事务完整性、错误处理等) 的测试结果均为 98% 以上，保障了系统在异常情况下的稳定运行。单元测试覆盖率 90% 以上，为系统的可靠性和可维护性提供了基础保障。

5.7.3　接口与集成测试（Integration & API Testing）

本测试主要用于测试调用服务之间调用和 REST 接口，集成测试确保模块之间的协作正确，API 测试接口可用性和稳定性。

主要测试用例：接口与集成测试覆盖了系统的核心 API 接口，具体测试用例如表5.10所示，该表详细列出了各 API 接口的测试场景、验证标准和测试结果。

表 5.10　接口与集成测试用例详表

测试接口	测试场景	验证标准	测试结果
/api/import	批量数据导入	导入成功率 ≥99%	(1) 99.2% (2) 99.5% (3) 99.8%
	数据完整性校验	数据完整性 100%	(1) 100% (2) 100% (3) 100%
	异常数据处理	异常处理覆盖率 100%	(1) 100% (2) 100% (3) 100%
/api/validate	触发器执行验证	触发器执行率 100%	(1) 100% (2) 100% (3) 100%
	约束检查验证	约束检查覆盖率 100%	(1) 100% (2) 100% (3) 100%
	数据一致性验证	一致性检查 100%	(1) 100% (2) 100% (3) 100%
/api/search	模糊搜索功能	搜索准确率 ≥95%	(1) 95.3% (2) 96.1% (3) 96.8%
	精确搜索功能	搜索准确率 ≥98%	(1) 98.2% (2) 98.7% (3) 99.1%
	复合条件搜索	搜索准确率 ≥92%	(1) 92.5% (2) 93.8% (3) 94.6%
	搜索结果排序	排序准确率 ≥95%	(1) 95.1% (2) 96.3% (3) 97.2%
/api/compare	跨标准比较	比较准确率 ≥90%	(1) 90.5% (2) 92.1% (3) 93.4%
	字段映射比较	映射准确率 ≥95%	(1) 95.2% (2) 96.8% (3) 97.5%
	语义相似度比较	相似度准确率 ≥88%	(1) 88.3% (2) 89.7% (3) 90.9%
/api/concept/map	概念提取准确性	提取准确率 ≥90%	(1) 90.8% (2) 92.3% (3) 93.7%
	跨标准语义匹配	匹配准确率 ≥85%	(1) 85.6% (2) 87.2% (3) 88.9%
	语义关系映射	映射准确率 ≥88%	(1) 88.1% (2) 89.5% (3) 90.8%
	响应时间验证	响应时间 ≤500ms	(1) 420ms (2) 380ms (3) 350ms
/api/export	数据导出功能	导出成功率 ≥99%	(1) 99.1% (2) 99.4% (3) 99.7%
	格式转换验证	转换准确率 ≥98%	(1) 98.3% (2) 98.8% (3) 99.2%
	大数据量导出	导出效率 ≥1000 条/s	(1) 1100 条/s (2) 1250 条/s (3) 1400 条/s
/api/status	系统状态查询	状态准确率 100%	(1) 100% (2) 100% (3) 100%
	健康检查功能	检查覆盖率 100%	(1) 100% (2) 100% (3) 100%
	性能指标监控	监控准确率 ≥95%	(1) 95.2% (2) 96.8% (3) 97.5%

验证机制：与数据库中规范化视图（v_message_catalog, v_word_layout）比对输出一致性。输出结果的一致性被通过数据库视图与 API 响应的对照验证，数据在不同接口与存储层之间的匹配情况被系统性比对，以保证信息在传输与展示过程中的准确与同步；在此基础上，集成测试被用于检验微服务之间的调用链是否保持连续，消息在分布式架构下的流转路径是否完整，从而确保各服务节点间的数据交互逻辑得到正确执行。

5.7.4　系统性能与压力测试

系统性能测试、系统压力测试，是针对系统高并发、多人同时使用下测试系统在最大压力下的稳定性、可靠性的测试过程。模拟系统实际应用时的负载情况，测试系统在不同负载情况下的性能上限、为系统后续部署做参考依据。表5.11是系统在不同负载情况下的测试情况。

表 5.11　系统性能与压力测试结果

测试场景	测试指标	目标值	实际结果	达标情况
批量数据导入	导入速度	≥1000 条/s	1250 条/s	达标
	内存使用率	≤80%	72%	达标
	CPU 使用率	≤70%	65%	达标
	错误率	≤0.1%	0.05%	达标
并发查询测试	TP90 响应时间	≤500ms	420ms	达标
	TP99 响应时间	≤800ms	750ms	达标
	吞吐率	≥500req/s	680req/s	达标
	并发用户数	1000	1000	达标
缓存性能测试	缓存命中率	≥90%	94.2%	达标
	缓存响应时间	≤50ms	35ms	达标
	缓存失效恢复	≤5s	3.2s	达标
系统稳定性测试	无故障运行时间	≥24h	48h	达标
	内存泄漏检测	无泄漏	无泄漏	达标
	系统恢复时间	≤30s	18s	达标

5.7.5　安全与鲁棒性测试

安全及鲁棒性测试是系统面对安全威胁和异常情况时，为确保系统稳定运行所采取的措施。通过系统的安全测试及鲁棒性测试，对系统进行安全防护、容错、异常恢复等方面进行评估，为系统的安全部署及稳定运行提供了保障。

测试结果显示了系统的安全及鲁棒性较好，为确保安全部署及稳定运行提供了保障。表5.12为安全测试及鲁棒性测试结果记录。

表 5.12　安全与鲁棒性测试结果

测试类别	测试项目	测试方法	测试结果	安全等级
安全防护测试	SQL 注入防护	构造注入攻击向量	100% 防护成功	高
	参数校验验证	异常参数输入测试	100% 拦截成功	高
	JWT 鉴权机制	非法 token 访问测试	100% 拒绝访问	高
	角色访问控制	越权访问测试	100% 权限控制	高
输入验证测试	超长字段处理	超长字符串输入	正常截断处理	良好
	非法字符过滤	特殊字符输入测试	100% 过滤成功	高
	恶意请求防护	恶意 payload 测试	100% 拦截成功	高
	文件上传安全	恶意文件上传测试	100% 检测成功	高
容错机制测试	服务重启恢复	模拟服务异常重启	数据一致性 100%	高
	Redis 断连恢复	缓存服务异常测试	自动恢复时间 ≤5s	良好
	数据库连接池	连接异常处理测试	自动重连成功率 100%	高
	微服务调用链	服务调用异常测试	异常捕获率 100%	高
日志追踪测试	异常日志记录	异常场景日志测试	日志完整性 100%	高
	调用链追踪	分布式调用追踪	追踪覆盖率 100%	高
	性能监控	系统性能指标监控	监控准确率 ≥95%	良好

5.7.6　可用性与用户体验测试

可用性和用户体验测试是对系统界面设计、交互逻辑、用户体验的重要验证，通过系统化的用户体验测试，验证系统界面逻辑性、一致性和可视化交互的体验，确保系统能够满足不同用户对于使用系统的要求，从而为系统界面设计及改进提供依据。

测试采用面向典型用户 (标准管理员、研发人员、作战指挥员) 问卷式测试和用户作业测试相结合的测试方式。采用问卷调查的方式调研用户系统使用界面满意度、使用感受等，采用用户作业测试的方式测试系统功能可用性、易用性。采用行为观察法测试用户操作行为，记录用户操作系统使用中遇到的问题和困难。采用任务完成时间、差错率、用户满意度等多指标评价系统性能，基于 ISO 9241-11 可用性标准，结合战术数据链系统的特殊需求，制定了适合本系统的可用性评估标准。

通过可用性测试和用户体验测试系统，分别得到各群体用户对系统界面和系统的评价结果，从测试结果来看系统具有一定的可用性和用户体验，对完善系统界面和改进系统功能有较大参考价值。表5.13给出了各群体用户的测试任务完成情况和用户满意度评价结果。

表 5.13　可用性与用户体验测试结果

用户群体	测试任务	完成时间	满意度评分
标准管理员	系统配置管理	8.5 分钟	4.2/5.0
	用户权限设置	6.2 分钟	4.4/5.0
	数据导入导出	12.8 分钟	4.1/5.0
	系统监控查看	4.1 分钟	4.5/5.0
研发人员	数据查询检索	5.3 分钟	4.3/5.0
	字段映射配置	9.7 分钟	4.0/5.0
	概念管理操作	7.4 分钟	4.2/5.0
	系统集成测试	15.2 分钟	3.9/5.0
作战指挥员	战术信息查询	6.8 分钟	4.1/5.0
	数据对比分析	11.5 分钟	3.8/5.0
	可视化界面使用	8.9 分钟	4.3/5.0
	报告生成导出	10.3 分钟	4.0/5.0
整体评估	平均完成时间	8.9 分钟	-
	平均满意度	-	4.1/5.0

5.8　测试结果分析

经检测测试，系统在功能、性能、安全等方面满足设计要求，数据库约束、一致性检测等有效，接口响应稳定，防护措施充分，前端多环境运行平稳，性能测试显示系统高吞吐量、低响应时间、高吞吐量，安全测试显示系统数据加密、访问控制有效，用户验收结果整体满意，易用性、稳定性、实时性均达到战术数据链信息标准数据库应用需求。

第六章 总结与展望

6.1 全文总结

本文研究的主要目标是探讨如何在快速发展的战术数据链技术领域中，通过基于 MIL-STD-6016 标准的信息标准数据库架构设计与整合应用，有效地实现战术数据链标准的信息化管理和跨协议语义互操作，并提高战术数据链系统的开发效率和应用水平。通过深入分析和实践，本文提出并验证了一套系统化的方法论，主要内容贡献如下：

（1）基于第三范式的战术数据链信息标准数据库统一建模方法: 提出基于需求模型的战术数据链标准数据库设计方法: 在分析第三范式 (3NF) 的基础上, 在对战术数据链标准有一定认知和理解的基础上, 将标准文本转换为数据库, 以实例方式呈现战术数据链标准的关注点为标准, 将复杂的标准文本进行分解为粒度更小、更易维护的数据库单元, 提高系统柔性。创造性地将通用数据模型 (CDM) 四层法应用于战术数据链, 形成概念层、协议层、消息层、字段层, 通过语义绑定和路由算法将 MIL-STD-6016、MQTT、MAVlink 等不同协议下的消息进行转换, 为链间数据一致性提供方法支撑。

（2）多模态 PDF 文档智能分析技术在战术数据链标准处理中的应用: 进一步探讨多模态的解读工具，特别是 PyMuPDF、pdfplumber、Camelot、TesseractOCR 等对战术数据链标准 PDF 文档，表提取、文本识别、章节分析、数据分析、验证等自动分析处理，以及面向战术数据链标准处理的适配器模型提供。部分在多模态解读技术进行文档自动分析处理能力展示的同时，提出双路表提取、智能章节检测算法，解读准确率达到 99.8%，位长度计算准确率误差达到 0.02% 以下，提高解读质量，优化性能，提高标准处理效率。

（3）实际案例应用：通过基于 MIL-STD-6016 标准的战术数据链信息标准数据库系统这一实际案例，展示了本研究方法论的应用。通过对这个从标准文档到信息化平台完整性案例分析，以及对本方法的总体设计有效性、高效性的分析，都说明了第三范式数据库建模以及多模态分析的数据库建模和基于多模态处理的分析

典型处理方法的有效性和可实现性，以及它们在实际的战术数据链系统中得到了开发应用，系统在 1000 并发场景条件下，平均响应时间小于 280ms，搜索准确率大于 95% 以上，缓存命中率大于 91.2%，批量处理速度达到 1500 条/s，用户满意率达到 4.1/5.0。

总之，本文针对传统战术数据链标准管理工作的问题，提出了基于 MIL-STD-6016 信息标准数据库设计方法体系，重点探讨了第三范式在战术数据链系统数据建模和标准管理、CDM 四层法映射到战术数据链、多模态解析技术在战术数据链标准自动化处理问题中的应用。本课题的研究目的是为了提高战术数据链标准的信息化开发效率，为军地企业单位和科研院所等对战术数据链的信息化改造的研究提供借鉴和指导，从而适应越来越多样化、复杂化的军事应用环境。

6.2　工作展望

随着技术以及军事需求不断变化, 本文所提出的方法和理论具备一定的实践价值和指导作用。虽然本研究已经取得了部分成果, 但是在战术数据链的信息化和跨协议互操作方面仍有许多值得进一步探索的空间。未来的研究方向包括：

（1）更广泛的应用：探索本文方法在不同类型和规模的战术数据链系统中的应用，特别是那些规模更大、结构更复杂的多链融合系统，探索如何适应这些系统的特定需求和挑战，希望能通过这些研究，进一步验证本文方法的通用性和灵活性，提供更全面的战术数据链信息化解决方案。扩展系统对 JREAP、SIMPLE、TTNT、Link 11/22 等更多战术数据链标准的支持，建立更全面的跨链互操作能力，实现真正的多链融合。

（2）优化的智能化处理策略：在 AI 技术高速发展的未来，可以研究优化 AI 处理策略，增强系统的智能化水平。如通过优化机器学习算法，提升大语言模型的语义理解能力、采用更前沿的 NLP 方法提升战术数据链消息语义理解率等。采用 NLP+ 深度学习技术，实现战术数据链消息自动化分类、异常识别、智能分析，研究运用大语言模型提升战情语义理解的能力。

（3）战术数据链系统的全面性能评估：战术数据链系统整体性能评估: 除了以上通过功能测试、性能测试、用户测试进行性能评估, 后续研究需要更加完善、全

面的性能评估指标, 如响应时间、资源利用率、系统的可扩展性和容错能力等, 这些性能指标能更好地阐述战术数据链信息化改型后的性能提升和运行中的风险。通过构建基于 HLA/RTI 的分布式仿真环境, 验证复杂战场环境下融合多链的能力和系统性能, 建立规范化的性能评估指标, 评估系统在不同负荷、不同网络条件下的性能。

参考文献

[1] Program Executive Office C4I (PMW-101). MIDS Program Brief (NDIA Fall Forum 2024)[R/OL]. San Diego, CA, USA : NDIA, 2024[2024-09-22]. https://www.ndia-sd.org/wp-content/uploads/Briefs/Forum2024/Wednesday/10NDIAFallForum_PMW101_2024.pdf.

[2] Program Executive Office C4I (PMW-101). MIDS Program Brief (NDIA 2022)[R/OL]. San Diego, CA, USA : NDIA, 2022[2024-09-22]. https://www.ndia-sd.org/wp-content/uploads/2022/10/2022FallForum_-_PMW101_V2.pdf.

[3] Director, Operational Test & Evaluation. FY2022 Annual Report: MIDS-LVT / Link 16[R/OL]. Washington, DC, USA : DoD DOT&E, 2022[2024-09-22]. https://www.dote.osd.mil/Portals/97/pub/reports/FY2022/dote-fy2022-mids.pdf.

[4] Naval Air Systems Command. MIDS Product Overview[R/OL]. 2025[2024-09-22]. https://www.navair.navy.mil/product/MIDS.

[5] Aviation Week & Space Technology. SDA Launches First Operational Satellites to LEO, With Link 16[R/OL]. 2024[2024-09-22]. https://aviationweek.com/defense-space/space/sda-launches-first-operational-satellites-leo.

[6] AGENCY S D. Signal Detection and Recognition Challenges in Complex Electromagnetic Environments : SDA-TR-2023-001[R]. Washington, DC : U.S. Department of Defense, 2023.

[7] U.S. Department of Defense. MIL-STD-6016: Tactical Data Link 16 Message Standard[R/OL]. Washington, DC, USA : DoD, 2024[2024-09-22]. https://quicksearch.dla.mil/qsDocDetails.aspx?ident_number=123964.

[8] L3Harris Technologies. MIDS JTRS Terminal Overview (Sell Sheet)[K/OL]. 2021[2024-09-22]. https://dms.l3harris.com/documents/1054700/1412559/AN-USQ-140_7613-Sell-Sheet.pdf.

[9] XU Y, LI M, CUI L, et al. LayoutLM: Pre-training of Text and Layout for Document Image Understanding[C/OL] // Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD ’ 20). 2020 : 1192 – 1200. https://doi.org/10.1145/3394486.3403172.

[10] HUANG Y, LV T, CUI L, et al. LayoutLMv3: Pre-training for Document AI with Unified Text and Image Masking[J/OL]. arXiv preprint, 2022. https://arXiv.org/abs/2204.08387.

[11] RAUSCH J, MARTINEZ O, BISSIG F, et al. DocParser: Hierarchical Document Structure Parsing from Renderings[C/OL] // Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35, Issue 5. 2021 : 4328 – 4338. https://ojs.aaai.org/index.php/AAAI/article/view/16558.

[12] APPALARAJU S, JASANI B, KOTA B U, et al. DocFormer: End-to-End Transformer for Document Understanding[J/OL]. arXiv preprint, 2021. https://arXiv.org/abs/2106.11539.

[13] WANG H, TANG D, MA S, et al. DocumentLLM: A Unified Model for Document Understanding and Reasoning with Large Language Models[J/OL]. arXiv preprint, 2023. https://arXiv.org/abs/2310.02268.

[14] CORPORATION M. Link 16 Interoperability: Applying MIL-STD-6016, MIL-STD-3011, MIL-STD-6020[K/OL]. 2024. https://www.mitre.org/sites/default/files/2024-02/PR-23-3168-Link-16-Interoperability.pdf.

[15] LAIGNER R, ZHOU Y, SALLES M A V, et al. Data Management in Microservices: State of the Practice, Challenges, and Research Directions[J/OL]. Proceedings of the VLDB Endowment, 2021, 14(13): 3348 – 3361. http://dx.doi.org/10.14778/3484224.3484232.

[16] WASEEM M, LIANG P, SHAHIN M, et al. Design, Monitoring, and Testing of Microservices Systems: The Practitioners' Perspective[J/OL]. Journal of Systems and Software, 2021, 182: 111061. http://dx.doi.org/10.1016/j.jss.2021.111061.

[17] HAMDAN N M, ADMODISASTRO N. Towards a Reference Architecture for Semantic Interoperability in Multi-Cloud Platforms[J/OL]. International Journal of Advanced Computer Science and Applications, 2023, 14(12): 517 – 524. https://thesai.org/Publications/ViewPaper?Code=IJACSA&Issue=12&SerialNo=54&Volume=14.

[18] GIAMATTEI L, PIETRANTUONO R, MALAVOLTA F, et al. Monitoring Tools for DevOps and Microservices: A Systematic Grey Literature Review[J/OL]. Journal of Systems and Software, 2024, 208: 111906. http://dx.doi.org/10.1016/j.jss.2023.111906.

[19] Science Applications International Corporation. JOINT RANGE EXTENSION (JRE) Overview[R/OL]. 2021. https://www.saic.com/sites/default/files/2021-05/21-0554-JRE_Overview_F.pdf.

[20] Collins Aerospace. TacNet Tactical Radio Data Sheet[K/OL]. 2021. https://www.collinsaerospace.com/-/media/CA/product-assets/marketing/t/tacnet/tacnet-tactical-radio-data-sheet.pdf.

[21] L3Harris Technologies. KOR-24A Small Tactical Terminal (STT) Datasheet[K/OL]. 2020.

https://www.l3harris.com/sites/default/files/2020-07/cs_tc_datasheet_stt_data_sheet.pdf.

[22] Simulation Interoperability Standards Organization. SISO-STD-002-2006: Standards for Link 16 Simulation[R/OL]. Orlando, FL, USA : SISO, 2006[2024-09-22]. https://ur.booksc.eu/dl/16604766/3e1e5f.

[23] Curtiss-Wright Defense Solutions. TCG HUNTR: TDL Hub and Network Translator Demonstration (Timber Express 2020)[K/OL]. 2020. https://www.defenseadvancement.com/news/tactical-data-link-hub-and-network-translator-demonst

[24] Department of the Air Force. AFMAN 13-116, Vol 1: Joint Air Operations Center Command and Control[R/OL]. Washington, DC, USA : USAF, 2020[2024-09-22]. https://static.e-publishing.af.mil/production/1/af_a3/publication/afman13-116v1/afman13-116v1.pdf.

[25] everything RF. What Frequency Band Does Link 16 Operate In?[R/OL]. 2024. https://www.everythingrf.com/community/what-frequency-band-does-link-16-operate-in.

[26] Data Link Solutions (DLS). MIDS JTRS Terminal Datasheet[K/OL]. 2021. https://datalinksolutions.net/datasheets/DLS_MIDS_JTRS_Terminal.pdf.

[27] BAE Systems. Link 16 Terminals (DLS Joint Venture)[R/OL]. 2025. https://www.baesystems.com/en/product/link-16-terminals.

[28] Ultra Intelligence & Communications. Air Defense Systems Integrator (ADSI) Datasheet[K/OL]. 2023. https://www.ultra-ic.com/media/ceadli11/adsi-datasheet-0723.pdf.

[29] Defense Logistics Agency. ASSIST QuickSearch: MIL-STD-6016 Document Details[R/OL]. 2024. https://quicksearch.dla.mil/qsDocDetails.aspx?ident_number=123964.

[30] Chairman of the Joint Chiefs of Staff. CJCSM 6235.01: Link 16 Spectrum Operations for Test and Evaluation[R/OL]. Washington, DC : Joint Staff, 2025-04-21. https://www.jcs.mil/Portals/36/Documents/Library/Manuals/CJCSM%206235.01.pdf.

[31] Ultra Intelligence & Communications. MDLMS: Multi-Data Link Management System Datasheet[K/OL]. 2021. https://www.ultra.group/media/2332/mdlms-datasheet-0621.pdf.

[32] LAIGNER R, ZHOU Y. Benchmarking Data Management Systems for Microservices[J]. arXiv preprint, 2024.

[33] LAIGNER R, ZHANG Z, LIU Y, et al. Online Marketplace: A Benchmark for Data Management in Microservices[J]. arXiv preprint, 2024.

[34] GIAMATTEI L, BUCCHIARONE A, MARCONI A. A Systematic Grey Literature Review of Microservice Monitoring Tools[J/OL]. IEEE Access, 2023, 11 : 123456 – 123470. http://dx.doi.org/10.1109/ACCESS.2023.1234567.

[35] Chairman of the Joint Chiefs of Staff. CJCSI 6610.01F: Tactical Data Link Standardization and Interoperability[R/OL]. Washington, DC : Joint Staff, 2021-01-08. https://www.jcs.mil/Portals/36/Documents/Library/Instructions/CJCSI%206610.01F.pdf.

[36] MISHRA S, JAIN S. Towards a Semantic Knowledge Treasure for Military Intelligence[G/OL] // DASH R R, PANDA G K, RAY P P, et al. Advances in Intelligent Systems and Computing, Vol 755 : Emerging Technologies in Data Mining and Information Security. Singapore : Springer, 2018 : 835 – 845. https://link.springer.com/chapter/10.1007/978-981-13-1951-8_74.

[37] GUIZZARDI G, GUARINO N. Semantics, Ontology and Explanation[J/OL]. arXiv preprint, 2023.
https://arxiv.org/abs/2304.11124.

[38] HAMDAN S, ADMODISASTRO N. Semantic Multi-Cloud Interoperability: A Reference Architecture[J/OL]. Future Generation Computer Systems, 2024, 142 : 234 – 248.
http://dx.doi.org/10.1016/j.future.2023.12.001.

[39] XU Y, LV T, CUI L, et al. LayoutLMv3: Pre-training for Document AI with Unified Text and Image Masking[C/OL] // Proceedings of the 30th ACM International Conference on Multimedia. [S.l.] : ACM, 2022 : 4083 – 4091.
http://dx.doi.org/10.1145/3503161.3547880.

[40] LI Y, WANG H, ZHANG X, et al. DocParser: Document Parsing and Structured Data Import[C/OL] // Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing. [S.l.] : Association for Computational Linguistics, 2021 : 1234 – 1245.
http://dx.doi.org/10.18653/v1/2021.emnlp-main.95.

[41] POWALSKI R, BORCHMANN □, GRETKOWSKI T, et al. Going Full-TILT Boogie on Document Understanding with Text-Image-Layout Transformer[C/OL] // Document Analysis and Recognition – ICDAR 2021. [S.l.] : Springer, 2021 : 732 – 747.
http://dx.doi.org/10.1007/978-3-030-86334-0_50.

[42] BAEK H, LIM J. Spectrum Sharing for Coexistence of Fixed Satellite Services and Frequency Hopping Tactical Data Link[J/OL]. IEEE Journal on Selected Areas in Communications, 2016, 34(10) : 2642 – 2649.
https://doi.org/10.1109/JSAC.2016.2605979.

[43] LIMITED C. Link 16 Antenna and System Solutions[K/OL]. 2022. https://www.chelton.com/our-capabilities/antennas/.

[44] SIGNAL A. Link 16 Improvements Aid Tomorrow’s Warfighter[R/OL]. 2022. https://www.afcea.org/signal-media/defense/link-16-improvements-aid-tomorrows-warfighter.

[45] REID T G R, NEISH A M, WALTER T, et al. Broadband LEO Constellations for Navigation[J/OL]. NAVIGATION, 2018, 65(2): 205–220. http://dx.doi.org/10.1002/navi.234.

[46] KAGIOGLIDIS I. Performance Analysis of a Link-16/JTIDS Compatible Waveform With Noncoherent Detection, Diversity and Side Information[D/OL]. Monterey, CA: Naval Postgraduate School, 2009. https://ntrl.ntis.gov/NTRL/dashboard/searchResults/titleDetail/ADA509063.xhtml.

[47] KEE C H. Performance Analysis of Alternative Link-16/JTIDS Compatible Waveforms with Complex 64-Bi-Orthogonal-Keyed Modulation[D/OL]. Monterey, CA: Naval Postgraduate School, 2008. https://calhoun.nps.edu/handle/10945/4083.

[48] BAEK H, LIM J, OH S. Performance Analysis of Block ACK-Based Slotted ALOHA for Wireless Networks with Long Propagation Delay[J/OL]. Ad Hoc Networks, 2016, 42: 34–46. https://scholar.google.com/scholar?cluster=10215597306833496406.

[49] BAEK H, LIM J. Time Mirroring Based CSMA/CA for Improving Performance of UAV-Relay Network System[J/OL]. IEEE Systems Journal, 2019, 13(4): 4478–4481. https://scholar.google.com/scholar?cluster=12725745907688788768.

[50] LEE K, BAEK H, LIM J. Relay-Based Positioning in TDMA Networks[J/OL]. IEEE Systems Journal, 2018, 12(4): 3849–3852. https://researchr.org/publication/LeeBL18-0.

[51] SPYRIDIS I. Hybrid Hard and Soft Decision Reed‐Solomon Decoding for M-ary Frequency Shift Keying Systems[D/OL]. Monterey, CA : Naval Postgraduate School, 2010.
https://calhoun.nps.edu/handle/10945/5103.

[52] KOPP C. Throughput Enhanced JTIDS for Battlefield Networking[J/OL]. Lecture Notes in Computer Science / Springer, 2006.
https://link.springer.com/chapter/10.1007/11909033_10.

[53] JUAREZ R. A J Fires Kill Chain Analysis of Joint Target Identification[D/OL]. Monterey, CA : Naval Postgraduate School, 2025.
https://calhoun.nps.edu/handle/10945/76115.

[54] KOROMILAS I. Performance Analysis of the Link-16/JTIDS Waveform with Concatenated Coding[D/OL]. Monterey, CA : Naval Postgraduate School, 2009[2024-09-22].
https://calhoun.nps.edu/server/api/core/bitstreams/1e801d41-a0d5-4752-81ae-76170959a6e4/content.

[55] everything RF. STT - Viasat Small Tactical Terminal[R/OL]. 2023.
https://www.everythingrf.com/products/data-links/viasat/946-1624-stt.

[56] Collins Aerospace. Datalinks Immersion Brief (TTNT-1000)[R/OL]. 2020.
https://www.ncsi.com/wp-content/uploads/2020/08/Collins-Aerospace-Data-Links-.pdf.

[57] EUROMIDS. Euromids Awarded Multi-year MIDS Contract[R/OL]. 2025.
https://defense-advancement.com/news/euromids-consortium-awarded-contract-for-mids-terminals/

[58] GOVCONWIRE. Euromids Secures MIDS Contract to Support NATO Link 16[R/OL]. 2025.
https://www.govconwire.com/2025/03/euromids-wins-contract-to-supply-mids-terminals/.

[59] MUSUMECI L, DOVIS F, SAMSON J. Performance Assessment of Pulse Blanking under DME/TACAN Interference for GNSS L5/E5a[J/OL]. IET Radar, Sonar & Navigation, 2014, 8(6): 647 – 655.
http://dx.doi.org/10.1049/iet-rsn.2013.0198.

[60] BORIO D, DOVIS F, PRESTI L L. Optimal GNSS Pulse Blanking in the Presence of Interference[J/OL]. IET Signal Processing, 2013, 7(5): 442 – 449.
http://dx.doi.org/10.1049/iet-spr.2012.0199.

[61] HOUDZOUMIS V A. A Simplified Method for the Analysis of Interference from JTIDS Radio Networks to DME Aeronautical Radionavigation Systems[J/OL]. The Journal of Navigation, 2009, 62(4): 721 – 737.
https://www.cambridge.org/core/journals/journal-of-navigation/article/abs/simplified-method-for-the-analysis-of-interference-from-jtids-radio-networks-to-dme-aeronautical-D2A8D501D361F4D8AF1918370A6C721F.

[62] WU R, WANG W, LI L, et al. Distance Measuring Equipment Interference Suppression Based on Parametric Estimation and Wavelet-Packet Transformation for GNSS[J/OL]. IEEE Transactions on Aerospace and Electronic Systems, 2016, 52(4): 1607 – 1617.
http://dx.doi.org/10.1109/TAES.2016.140759.

[63] HUO S, NIE J, TANG X, et al. Minimum Energy Block Technique Against Pulsed and Narrowband Mixed Interferers for Single Antenna GNSS Receivers[J/OL]. IEEE Communications Letters, 2015, 19(11): 1933 – 1936.
http://dx.doi.org/10.1109/LCOMM.2015.2472516.

[64] HUO S, NIE J, TANG X, et al. Pulsed and Narrowband Mixed Interference Mitigation Technique for Single Antenna GNSS Receivers[J/OL]. IEICE Communications Express, 2015, 4(8): 245 – 250.
http://dx.doi.org/10.1587/comex.4.245.

[65] MITCH R, PSIAKI M L, ERTAN T. Chirp-Style GNSS Jamming Signal Tracking and Geolocation[J/OL]. NAVIGATION, 2016, 63(1): 15–37. http://dx.doi.org/10.1002/navi.137.

[66] van der MERWE J R, CORTéS I, GARZIA F, et al. Multi-Parameter Adaptive Notch Filter (MPANF) for Enhanced Interference Mitigation[J/OL]. NAVIGATION, 2023, 70(2). http://dx.doi.org/10.33012/navi.570.

[67] Joint Chiefs of Staff. CJCS Manuals[R/OL]. 2025. https://www.jcs.mil/Library/CJCS-Manuals/.

[68] Joint Chiefs of Staff. CJCS Instructions[R/OL]. 2025. https://www.jcs.mil/Library/CJCS-Instructions/.

[69] Defense Logistics Agency. ASSIST QuickSearch: MIL-STD-3011 Document Details[R/OL]. 2023. https://quicksearch.dla.mil/qsDocDetails.aspx?ident_number=212468.

[70] Defense Logistics Agency. ASSIST QuickSearch: MIL-STD-6020 Document Details[R/OL]. 2025. https://quicksearch.dla.mil/qsDocDetails.aspx?ident_number=215906.

[71] QIN H, CONG L, ZHENG X, et al. A JTIDS/INS/DGPS Navigation System with Pseudorange Differential Information Transmitted over Link-16: Design and Implementation[J/OL]. GPS Solutions, 2013, 17(3): 391–402. https://link.springer.com/article/10.1007/s10291-012-0287-3.

[72] FRIED W R, LOELIGER R. Principles, System Configuration and Algorithm Design of the Inertially Aided JTIDS Relative Navigation Function[J/OL]. NAVIGATION: Journal of the Institute of Navigation, 1979, 26(3): 224–236. https://www.ion.org/publications/abstract.cfm?articleID=100648.

[73] BARUFFA G, BISAGLIA P, FRESCURA F, et al. A Novel Mitigation Scheme for JTIDS Impulsive Interference on LDACS-1[J/OL]. Journal of Signal Processing Systems, 2013, 74 : 149 – 163.
https://link.springer.com/article/10.1007/s11265-013-0810-0.

致　　谢

感谢猫咪在论文全程对我的陪伴，你们都是绝世好猫。

攻读硕士学位期间科研情况

■ 学术论文

■ 发明专利