南京大學

研 究 生 毕 业 论 文

（申请工程硕士学位）

论 文 题 目	基于知识图谱的众测报告融合系统的设计与实现
作 者 姓 名	李文龙
学科、专业名称	工程硕士（软件工程领域）
研 究 方 向	软件工程
指 导 教 师	陈振宇 教授，冯洋 助理研究员

2020 年 5 月 20 日

学　　号：**MF1932097**

论文答辩日期：**2020 年 5 月 20 日**

指 导 教 师：　　**（签字）**

The Design and Implementation of Crowdsourced Testing Report Fusion System Based on Knowledge Graph

By

Wenlong Li

Supervised by

Professor **Zhenyu Chen**

Research Associate **Yang Feng**

A Thesis

Submitted to the Software Institute

and the Graduate School

of Nanjing University

in Partial Fulfillment of the Requirements

for the Degree of

Master of Engineering

Software Institute

April 2021

南京大学研究生毕业论文中文摘要首页用纸

毕业论文题目：基于知识图谱的众测报告融合系统的设计与实现

工程硕士（软件工程领域）专业 2019 级硕士生姓名：李文龙

指导教师（姓名、职称）：陈振宇 教授，冯洋 助理研究员

摘 要

随着互联网的飞速发展，软件技术已经应用到生活中的方方面面，测试技术也层出不穷。众包测试指的是在互联网上开展的、在一个规定的时间周期内由雇主雇佣众包工人对指定目标进行测试并提交测试结果的测试方法。众包测试由于参加人数多、工人间缺乏沟通，具有报告重复率高、描述冗余的特点。众测中工人提交的报告不能直接进行交付，需要特定的人员识别重复报告、整理内容并生成最终的交付报告。为准确识别重复报告、提高报告整编效率，本文设计了基于知识图谱的众测报告融合系统。

本文依托于众包测试平台，设计并实现了基于知识图谱的众测报告融合系统，创新性地构建众测知识图谱，挖掘报告文本描述间的语义联系。本文使用自然语言处理技术提取报告描述文本中的实体及关系，并引入分类知识图谱进行补充，使用翻译模型对知识图谱进行向量翻译，根据翻译后的向量计算缺陷报告实体相似度，将重复报告聚合在同一报告簇中。同时本文计算了报告中的图片特征用于辅助重复报告识别。其次，针对描述冗余问题，本文使用 PageRank 算法计算报告在对应报告簇中的权重，提取报告簇中的主要报告，并将类簇中的报告拆解为文字或图片的单一补充项，提取出主报告中未描述到的内容生成补充点，本文还对报告描述文本进行识别，查看是否存在描述相悖的内容，针对描述相悖内容生成歧义点。本文首先将任务内的重复报告聚合成报告簇，然后将报告簇拆分成主要报告、描述补充点及描述歧义点，这使得缺陷报告的分类和信息获取更加高效，进而提高了整编人员的效率。

本文主要划分为：知识图谱模块、知识图谱报告融合模块、图片计算模块和报告整编模块。为实现系统的高效访问，系统使用了 Nginx 进行负载均衡，使用 Redis 作为缓存，使用 Thrift 框架为跨语言模块之间提供高效通讯。为响应信创国产软件的号召，本文还针对国产操作系统进行可移植性适配。

目前本系统已经在相关线上项目上得到应用，系统实现了众包测试重复缺陷报告识别，并对报告中的关键观点进行了提取，这极大提高了整编人员识别重复报告和信息获取的效率，整编人员所交付的报告的质量也得到了有效提升。

关键词：众包测试，知识图谱，报告融合，报告整编

南京大学研究生毕业论文英文摘要首页用纸

THESIS: The Design and Implementation of Crowdsourced Testing Report Fusion System Based on Knowledge Graph

SPECIALIZATION: Software Engineering

POSTGRADUATE: Wenlong Li

MENTOR: Professor **Zhenyu Chen**
Research Associate **Yang Feng**

Abstract

With the rapid development of the Internet, software technology has been applied to all aspects of life, and testing technologies are also emerging in endlessly. Crowdsourced testing refers to a test method carried out on the Internet, where the employer hires crowdsourced workers to test the specified target and submit the test results within a specified time period. Due to the large number of participants and lack of communication among workers, crowdsourced testing have the characteristics of high repetition of reports and redundant descriptions. The reports submitted by workers in the crowdsourced testing cannot be directly delivered, and specific personnel are required to identify similar reports, organize the content, and generate the final delivery report. In order to accurately identify similar reports and improve the efficiency of report compilation, this thesis designs a crowdsourced testing report fusion system based on knowledge graph.

Based on the crowdsourced testing platform, this thesis designs and implements a crowdsourced testing report fusion system based on the knowledge graph. The system innovatively constructs a crowdsourced testing knowledge graph, and uses knowledge graph technology to assist report fusion. This thesis uses Natural Language Processing technology to extract the entities and relationships in the report description, and introduces the classification knowledge graph to supplement, explores the semantic connection between the reports, uses translation model to perform vector translation of the knowledge graph, and aggregates the reports describing similar defects in the same cluster. At the same time, this thesis calculates the image features in the report to assist

in report fusion. Secondly, for the problem of description redundancy, this thesis uses the PageRank algorithm to calculate the weight of the report in the report cluster, extracts the main report in the report cluster, and disassembles the report in the cluster into a single supplementary item of text or picture, and extracts the content not described in the main report forms supplementary points. This thesis also identifies the report description, checks whether there is any content that contradicts the description, and generates ambiguity points for the contradictory content. This thesis first aggregates similar reports within the task into report clusters, and then splits the report clusters into main report, description supplementary points, and description ambiguity points. This makes the classification and information acquisition of defect reports more efficient, thereby improving the efficiency of reviewer.

This thesis is mainly divided into: knowledge graph module, knowledge graph report fusion module, picture calculation module and report summary module. In order to achieve efficient access to the system, the system uses Nginx for load balancing, Redis as a cache, and Thrift framework to provide efficient communication between cross-language modules. In response to the call of Xinchuang's domestic software, this thesis also carries out portability adaptation for domestic operating systems.

At present, the system has been applied to related online projects. The system has realized the identification of similar defect reports in crowdsourced testing, classified and extracted the opinions in the reports, and improved the efficiency of reviewer to identify similar reports and information acquisition. The quality of delivered reports has also been effectively improved.

Keywords: Crowdsourced Testing, Konwledge Graph, Report Fusion, Report Integration

目录

表 目 录

图 目 录

第一章　引言

1.1　项目背景及意义

近年来，国内外互联网发展迅猛，软件技术已经应用到社会生活的方方面面，保障软件应用质量的测试技术也随之有着越发重要的地位。众包是互联网带来的一种分布式问题解决和生产组织模式 [1]，自从这个概念提出之后，众包已经在人工智能、自然语言处理、人机交互等领域得到应用并成为一个新的研究热点 [2]，众包可以将任务发布到互联网，并开放式的召集工人来完成一些计算机难以解决的问题 [3]。

近些年来这种模式也成功运用在软件工程领域，Mao 等人 [4] 对众包技术在软件工程领域中的应用进行了概括总结。在软件测试领域，众包模式也成功得以运用，称之为众包测试，对此 Zhang 等人 [5] 也对众包在软件测试领域的运用进行了概述。数量众多的众包工人参加并完成待测任务，可以很好的模拟软件应用场景及用户表现。众包测试的开展具有成本低廉且评测周期短的特点 [6]。在许多大型互联网公司内部，新版本发布前通常会在组织内部开展一次众包测试，通用的众包测试也得到许多商用测试企业的支持与发展。相比于传统的软件测试方法，线上开展的众包测试有许多优势，如众包测试因为参与用户多、测试环境不尽相同，相对更容易找到系统中的缺陷，更好的对系统在多种环境中的边界进行测试。此外，众包工人所提供的体验更接近于真实用户的体验，这使得产品的迭代更加接近于真正的用户思维。另外，众包工人用人成本低，无需雇佣专业的测试员工，这对小型公司来说是一个不错的选择。

在众包测试系统中，参与方主要是众测任务提供方、众测平台和众包工人 [7]。在传统软件测试过程中，测试人员大多是经过专业培训的人员，但在众包测试中，测试人员数量众多且绝大部分测试人员没有经过专业的培训。众包测试分为协作式和竞争式两种 [8]，在美团、字节等大型互联网企业内部，众包测试工作机制大多是竞争式的，众包工人之间不会共享信息，在这种情况下，系统中的同一个缺陷通常会被不同的测试人员报告数遍。这就造成了众包测试产出的报告具有数量众多但重复率高的问题 [9]。基于此，有公司研发了协作式的众包测试系统，众包工人在填写报告时可以看到其他工人填写的报告数据，可以选择对其他人提出的缺陷报告进行补充说明，系统称之为树状报告，这一定程度上减少了报告的重复率，但仍然存在重复的情况。因此众包测试通常需要报告整编人员对产出的报告进行整编聚合、信息提取。大量重复的报告和冗余的

信息无疑对报告整编人员造成了巨大的困难，如何识别重复缺陷报告及对报告关键内容进行抽取成为了一个新的研究方向。

在目前现有的软件测试平台中，每一次的众包测试都会产生众多描述同一缺陷的 Bug 报告，即使是协作式众包测试平台，其报告的重复率也居高不下。整编人员无法便捷的处理众多的 Bug 报告，通常是将用户提交的所有缺陷报告全部交付给任务发布方，这对整编人员和任务需求方来说都是不理想的结果，影响了众测业务后续开展。本系统面向众测报告整编，针对众包测试报告重复度高、缺陷报告缺乏合理的识别和分类机制的情况引入了报告融合流程，利用自然语言处理和知识图谱技术挖掘文本描述内容中的语义联系，识别重复报告，将描述相似度高的报告合并到同一报告簇中；本文还利用知识图谱图结构和工人互动数据使用 PageRank 算法提取了报告簇中的主报告，除了主报告外还提取了报告簇内的补充点和歧义点。整体而言，本系统引入知识图谱技术提高了重复报告识别的准确程度，同时还提取了报告簇中的主报告、补充点和歧义点供整编人员查看，提高了信息获取效率的同时改善了报告整编的整体流程。

1.2　国内外研究现状

1.2.1　知识图谱研究现状

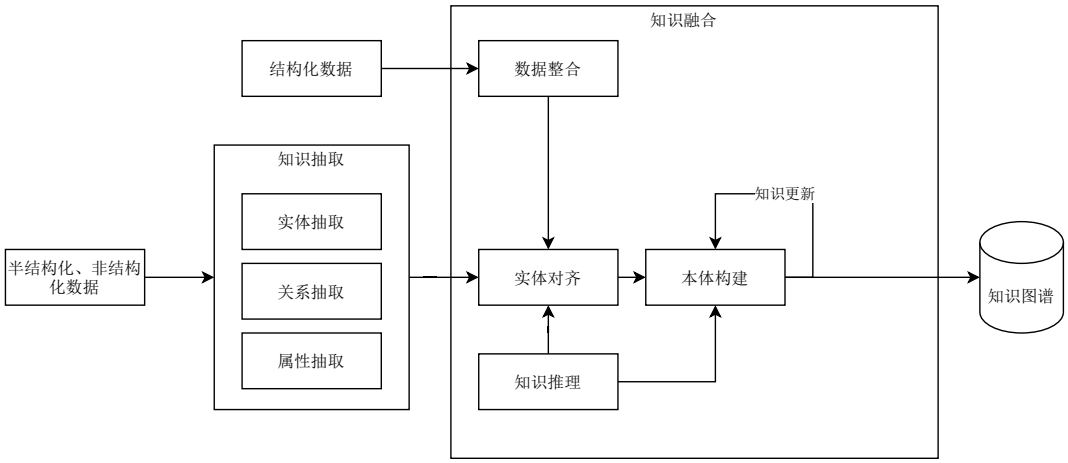

图 1.1: 知识图谱体系结构

知识图谱是人工智能的一个重要应用，知识图谱通过构建开放的、具有语义理解能力的知识库，可以在包括搜索引擎、个性化推荐、问答服务等众多服务

中产生较大的应用价值 [10]。知识图谱 (Knowledge Graph) 又可以称为知识域的可视化，它能够通过可视化的形式将知识通过节点与关系的方式进行展示，知识图谱技术能够对知识进行存储、描述、分析及挖掘，从而构建起知识间的联系，形成语义网络。在很多领域，知识图谱都能够进行很好的应用。从本质上来说，知识图谱即一种描述知识实体相互关系的语义网，图 1.1展示了知识图谱的体系架构。

知识图谱就是在已知的、可获取的结构化或非结构化数据上经过知识抽取、知识融合从而形成知识图谱。知识抽取执行后，可以在数据中提取出知识实体和相互关系及知识属性等信息。知识融合阶段经过消除实体、关系的歧义后，形成质量较高的知识库。在知识构建完成后，可以使用知识推理技术挖掘知识库中隐藏的知识，达到扩展、丰富知识图谱的目的。

在知识存储方面，Neo4j 公司开发了 Neo4j[11] 图数据库，并推出了用于 Neo4j 查询的 Cypher 语言。Neo4j 是一种性能极好的非关系型数据库（NoSQL），可以很好地将数据保存为单个的节点以及节点间的关系。

在知识表示上，Antoine Bordes[12] 等人提出了词嵌入模型 TransE，知识图谱中的知识可以看成是由许多（实体，关系，实体）的三元组组成。图谱中的知识实体和相互关系都能够翻译成一个低维度的向量进行表示。基于 TransE 可以对知识图谱进行知识推理，TransE 作为经典算法，还衍生出了 TransH[13] 和 TranR[14] 等一系列向量翻译算法。

1.2.2　缺陷报告相似检测研究现状

缺陷报告指的是用于描述程序发生缺陷时的结构化描述文本 [15]，缺陷报告相似检测指的是运用一系列算法进行度量缺陷报告之间的相似性，然后为指定报告推荐相似报告或直接标记雷同报告 [16]。国内外学者在缺陷报告相似检测从许多方面进行过不同的尝试。

例如 P. Runeson[15] 提出了一种将自然语言处理技术应用于软件缺陷报告相似度度量的方法。Wang 等人 [17] 提出了基于软件执行记录的重复报告检测方法。基于主题模型，Somasundaram[18] 也提出了一种缺陷报告重复度检测的技术，在为不同术语描述同一缺陷检测时提供了新的思路。Nguyen[19] 将文本检索和主题模型相结合，提出了一种根据主题和术语对报告进行重复检测的技术。Hindle[20] 则是利用了报告的上下文信息结合信息检索对报告重复检测进行了改进。Yang[21] 等人除了采用了词嵌入技术外还考虑到缺陷报告的组件信息等属性用于检测重复报告。

以上研究从很多技术的方面和角度对报告的重复检测进行了不同的尝试。

但基本都集中在对报告文本的处理上。考虑到众包测试报告中存在报告截图，特别是在针对移动 App 的功能测试中，用户提交的图片众多，报告中的图片也应该得到度量。Feng[22] 等人在报告相似性度量中，同时计算了缺陷报告文本相似度和图片相似度，并使用特定公式对二者进行计算，得到最终的报告相似度。对于缺陷报告图片的度量中，Wang[23] 也提出了一种新的思路，即在图片进行特征提取后，分情况计算缺陷报告的相似程度，图片相似度大于一个阈值的情况下使用文本相似度作为报告相似度，小于该阈值的情况下，使用图片相似度和文本相似度的均值作为报告相似度。这在一定程度上给了本文启发。余笙 [24] 等人提出了一种知识驱动的相似缺陷报告检测方法，主要使用知识图谱对缺陷报告进行关系提取，根据在知识图谱中有共同指向实体的报告进行初筛，用于提高后续根据文本描述识别雷同报告的准确性。

本文构建众测任务知识图谱，使用知识图谱翻译技术将报告实体翻译成向量，比较向量相似度用于发现重复报告。

1.2.3　缺陷报告摘要提取研究现状

缺陷报告摘要提取是文本摘要提取的一种，自动化文本摘要提取主要分为两种：抽取式摘要和生成式摘要。

抽取式摘要算法的核心在于从文档中提取排名靠前的句子来代表文档内容[25]。词频逆文本频率指数 TF-IDF[26]（Term Frequency–Inverse Document Frequency）提出后，不少研究者将 TF-IDF 技术应用在缺陷报告摘要提取中。EL Beltagy[27] 提出了一种基于规定的关键词列表使用 TF-IDF 对文档进行摘要提取的方法，目的是减小 TF-IDF 对重要关键字的影响。PageRank[28] 算法和 TextRank[29] 算法提出之后，迅速在文本摘要提取领域得到了应用。Mihalcea[30] 提出了一种基于 PageRank 和 TextRank 在文本中提取关键词生成摘要文本的技术。对于多文档的摘要提取，Khan[31] 等人基于语义网络提出了一种新的方法，具体做法是将句子抽象为图中的节点，句子与句子之间的相似度作为句子节点之间的加权边，根据图结构的排序算法对待提取文档中的单一句子进行排序最终生成文本摘要。Hao[32] 提出了一种使用 PageRank 算法计算多份文档中主报告的方法，同时提取出其他报告描述不同的语句作为 “补充点” 信息，对补充点信息进行融合合并成差异观点。

抽取式摘要算法是对文档内的内容进行提取，生成式摘要则有很大不同，其是对文档的语义信息进行提取，通过模型训练，自动生成能够代表文章中心思想的语句。Hinton[33] 首次将深度学习应用到摘要提取中，其使用无监督学习的方式改善了有监督依赖标记数据的缺陷。Nallapati[34] 等人提出了一种 Sequence-

to-Sequence 模型，在信息抽取、翻译等领域都有应用。

系统中的报告融合阶段主要使用了缺陷报告摘要提取技术，用于确认报告簇的主要报告，并针对报告簇内容分别提取了补充点和歧义点。

1.3　本文主要工作

本文所实现的系统是基于协作式众包测试平台上开展的，目的是针对当前众测系统缺乏合理的报告整编机制、基于文本相似度的重复缺陷报告识别准确度不高、缺乏有效的报告关键内容提取的情况。协助整编人员解决报告整编工作量大、重复报告识别效果不好的问题。

本文通过以下几个方面来实现目标：

一方面提供重复报告识别及对报告内容进行信息抽取的功能，考虑报告中描述具有语义联系，本文引入知识图谱技术来对报告描述中的语义关系进行发掘，提供准确度较高的重复报告识别，并对报告内容进行了信息抽取，提取了主报告、补充点和歧义点信息。这一阶段的难点在于如何将知识图谱技术和现有技术整合以进行重复报告识别。

另一方面改善报告整编设计的业务流程，提高整编人员的效率，提高系统的易用程度。系统提供了报告融合、融合视图查看、树状视图查看、报告补充点及歧义点提取、报告管理及多格式导出等功能。

综合上述方案和设计，本文主体部分使用 Java 基于 Spring Boot 框架进行开发，知识图谱部分作为单独的服务使用 Python 语言编写，跨语言模块之间通过 Thrift 进行通讯，主体系统通过 Thrift 和 HTTP 两种方式对外提供接口，为了保障系统的性能和可用性，系统使用 Nginx 进行负载均衡，使用 Redis 作为缓存。数据库方面采用主从备份的方式保障系统的可靠性。

1.4　本文组织结构

本文一共分为六个章节，文章组织结构如下：

第一章为引言，本章主要介绍了众包测试的定义，并就对众包测试开展过程中遇到的报告整编问题进行了介绍。介绍了国内外的研究现状，并描述了本文在解决问题上的思路。

第二章为相关技术概念介绍。本章主要介绍了项目实现中使用到的知识图谱技术、自然语言处理技术、相似度计算方法、PageRank 算法和远程过程调用框架 Thrift 等。

第三章为本文的需求分析与概要设计。本章介绍了本文的开发背景及预期目标，分别分析了系统的功能性需求和非功能性需求，描述了系统的主要设计

思路，并对系统中的关键模块进行了介绍。详尽描述了各模块的架构设计、类图设计、数据库设计和流程设计。

第四章在第三章需求分析与概要设计的基础上，介绍了系统的详细实现。本章对第三章中介绍的模块进行了详尽的分析，使用顺序图展示各模块的调用流程，并对模块中的关键代码进行说明，本章还对系统所实现的界面进行了展示。

第五章为系统测试，介绍了本系统所进行的各项测试工作，主要包括对系统的功能测试、可用性测试、可移植性测试及性能测试，并展示测试结果。

第六章为总结与展望。本章节简单总结了本文与本系统的各项工作与贡献，同时针对系统的不足之处进行了分析，并提出今后的改进方向。

第二章　相关技术概述

2.1　知识图谱技术简介

早在 2006 年，Berners-Lee 等 [35] 提出了链接数据的想法，并提出了语义网的概念，这种技术能将知识用图的形式进行处理并展示。知识图谱的概念首先由美国谷歌公司提出并在实际的项目中得到应用，其推出的知识图谱技术能够帮助理解与语义相关的信息检索问题。

2.1.1　Neo4j 图数据库

Neo4j 是一个可以存储知识图谱图形结构及节点关系的图形数据库。图数据库 (Graph Database) 能够将图形结构的数据进行存储并提供对图结构的查询，是一种新型的非关系型数据库。其基于图论的思想进行实现，数据库存储结构和数据的查询方式都以图论为基础，节点和节点之间的边是图论中的基本结构，分别在图数据库中对应图节点和节点间关系。

Neo4j 是目前使用最广泛的图数据库。其专门为图结构的存储及管理进行了特别的优化，在知识图谱中相互联系的节点在物理地址存储中也相互对应，这种物理结构上的对应关系更能发挥出 Neo4j 存储图形结构数据的优势。而在知识图谱中，知识就是使用图结构来表示的，因此非常适合使用 Neo4j 来存储知识图谱。

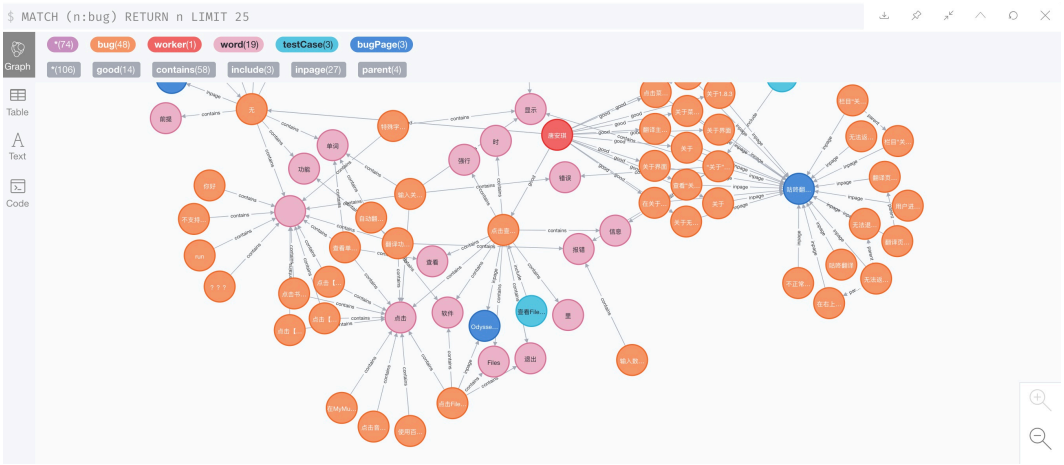

图 2.1: Neo4j 运行示意图

从物理层面上看，Neo4j 针对图结构的存储形式主要是节点和边两种。节点对应的就是知识图谱中的知识实体，而边对应的则是实体间的关系，关系可以是有向的也可以是无向的，具有普遍的适用性，其中节点上可以使用标签来对实体进行分类，这在知识图谱中对应实体的属性。如图2.1所示为 Neo4j 运行示意图。

2.1.2　TransE 翻译模型

在知识库中，知识被存储为三元组的形式，如 (h,r,t)，其中 r 代表关系，h 和 t 则分别是头实体和尾实体。TransE 模型是一种知识的表示方法，其可以根据三元组的结构学习知识图谱中的实体和关系，并将其映射到低维向量中。2013 年 Bordes[12] 等人提出了知识图谱的词嵌入模型 TransE 方法，TransE 方法的提出主要受到 Word2Vec[36] 具有的平移不变性启发，即在知识图谱的语义空间内，关系向量也具有同样的平移不变性。知识图谱中的实体及关系都能够使用一个低维度的向量进行表示。将知识实体和关系表示为低维向量后，能够更加便捷地在知识图谱内进行各种计算和基于现有知识的推理。

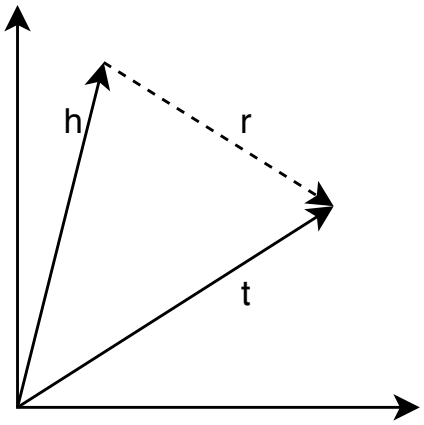

图 2.2: TransE 实体关系示意图

如图2.2所示是 TransE 实体及关系向量空间示意图。TransE 的基本思路为在 (h,r,t) 这样一对三元组中，将关系 r 看成是实体 h 到实体 t 的一个翻译，在模型训练过程中，通过不断的调整 h、r 和 t 向量，使得 h 和 r 向量之和尽可能的与 t 向量接近。在开始训练时，实体向量和关系向量都是随机生成的，每次迭代中，首先对知识实体的嵌入向量进行归一化，然后选取一部分三元组一批一批对数据进行训练，对于每个三元组都生成对应的负例（损坏的三元组）并组成 T_bacth，对所生成的 T_batch 进行训练，使用梯度下降方法进行调参。最终得到该知识图谱中的实体和关系向量。

2.2　NLP 技术分析

自然语言处理 NLP（Natural Language Processing）旨在使得计算机能够像人一样地理解自然语言并进行处理 [37]。NLP 研究的内容为人机之间使用自然语言进行沟通、交互的各种方法，是一个涉及许多研究方向的综合性科学。语言理解是人工智能领域最具现实意义和实现价值的研究之一，是人类和机器之间进行沟通的桥梁。本文主要使用了中文 NLP 技术中的分词、词性识别与依存分析技术，用于提取缺陷报告描述中的三元组关系。

2.2.1　中文分词

在英文中，英文单词是用进行语言表达的基本单位，英文句子的词与词之间通常使用空格进行间隔。但在汉语体系中，句子是由字所组成的，词语同样由字所构成，字词之间并没有具体的区分标准，因此要对中文句子进行理解，首先需要对句子进行中文分词，分词是进行中文自然语言处理的基础。Jieba 分词是一个用 Python 开发的、使用广泛的中文分词工具，其进行分词的思路主要为：（1）依照汉语词典对句子构建有向无环图（DAG），考虑到句子中所有能成词的可能。（2）对于不在词典中词语，使用 HMM 模型的 Viterbi 算法进行分词处理。（3）使用动态规划算法寻找概率最大的成词路径，并最终找到基于词频处理的句子切分组合。

2.2.2　词性标注

词性标注（Part-Of-Speech tagging）也可以称为语法标注，是一种对词语进行词性标记的技术，主要考虑单词自身的语言含义及其在句中上下文环境。在中文中，有很多词语具有同样的语法作用，能够在句子中出现在同样的位置，这些字词可以共同归类为一个范畴。中文词性标注具有很强的层次性，从最基本的字词可以分解为实词与虚词，如实词指的是具有具体含义的词，依照不同的使用情景，实词可以分为动词、名词与形容词等。词性标注技术能够在句子中判断每个词语的词性并标注，是中文 NLP 中一项非常基础但重要的工作。

2.2.3　依存分析

句法分析是自然语言处理的一项重要技术，其目的是针对输入的句子进行分析从而得到句子的句法处理过程。句子的依存关系分析是句法分析的一种也可以称之为依存句法分析。依存分析将句子拆分成树状结构的句法树，描述句中词之间的依存关系分析。在依存句法分析中，谓词是一个句子的核心，其他词

汇可以直接或间接的与谓词生成联系，句子依存关系分析树的根结点通常是句子中的谓词。

LTP（Language Technology Platform, LTP）是一个中文自然语言处理开源基础技术平台。Pyltp 是对 LTP 平台的一个封装，提供常见的 NLP 功能，包括分词、词性标注、依存分析等。如表2.1所示是依存句法中的关系类型举例。

表 2.1: 依存句法核心关系类型示例

关系类型	分析标签	举例
核心关系	HED	他送我一个苹果 (送)
主谓关系	SBV	他送我一个苹果 (他 ← 送)
动宾关系	VOB	他送我一个苹果 (送 → 苹果)
间宾关系	IOB	他送我一个苹果 (送 → 我)
前置宾语	FOB	他什么苹果都吃 (苹果 ← 吃)
定中关系	ATT	红苹果 (红 ← 苹果)
状中结构	ADV	非常好吃 (非常 ← 好吃)
动补结构	CMP	吃完了苹果 (吃 → 完)
并列关系	COO	香蕉和苹果 (香蕉 → 苹果)
介宾关系	POB	在树林内 (树林 → 内)
左附加关系	LAD	香蕉和苹果 (和 → 苹果)
右附加关系	RAD	香蕉和苹果 (香蕉 ← 和)

2.3　PageRank 算法

PageRank[28] 算法是由知名的谷歌公司创始人 Page 和 Brin 在 1998 年提的一种算法，是关于图之间链接关系分析的代表性算法。PageRank 算法最早被谷歌在搜索引擎用于网页间排序，其可以应用在文本摘要提取上。本文使用 PageRank 算法对报告簇中的报告进行排序。

2.3.1　算法介绍

PageRank 能够以网页间的超链接个数和网页质量作为主要参考因素进行网页的重要性计算 [38]。其拥有两大假设，数量假设认为越重要的页面会更多的被其他网页所引用（即超链接），如果一个页面所受到的其他页面链入数量越多，那么这个页面越重要；质量假设认为质量高的页面链接向其他页面时，其他页面也会获得更高的质量评估，即质量高的页面所链接的页面质量通常也较高。简单来说，PageRank 能够在图结构中根据节点权重及节点之间关系确认节点重要程度，其依托的是节点之间进行相互投票，根据投票的结果确定图中节点的重要性。算法步骤如下：（1）图结构构建完成后，为每个图中的节点设置初始值，初始值一般是相同的。（2）在每一轮的计算中，根据节点间链接及节点权重值

不断更新每个图节点的 PageRank 得分，当每个节点的得分都得到更新后，则当前轮的计算结束。算法结束的条件是当节点当前的 PageRank 得分与上一轮相差在规定范围内时，计算结束。

2.3.2　算法示例

假设图中有 4 个节点分别是 A，B，C 和 D，定义 PR(x) 为 x 节点的 PageRank 得分，L(x) 为 x 节点的出度。如果在图结构存在 B、C、D 均指向 A，那么节点 A 的得分是节点 B、C、D 得分之和。

$$PR(A) = PR(B) + PR(C) + PR(D) \tag{2.1}$$

如果只有节点 B 链接到 A，节点 C 除了 A 之外还链接到一个节点，那么节点 A 从节点 C 处所获得的 PageRank 值也会减半，节点 D 除了 A 还链接到另外两个节点，那么 A 所获得的 PageRank 值如下所示。

$$PR(A) = \frac{PR(B)}{1} + \frac{PR(C)}{2} + \frac{PR(D)}{3} \tag{2.2}$$

也就是说，节点 PageRank 的得分为其所链入节点的 PageRank 值除以链入节点出度之和。

$$PR(A) = \frac{PR(B)}{L(B)} + \frac{PR(C)}{L(C)} + \frac{PR(D)}{L(D)} \tag{2.3}$$

2.4　相似度计算

余弦相似性使用向量的余弦计算公式对向量间余弦值进行计算，计算的余弦值可以用作评估向量间的相似度，在向量空间中，两个向量的夹角越小，说明两个向量之间越相近，相似度程度也就越高。当两个向量几乎完全相同时，向量间的夹角为 0，余弦相似度为 1。对于向量空间中的 A、B 向量，其对应的余弦相似度计算公式如公式 1.4 所示：

$$\cos\theta = \frac{A \cdot \mathrm{B}}{\|A\| \cdot \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}} \tag{2.4}$$

2.5　RPC 框架 Thrift

RPC（Remote Procedure Call Protocal）远程过程调用协议指的是客户端在不关心底层实现的背景下，能够远程调用远端服务端中的对象。调用过程如同客户端在调用程序本地对象，RPC 是一种利用网络进行远程调用服务的协议。

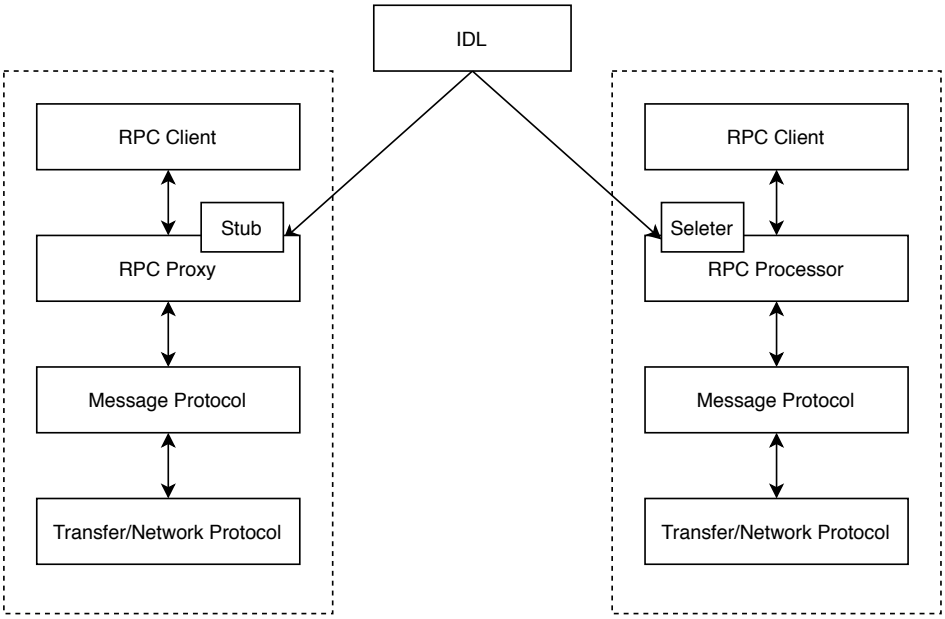

图 2.3: RPC 调用示意图

如图2.3所示是 RPC 框架流程示意图，“RPC Client”是服务调用方，“RPC Sever”是远程方法实现，其对服务调用方是无感知的。IDL（Interface Define Language）为接口定义语言，在跨语言的 RPC 服务调用中，IDL 是必要的、用于描述服务间提供的接口及接口定义的参数。“RPC Proxy”即代理，其在客户端管理消息格式及网络传输协议，判断并处理调用过程中的异常情况。“Processor”在 RPC 服务端中负责执行接口调用实现，其功能包括对接口注册的管理、判断调用方权限等。“Message Protocal”即消息管理层，实现了对网络传输信息进行编码及解码的功能。“Transfer/Network Protocal”是传输协议层，负责管理框架所使用的网络协议，Thrift 框架使用的网络协议是传输层的 TCP 协议。

Thrift 是一个轻量级的、支持多语言的 RPC 框架，它支持为 RPC 通讯自动生成代码。Thrift 框架为过程调用中数据的传输及序列化提供支持，对应用层提供了简洁明了的抽象 [39]。RPC 框架为服务做了封装处理，对服务间通讯的效率进行了优化。相比 HTTP 传输协议来说，RPC 框架的调用简化了传输内容，提高了通讯的效率。Thrift 是一种常用的跨语言 RPC 框架，由美国公司 Facebook 开

发，能够实现在多种语言编写的程序见进行远程调用 [39]。Thrift 提供了的 IDL 文件经过 Thrift 编译之后可以生成多语言版本的接口文件，用户可以根据自己采用的语言进行使用。Thrift 还支持为程序提供多种工作模式，例如非阻塞模式和线程池模式等，其所拥有的广泛的适用性可以为服务提供高效的对外服务，具有优秀的性能及跨语言能力。

2.6　本章小结

本章主要就本系统相关的技术进行了概述说明。首先对于知识图谱技术进行了介绍，包括图数据库与知识图谱翻译模型。第二，对 NLP 技术进行了介绍，NLP 技术在本系统中主要用于分析报告文本描述内容进行关系提取。第三，对 PageRank 算法进行了介绍，PageRank 算法在本系统中用于在报告簇中提取主报告。第四，对向量相似度计算方法进行了介绍，余弦相似度在系统中被用于计算向量相似度。最后，对远程过程调用框架 Thrift 进行了概述，该框架用于系统跨语言模块间通讯。

第三章　基于知识图谱的众测报告融合系统需求分析与概要设计

本章就基于知识图谱的众测报告融合系统的需求分析与系统设计进行说明，首先分析了系统的总体需求和需要实现的相应功能，然后就系统中的功能性需求和非功能性需求进行了分析，并详细描述了系统中的主要用例。之后对系统进行关键模块划分，描述了系统的总体设计，并通过 4+1 视图的方式阐述系统顶层设计。最后就所实现系统的主要模块进行具体说明，主要描述了模块的整体设计、核心类图设计和数据库设计。

3.1　系统整体概述

在目前的众包测试系统中，因为众包机制的问题，参与者人数众多且专业水平参差不齐。用户在系统中提交的测试报告具有重复性高、重复描述同一缺陷的特点；协作式众测平台虽然在一定层面上缓解了这个问题，但是在每场众测任务中仍然有不少缺陷报告。系统在向众测任务发布方交付任务最终报告时，由于缺乏合理科学的分类方法和报告整编机制、无法将用户提交的描述同一缺陷的 Bug 报告合理的聚合在一起供整编人员整编，也没有就缺陷报告的内容进行分析。本文利用知识图谱和融合算法等技术，改进了这一流程，将任务中的重复报告合并到同一报告簇中，并提供相似报告推荐供整编人员融合，融合过程中本系统提取了不同报告中的主要观点、补充点和歧义项，并对报告簇中的报告使用 PageRank 算法进行排序，方便整编人员从报告簇中提取报告的主要信息，从而使得整编人员的使用效率得到大幅改善。

如图3.1所示是系统的处理流程。众包工人在众测任务中提交的原始报告主要描述了用户的操作步骤和系统的反馈，如“点击按钮”、“输入验证码”、“App 提示格式错误”等用户操作及系统反馈。原始报告首先经过同位词替换进行数据增强，然后使用 NLP 技术对报告描述进行关系提取，主要是提取报告描述中用户的操作和系统的反馈等内容。NLP 技术处理中使用了分词、词性分析和依存关系分析等。然后就系统内的测试用户、报告数据、用户数据及报告点赞点踩等结构化数据进行知识图谱构建，并引入第三方分类知识图谱进行补充。这一过程主要使用了 Neo4j 这一图数据库进行知识的存储。构建好的模型使用 TransE 模型进行训练并将知识图谱进行向量化翻译。

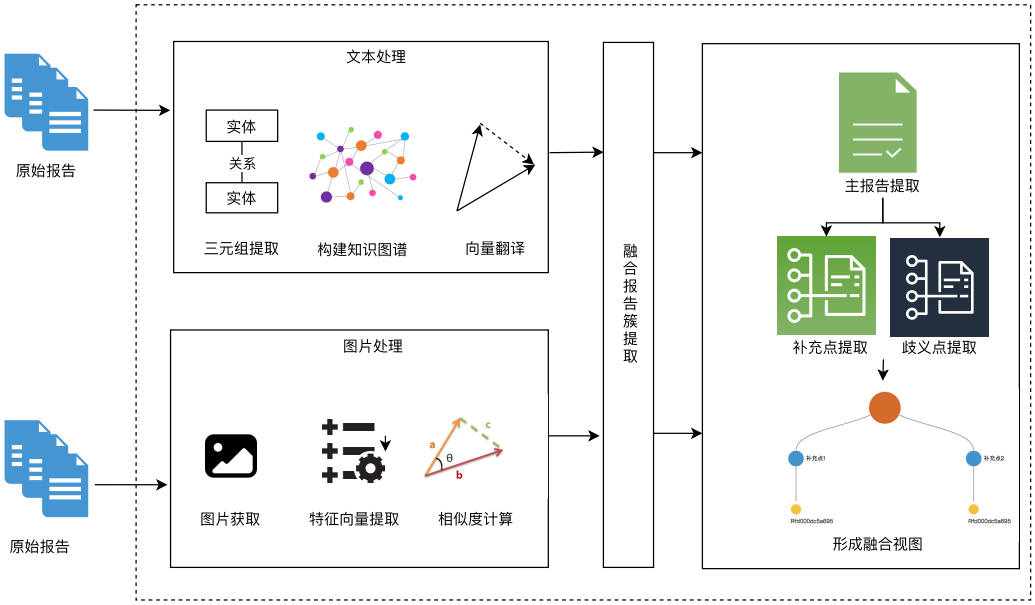

图 3.1: 系统整体流程图

树状报告是协作式众测系统中一种用户自发生成的描述同一 Bug 的报告，用户可以选择一份报告作为父报告进行完善补充，从而生成一份完整的缺陷描述报告。如图3.2所示是系统树状报告示意图。

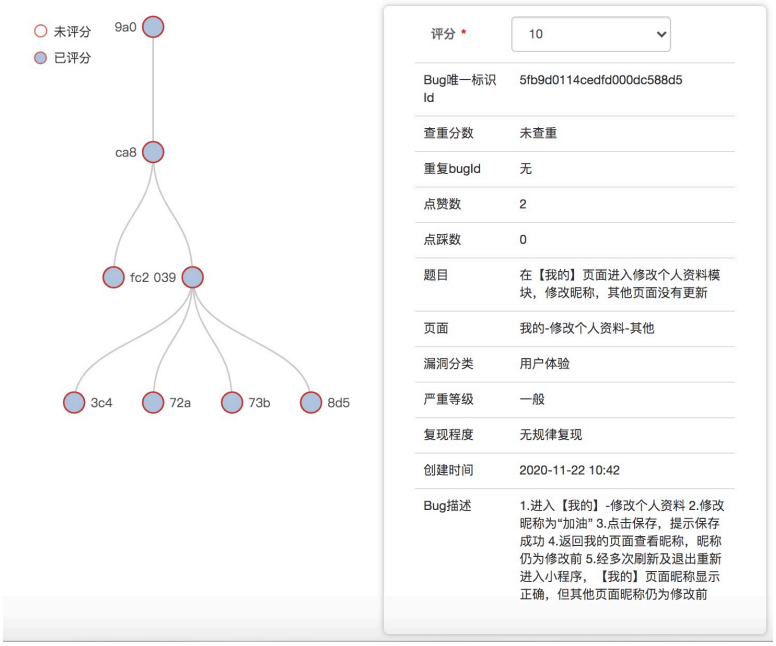

图 3.2: 树状报告示意图

系统依照翻译后的实体向量计算报告间实体相似度进行重复报告识别，将重复报告合并到同一报告簇。另一方面，原始报告中的图片数据经过特征向量提取并进行相似度计算后，如果报告间图片相似度大于阈值，并且实体相似度不低于指定阈值则合并到同一报告簇，二者结合共同生成报告簇。此外，系统将对于报告簇提供相似报告推荐的功能，系统会列举出所推荐的报告并列举出报告推荐的原因，整编人员可以选择将推荐报告加入到当前报告的报告簇中，作为自动化报告识别的补充，然后对报告簇进行报告融合。报告融合阶段，首先从知识图谱中提取报告簇的关系子图，然后使用 PageRank 算法对报告簇中的报告进行排序，确定报告中的主报告。系统对报告簇中的每一份报告的内容进行拆分，拆分的单位为单一句子和单一图片，对于和主报告中相似度低的句子/图片进行凝聚层次聚类，将相似的报告内容聚合在同一类簇中。系统对于最终生成的报告簇使用可视化的方式进行展示，并对报告内容中的差异观点进行颜色标记，报告融合过程中，系统还对报告簇中描述歧义的内容进行标明，如用户 A 使用某 H 牌手机出现“不显示头像”的缺陷，用户 B 使用 M 牌手机出现“正常显示头像，但背景图不显示”的缺陷，后者可能是对前者的补充或该缺陷仅发生在特定设备上，系统提取了报告簇中描述歧义的内容，并进行标明。方便整编人员进行信息获取，省去了整编人员手动比较两份不同报告差异点和歧义点的工作。该技术通过将描述同一缺陷的重复测试报告聚合在一起，对于合并在同一报告簇中的报告进行了内容的提取和分类，大大提高了整编人员进行报告整编的效率，加快了任务交付的时间，提高了交付报告的质量。

3.2　需求分析

3.2.1　功能性需求分析

如表3.1所示为系统的功能性需求分析。整编人员进入到系统内，首先要查看系统中的任务列表，系统将展示目前存在的任务，并用直观的方式展示出任务的基本完成情况。在任务列表中点击指定的报告可以查看任务的详细情况，其中包含用户提交的原始报告信息和任务总体的审核情况。

在众测任务完成之后，整编人员可以在系统中查看用户提交的树状报告，树状报告列表会展示树状报告簇内报告数及报告的主要信息，方便整编人员对该树状结构报告簇的主要信息有基本的认识。

树状报告详情页面将展示各报告的的主要内容，并从根节点到叶子节点逐条展示报告的基本信息，并将树状报告进行可视化展示。该页面可以对报告进行整编，并可以设置该树状报告的审核状态。

表 3.1: 功能需求列表

需求 ID	需求名称	需求描述
R1	查看任务列表	整编人员能够查看系统中已经存在的任务，能够看到任务的基本完成情况，对任务完成情况有直观的认识。
R2	查看任务详情	用户能够选择具体的众包任务查看其对应的众包任务详情，包含该任务提交的报告列表和任务总体审核情况。
R3	报告融合	系统基于众包测试报告数据，根据系统内的结构化数据和报告中提取的关系信息构建属于该任务的知识图谱，并对知识库内的报告实体进行向量翻译，对系统内的相似报告合并到同一报告簇中，其目的是将描述同一 BUG 的缺陷报告聚合在同一报告簇中。
R4	查看融合报告列表	整编人员能够查看经过报告融合处理后的众包任务的融合报告列表页，融合报告簇是系统进行报告簇合并后生成的。
R5	查看融合报告详情	用户能够选择具体的融合报告查看融合报告详情。详情主要内容包括主报告的内容、补充点及歧义点信息，对于报告内的重点信息进行高亮显示。该页面还对报告簇中知识图结构进行展示。
R6	详情页报告融合	在融合报告报告详情页，系统会推荐一些和当前报告簇相似程度较高的报告，整编人员可以决定是否将该报告加入到当前报告簇中，并可以对当前报告簇再次进行报告融合，作为自动化报告融合的补充。
R7	查看树状报告列表	用户能够查看某次任务中的树状报告列表，树状报告由众包工人在众包测试中通过协作的方式产生，结构如图3.2所示。
R8	查看树状报告详情	用户能够选择具体的树状报告查看树状报告详情。详情主要内容包括根节点报告的内容及子节点内容。
R9	报告整编	用户能在融合报告详情页、树状报告详情页基于页面内容整编新的报告，并可设置当前报告的审核状态。
R10	交付报告管理	用户可以将在整编过程中产生的交付报告进行导出，系统提供多种格式的报告导出，包括 HTML 网页类型和 Excel 表格格式。

整编人员在任务详情页面执行报告融合操作之后，系统启动对报告的融合处理，首先对原始文本报告进行 NLP 处理，提取报告中的关系，然后依照系统内存在的结构化数据及报告中的关系进行知识图谱构建，并引入第三方分类知识图谱进行补充，构建完成后使用知识图谱翻译模型进行翻译，得到报告之间的实体关系之后将相似程度高的报告进行报告簇合并，将相似程度高的的缺陷报告聚合在同一报告簇中。然后对报告簇内的报告进行分句，拆分成不同的补充点，对补充点信息进行聚类；并对报告簇内的报告依照知识图谱中所提取的实体关系进行歧义点识别，形成歧义项；对于报告簇中的报告对其所在的知识

图谱图结构中使用 PageRank 算法进行排序，确认报告簇中的主报告。

报告融合完成后，整编人员可以在系统中查看融合报告，融合报告列表会展示融合报告簇内的报告数以及报告的主要观点、补充点信息和歧义点信息，方便整编人员对该融合报告簇的信息有直观的理解和认识。

融合报告的详情页面将展示各报告的主要内容，并将融合报告进行可视化展示，该页面可以对报告进行整编并生成交付报告，并可以设置该融合报告的审核状态。该页面还将展示系统推荐的和当前报告簇相似程度较高的报告，整编人员可以选择将推荐报告加入到当前报告簇并再次进行融合，该页面还对报告在知识图谱中的图结构进行展示。

交付报告管理功能中可以看到交付报告的基本内容及报告中的图片，并可以将交付报告导出成 HTML 和 Excel 表格格式。

3.2.2　非功能需求分析

如表3.2所示是系统的非功能性需求。

表 3.2: 系统非功能需求列表

需求名称	需求描述
可用性	本系统应该具有较好的可用性，应对数据进行备份，如因进程崩溃、网络问题等意外情况导致系统不可用时，系统应及时重启当前实例，由其他实例继续提供服务。
可扩展性	本系统应对系统关键功能功能提供高层次抽象，做好系统功能模块之间的解耦，以方便后续进行功能上的扩展、算法上的的升级与改进。
易用性	系统界面设计得简洁高效、人机交互设计合理，使得没有经验的用户也能容易上手该系统。
可移植性	随着国产操作系统的不断完善及影响力的扩大，优秀的国产操作系统如 UOS、麒麟等也愈发的成熟，本系统应当在不同架构的国产操作系统服务器上也能运行。

系统需要保持较高的可用性，通过数据库主从备份的方式保证了数据库的可用性，通过使用 Nginx 负载均衡保证了系统内资源的优化分配，提高了系统应对风险的程度，保证了系统的可用性。同时，对于系统中关键的功能模块做好了抽象，对系统模块之间进行了解耦，方便系统在之后的升级改造中进行功能上的升级和模块上的替换，保证了系统的可拓展性。因系统面向真正的企业用户，因此在系统的界面的设计上追求简单高效，人机交互尽可能设计合理，无论是新手还是熟练使用系统的人员都能够高效地使用该系统，此外为了响应国产操作系统的号召，系统应当具备良好的可移植性，即系统可以在主流的国产操作系统上进行运行。

3.2.3　用例分析

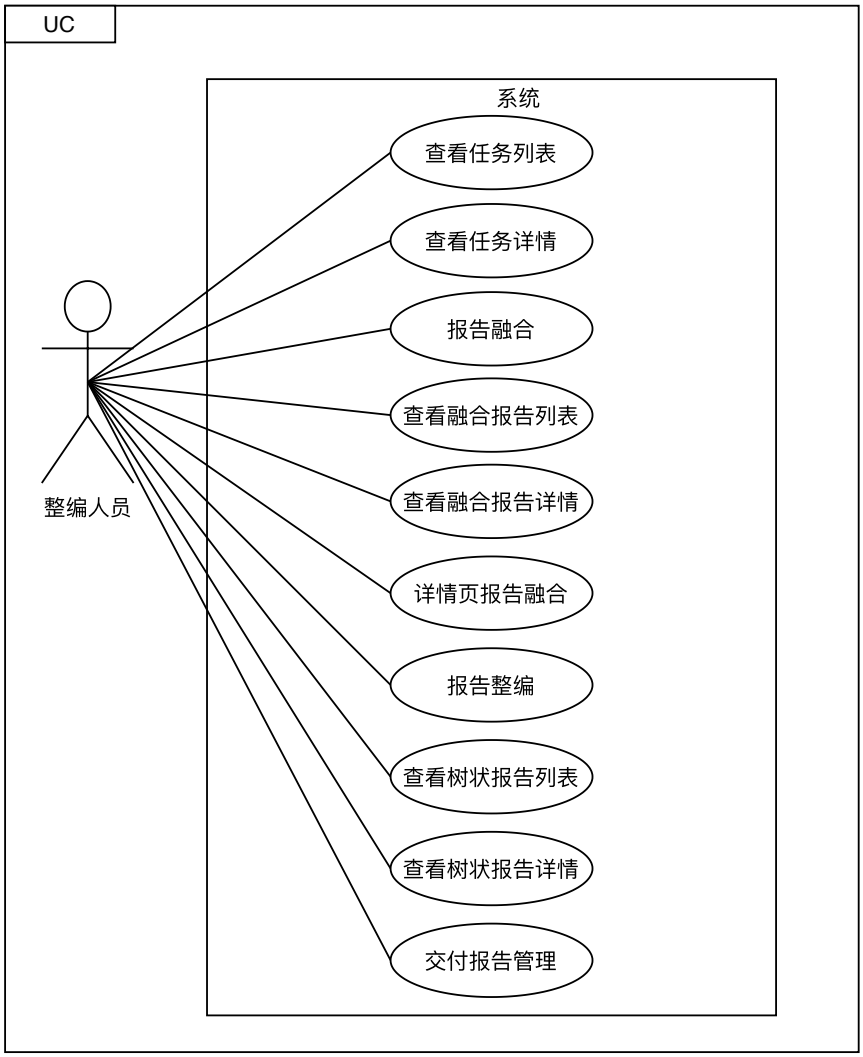

图 3.3: 系统用例图

根据上文中对系统功能性需求的分析，可以得到如图3.3所示的系统用例图，主要包含：查看任务列表、查看任务详情、报告融合、查看融合报告列表、查看融合报告详情、详情页报告融合、报告整编、查看树状报告列表、查看树状报告详情、交付报告管理这 10 个系统用例。

3.2.4　用例描述

查看任务列表是系统的入口，整编人员通过该功能可以对系统内的任务有大致的了解，对每个任务的完成情况有直观的认识。系统将任务列表页面的任务按照开始时间逆序的方式进行了排序，方便整编人员找到近期开展的众包任

务，此外任务列表页面还对具体任务的基本信息进行了展示，包括任务审核情况和任务名等信息。用例描述如表3.3所示。

表 3.3: 查看任务列表用例描述

ID	UC1
名称	查看任务列表
参与者	整编人员，目的是查看系统内的任务
触发条件	整编人员想要查看系统内的任务
前置条件	整编人员须已经被授权
后置条件	无
优先级	高
正常流程	1. 整编人员进入到系统 2. 系统展示所有任务，并将任务按照发布时间降序的方式进行排列展示

任务详情页展示了具体任务的详细信息，具体包括：任务名称、参与人数、缺陷报告数量、报告提交情况、页面覆盖情况、用户提交的报告列表等。系统对用户提交的原始报告进行了分页展示，并提供了关键字检索的功能，整编人员可以通过检索功能查找相关的报告，该页面为任务的基本信息提供了总览，该页面是单一任务的入口。用例描述如表3.4所示。

表 3.4: 查看任务详情用例描述

ID	UC2
名称	查看任务详情
参与者	整编人员，目的查看特定的任务信息
触发条件	整编人员选择一个特定的任务进行查看
前置条件	整编人员须已经被授权
后置条件	无
优先级	高
正常流程	1. 整编人员在任务列表选择一个任务进行查看 2. 系统显示任务的基本信息，并分页展示用户提交的缺陷报告 3. 整编人员输入关键字进行搜索 4. 系统显示检索后的数据 5. 整编人员按照特定的列进行排序 6. 系统按照特定的列进行升降序排序

报告融合功能是众测缺陷报告融合系统的核心功能，该功能由整编人员在任务详情页面上点击“报告融合”按钮触发。触发该功能后，系统在后台进行一系列对报告的自动化处理，包含知识图谱构建、知识图谱翻译、报告融合等过程。该功能一般耗时较长，报告自动化处理完成后系统自动更新报告在前端页面上的展示。用例描述如表3.5所示。

表 3.5: 报告融合用例描述

ID	UC3
名称	报告融合
参与者	整编人员，目的是对任务中的报告进行报告融合处理
触发条件	整编人员在任务详情页点击“报告融合”
前置条件	整编人员须已经被授权
后置条件	报告融合过程中产生的相关数据存入到数据库中
优先级	高
正常流程	1. 整编人员在报告详情页点击“报告融合”按钮 2. 系统进行报告融合操作。融合完成后在报告详情页面进行更新。
异常流程	2a. 系统出现异常导致融合失败 1. 系统提示报告融合异常，并给出对应的原因

等待报告融合功能执行结束后，整编人员可以在系统中查看融合报告列表。融合报告列表中的每一项代表一个融合报告簇，报告簇在系统中使用卡片的形式进行展示，每个卡片代表一个融合报告簇。报告簇展示了该报告簇所融合的报告数量及该报告簇主报告、补充点和歧义点信息，点击报告簇中的展开按钮可以查看补充点、歧义点信息。用例描述如表3.6所示。

表 3.6: 查看融合报告列表用例描述

ID	UC4
名称	查看融合报告列表
参与者	整编人员，目的查看某个任务下的融合报告列表
触发条件	整编人员在一个任务中点击查看融合视图
前置条件	该任务的报告融合功能已经被整编人员触发
后置条件	无
优先级	高
正常流程	1. 整编人员在报告详情页面点击报告融合视图 2. 系统显示融合报告列表并展示融合报告簇的基本信息 3. 整编人员选择报告审核状态进行筛选 4. 系统显示指定审核状态的融合报告 5. 整编人员对一个融合报告点击展开按钮 6. 系统显示该报告簇其他报告的信息

融合报告详情页将展示系统在触发报告融合后生成的融合报告信息，其内容包括主报告、补充点和歧义点信息。其中报告中的重要信息采用不同颜色的方式进行突出展示，方便整编人员获取到报告的关键点。该页面使用树状结构展示了该融合报告的报告树结构关系以及和报告簇相关的知识图谱子图结构，整编人员可以查看当前报告簇在众测知识图谱中的结构信息。在该页面整编人员

可以进行报告整编工作，即根据当前报告簇所展示的信息、归纳整理并生成最终的交付报告。在该页面，整编人员能够设置当前报告簇的审核状态。用例描述如表3.7所示。

表 3.7: 查看融合报告详情用例描述

ID	UC5
名称	查看融合报告详情
参与者	整编人员，目的查看某个融合报告的详细信息
触发条件	整编人员选择一个融合报告进行查看
前置条件	整编人员须已经被授权
后置条件	无
优先级	高
正常流程	1. 整编人员在融合报告列表选择一个融合报告进行查看 2. 系统显示融合报告主报告的基本信息 3. 整编人员点击补充点展开按钮 4. 系统显示报告补充点信息 5. 整编人员点击歧义点展开按钮 6. 系统显示歧义点信息 7. 整编人员点击查看子报告 8. 系统显示子报告的详细信息 9. 整编人员填写报告信息生成交付报告 10. 系统添加一条交付报告并提示添加成功 11. 整编人员设置报告为已审核 12. 系统将该报告的审核状态设置为已审核

表 3.8: 详情页报告融合用例描述

ID	UC6
名称	详情页报告融合
参与者	整编人员，目的是对某一报告簇进行报告融合
触发条件	整编人员在详情页点击“融合当前报告簇”
前置条件	整编人员须已经被授权
后置条件	无
优先级	高
正常流程	1. 整编人员进入到融合报告详情页 2. 系统显示和当前报告簇相似程度高的报告，并说明理由 3. 整编人员选中一份相似报告并点击“加入当前报告簇” 4. 系统显示将该报告加入到当前报告簇 5. 整编人员点击“融合当前报告簇” 6. 系统提示等待，并在融合完成后显示融合后的内容，同融合报告详情页

在融合报告详情页，系统会推荐不在当前报告簇中的相似报告，并给出推

荐该报告的理由，整编人员可以选择将推荐的报告加入到当前报告簇中，作为自动化重复报告识别的补充，并可再次触发报告融合操作，仅仅融合当前报告簇。触发融合后，系统在后台进行自动化处理，针对当前报告簇进行主报告提取、补充点分析和歧义点分析。报告簇融合过程结束后，当前页面展示报告融合的结果，以供整编人员生成交付报告。用例描述如表3.8所示。

表 3.9: 查看树状报告列表用例描述

ID	UC7
名称	查看树状报告列表
参与者	整编人员，目的查看某个任务下的树状报告列表
触发条件	整编人员在一个任务中点击查看树状视图
前置条件	该众包测试任务已完成
后置条件	无
优先级	高
正常流程	1. 整编人员在报告详情页面选择查看报告树状视图 2. 系统显示树状报告列表并展示树状报告根结点报告的基本信息 3. 整编人员对一条树状报告点击展开按钮 4. 系统显示该树状报告其他报告的信息

众包任务完成后，整编人员可以在系统中查看树状报告列表。树状报告是众包工人在协作式众测中通过对其他报告进行补充产生的树状结构报告簇，报告树中的子节点可以看成是对根节点的补充描述。该功能在任务结束后即可使用。树状报告列表中的每一项代表一个树状报告集合，展示了该报告集合的根节点报告信息，点击展开可以查看子报告的报告内容。用例描述如表3.9所示。

表 3.10: 查看树状报告详情用例描述

ID	UC8
名称	查看树状报告详情
参与者	整编人员，目的查看某个树状报告的详细信息
触发条件	整编人员选择一个树状报告进行查看
前置条件	整编人员须已经被授权
后置条件	无
优先级	高
正常流程	1. 整编人员在树状报告列表选择一个树状报告进行查看 2. 系统显示树状报告的基本信息 3. 整编人员填写报告信息生成交付报告 4. 系统添加一条交付报告并提示添加成功 5. 整编人员设置报告为已审核 6. 系统将该报告的审核状态设置为已审核

树状报告详情将展示用户在系统中生成的树状报告信息，包括了报告树中节点的报告信息。系统可视化展示了树状报告之间的关系，在该页面用户可以进行报告审核和报告整编工作。用例描述如表3.10所示。

表 3.11: 报告整编用例描述

ID	UC9
名称	报告整编
参与者	整编人员，目的对系统内的报告进行报告整编
触发条件	整编人员现在处在融合报告或树状报告页面
前置条件	整编人员须已经被授权
后置条件	无
优先级	高
正常流程	1. 整编人员查看报告信息并填写交付报告信息 2. 系统展示用户所填写的交付报告信息 3. 整编人员点击保存 4. 系统将该交付报告保存并提示用户保存成功 5. 整编人员点击删除 6. 系统删除该报告并提示用户删除成功 7. 整编人员设置报告审核状态 8. 系统变更报告审核状态并提示

用户在融合报告、树状报告详情页面可以使用报告整编功能，其功能是整编人员通过在当前页面浏览报告信息，归纳并总结成最终的交付报告。文字通过键盘键入，图片通过拖拽原始报告图片进行上传。用户还可以对该页面的交付报告进行编辑和删除，并设置报告审核状态。用例描述如表3.11所示。

表 3.12: 交付报告管理用例描述

ID	UC10
名称	交付报告管理
参与者	整编人员，目的对系统内的交付报告进行管理
触发条件	整编人员在任务详情页点击查看交付报告
前置条件	整编人员须已经被授权
后置条件	无
优先级	高
正常流程	1. 整编人员在任务详情页面点击查看交付报告 2. 系统显示所属该任务的交付报告列表 3. 整编人员输入关键字进行搜索 4. 系统显示检索后的数据 5. 整编人员选择导出 HTML(Excel) 格式的交付报告 6. 系统开始下载 HTML(Excel) 格式的交付报告

交付报告为整编人员最终生成的用于交付的报告，是整编人员基于原始报告整理而成的。交付报告管理功能能够对交付报告进行基本的管理，例如进行查看、删除等等。该页面可以对交付报告进行筛选，并可以对交付报告进行不同格式的导出，包括 HTML 格式和 Excel 格式。用例描述如表3.12所示。

3.3　总体设计

3.3.1　总体架构设计

基于知识图谱的众测报告融合系统是一个独立的 Web 服务平台，总体来说主系统基于 Spring Boot 和 Thymeleaf 进行开发，知识图谱部分使用 Python 编写并封装成独立服务，知识图谱结构使用 Neo4j 进行存储，服务间使用 Thrift 进行通讯。系统架构如图3.4所示，下面就系统中的主要模块进行描述。

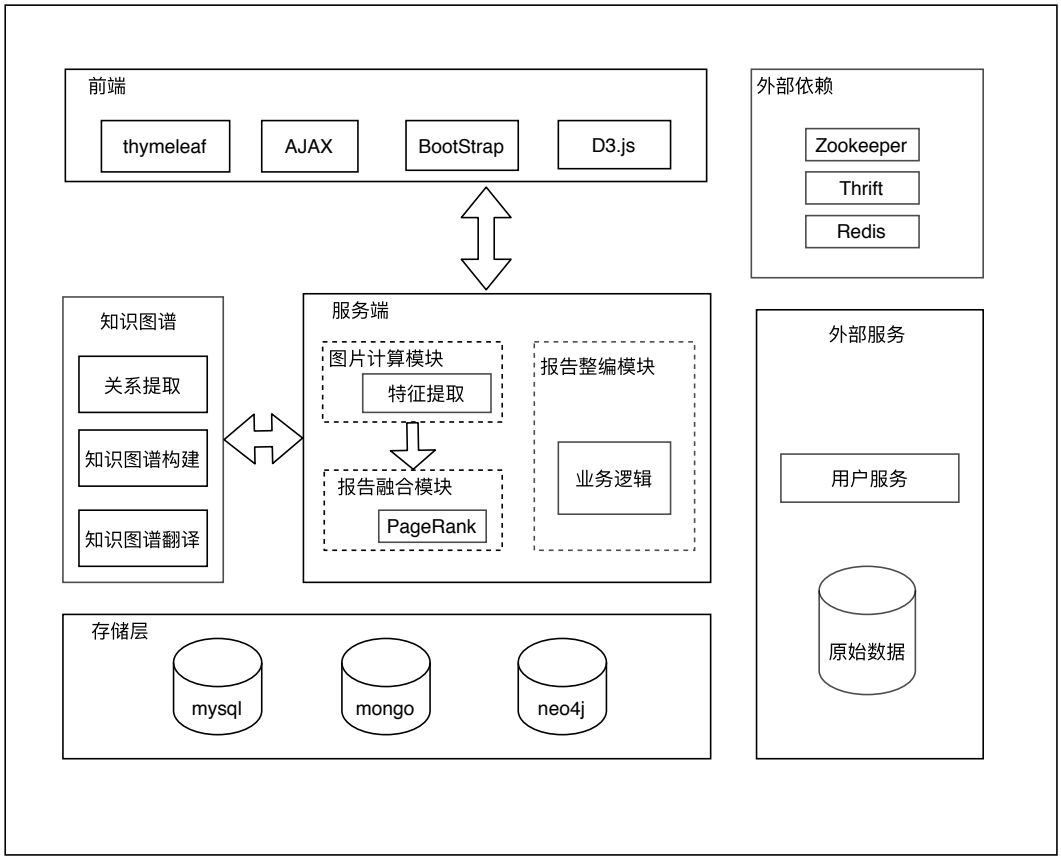

图 3.4: 系统架构图

前端方面，本系统采用 Bootstrap 和 jQuery 进行开发，使用了 Spring 中的 Thymeleaf 模板引擎。Thymeleaf 能够简单高效的进行开发前端页面，能够应对

复杂的处理逻辑，在数据量大的情况，Thymeleaf 能够很快的完成数据渲染，减轻 JavaScript 渲染的负担，避免了复杂 JS 代码的编写。前端 UI 组件库选用的是 Bootstrap，这是一个成熟的界面展示框架，有着上手方便、资源丰富及样式美观的优点。系统中的部分图表使用 D3 组件来绘制。在前端和后端的交互方面，系统中采用 Ajax 与后端进行异步请求，进一步提高系统的性能。

本系统的主体后端部分主要采用的是 Spring Boot 框架，该框架在编码、配置和部署时有着简洁高效的优点，得益于其自身丰富的功能和国内外强大的开发社区支持，该框架有着资源丰富、易于学习和开发的特点，并能和许多独立第三方组件进行集成，使用 Spring Boot 使得开发者能够关注逻辑代码本身。系统使用 Thrift 与一些第三方服务及知识图谱部分进行交互，Zookeeper 被用来做项目的配置管理和服务注册及发现，使得热修改系统的配置成为可能。在数据的存储方面，系统主要使用 Mysql 和 Mongo 来进行数据的持久化，知识图谱部分的数据存储在图数据库 Neo4j 中。知识图谱部分后端使用 Python 语言进行编写，作为单独的服务与系统进行交互。知识图谱部分使用了 NLP 技术进行了关系的提取，使用 TransE 模型进行了知识图谱向量化翻译。

此外，本系统引入了 Redis 作为系统的缓存，作为一个高效的缓存组件，相比于 Guvva 等程序内缓存，Redis 具有更好的性能和更强的拓展性，有利于系统后续进行升级和拓展，提高了系统的性能及可拓展性。

3.3.2　模块划分

如图3.4所示，依照高内聚低耦合的思想本系统可以分为：知识图谱模块、图片计算模块、知识图谱报告融合模块、报告整编模块这 4 个模块。在知识图谱构建及应用上知识图谱模块可以分解为关系提取、知识图谱构建及知识图谱翻译三个部分，关系提取主要是对众测任务中用户提交的原始报告的内容进行关系提取；知识图谱构建部分将对报告内容提取的关系及系统内存在的结构化数据进行知识图谱的构建，并引入了通用的分类知识图谱对众测知识图谱进行补全，其中图数据存储在 Neo4j 图数据库中；相似报告识别主要是利用 TransE 模型对知识图谱内的实体及关系进行翻译，将实体和关系都翻译成低维向量，使用余弦相似度方法计算报告实体之间的相似性，对于相似度高的报告合并到同一类簇中。知识图谱模块使用 Python 语言进行开发，其使用 Thrift 框架同主服务进行通信。图片计算模块主要是对原始报告中的图片进行特征提取，将图片转化成向量并使用余弦相似度计算图片之间的相似度关系，将图片相似度高于阈值的报告进行标记。知识图谱报告融合模块中，首先对于报告簇内的报告获取知识图谱中的报告簇子图结构，然后使用 PageRank 对图中的报告节点进行排

序，确认报告簇内的主报告，并对报告簇内的其他报告进行补充点拆分，拆分的粒度为单一句子或图片，将句子或图片与主报告进行比较，将相似度低的补充点进行聚类，形成补充项，然后对报告簇内的描述内容进行歧义点提取，将在不同报告中描述同一内容但相悖的文本合并成歧义点。报告整编模块实现的主要是系统的业务代码，主要实现报告整编的流程，是系统的入口，负责同前端进行交互,包括查看系统任务列表、查看融合报告、报告整编、交付报告管理等功能。

3.3.3　总体设计

本章节分别从逻辑视图、进程视图、开发视图和物理视图的角度来描述并分析系统的总体设计。系统的逻辑视图如图3.5所示，逻辑视图面向的是系统的用户，本系统中为整编人员，描述的是系统提供给用户的服务，这里用核心类图进行展示。

FusionService 是报告融合的核心类，它和许多类进行交互来完成报告融合的工作，其中 KGService 用于获取知识图谱模块识别的重复报告报告簇，其通过 Thrift 接口调用知识图谱模块所获得。ImgService 分析报告中的图片，计算与其他报告中图片的相似度信息，并就图片相似度高的报告进行重复报告识别。SupplementService 提供了融合报告中补充点报告的管理功能。AmbiguityService 提供了融合报告中歧义点报告的管理功能。ClusterAnalyseService 提供聚类的功能，在本系统中主要是对补充点信息进行聚类。MainReportService 提供了融合报告簇中主报告的管理功能，TaskService 提供了对系统内众包任务的查询和管理功能。DeliveryRepoortService 则为系统内的交付报告提供了管理功能。SourceBugService 主要是对众包系统内用户提交的原始报告提供查询功能。

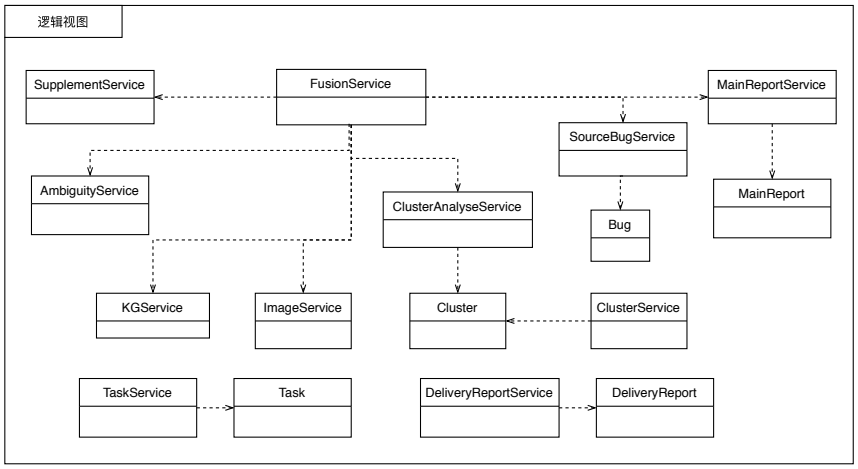

图 3.5: 逻辑视图

如图3.6所示是系统的进程视图，主要描述了系统运行相关的进程及进程之间的交互。

当系统运行时，系统主线程启动，首先向 Zookeeper 拉取配置信息并完成服务的注册和发现。启动之后主线程向外部服务进行同步调用获取任务和报告的数据。当融合服务触发时，主线程调用异步线程执行融合操作，异步线程启动计算并调用知识图谱获取报告簇信息，知识图谱构建和计算过程中产生的数据存入到数据库中。融合过程执行后，系统产生的融合报告和相关数据存入到数据库中，异步线程回调主进程，主进程将执行结果缓存到 Redis 中。

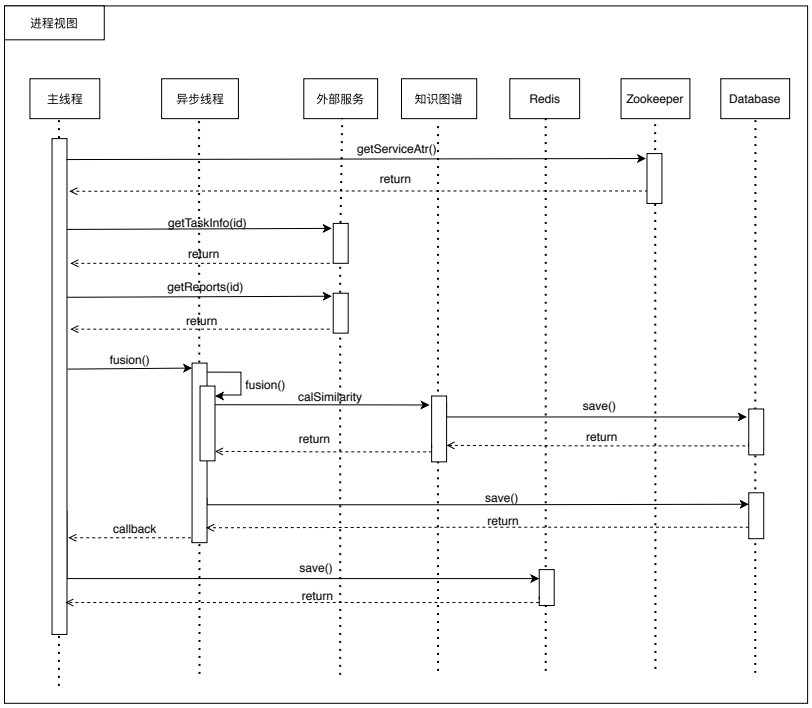

图 3.6: 进程视图

如图3.7所示是系统的开发视图，开发视图面向的是系统的开发人员，其中 UI 部分主要分为前端模板、CSS 样式和 JavaScript 资源文件。逻辑代码部分主要分为控制器（Controller）和逻辑服务（Service）两个大的部分，Controller 部分主要面向的是外部的 HTTP 调用，主要包括 TaskController、MainReportController、BugReviewController 及 DeliveryRportController。Service 部分封装了系统逻辑实现的代码。其中 FusionService 主要进行报告融合，其调用 KGService 和 ImgService 获取报告簇合并信息。融合过程中调用 SuplementService 和 AmbiguityService 分别获取报告簇的补充点和歧义点信息。FusionService、TaskService、DeliveryReportService 及 SourceBugService 主要为控制器提供逻辑代码的入口。

Tech Service 为系统的基础技术服务，为系统的提供技术支撑，主要包括进行 RPC 调用的 Thrift，对数据库接口进行封装的 SpringDataJPA，日志服务 Log4j 及为系统提供缓存服务的 Redis。

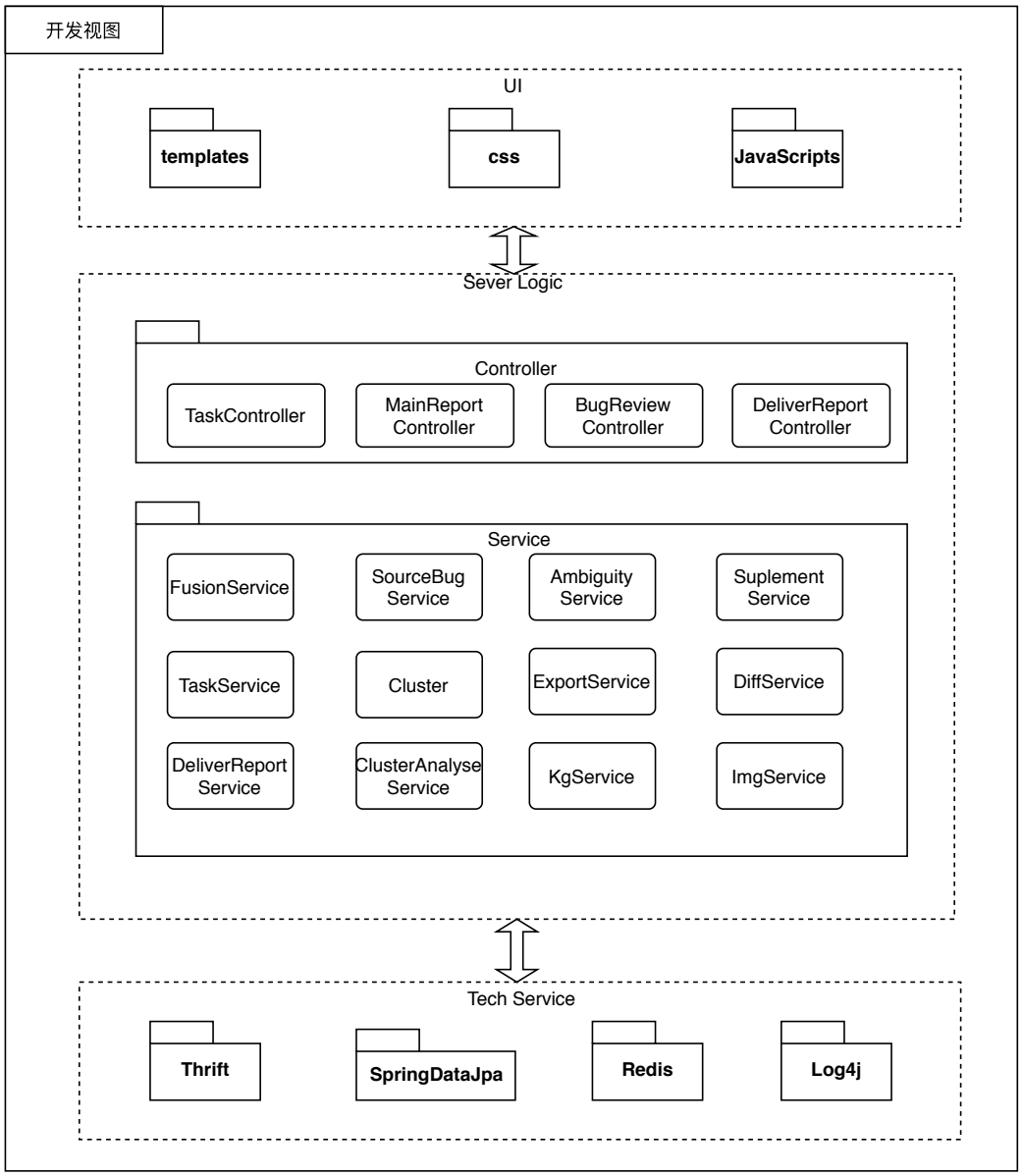

图 3.7: 开发视图

如图3.8所示是系统的物理视图，物理视图从系统运维人员的角度出发、描述了系统关键节点之间的部署和通讯情况。用户可以通过浏览器对系统发起 HTTP 请求调用，HTTP 请求经过防火墙过滤后到达服务端，服务端的 AppSever 进行逻辑处理，系统执行中缓存数据存储在 Redis 服务器上，应用服务器和 Redis 服

务期间通过 Redis 协议进行通讯。数据库服务是单独的服务，其和应用服务器通过基于 TCP 的 SQL 查询语言通讯。从数据库和主数据库件通过 Binlog 进行主从复制，保证了系统的可用性。

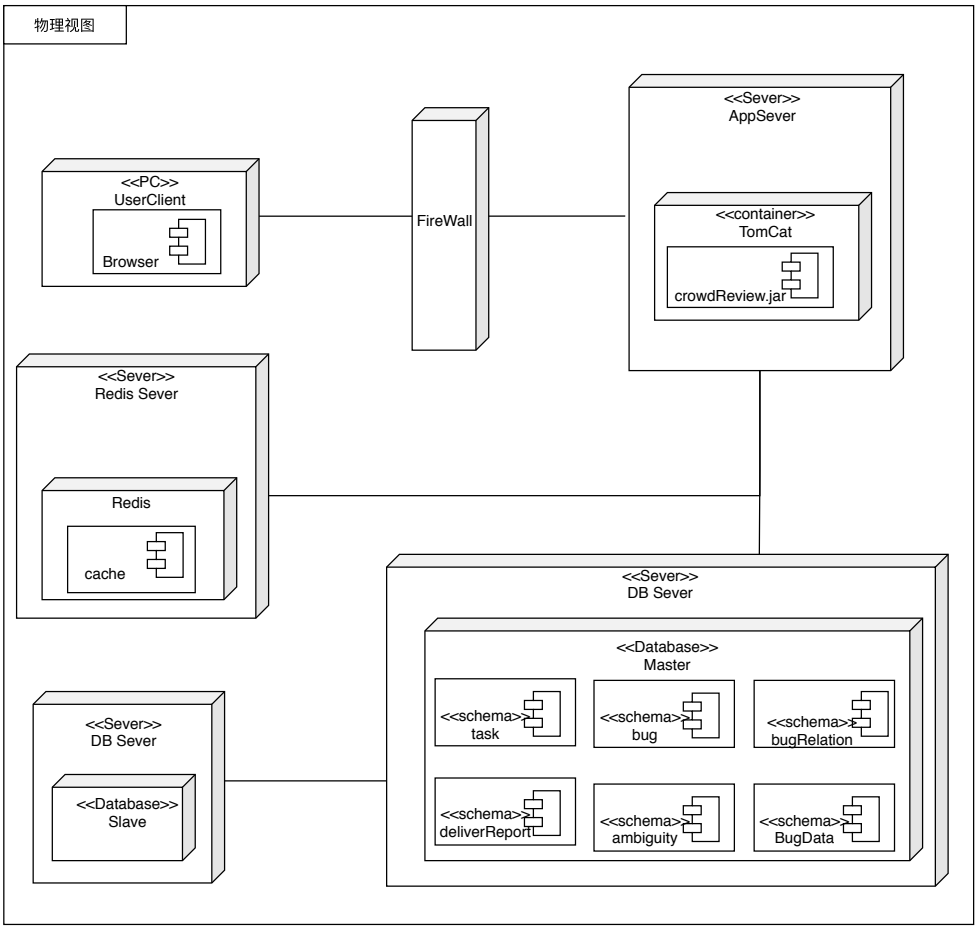

图 3.8: 物理视图

3.4　知识图谱模块设计

3.4.1　架构设计

如图3.9所示是知识图谱模块的架构设计，知识图谱模块的输入是用户在众测任务中产生的原始报告，该模块的目的是通过处理系统内的众测报告数据生成任务知识图谱。众包测试系统中主要有两种报告，一种是普通报告，即和其他报告没有关联的报告，另一种则是树状报告，用户通过 fork 操作指定某报告作为父报告，所填写的报告是对父报告的补充，形成树状结构的报告。我们可以认为，树状报告中的报告描述的是同一个系统缺陷。因系统中主要是用户信

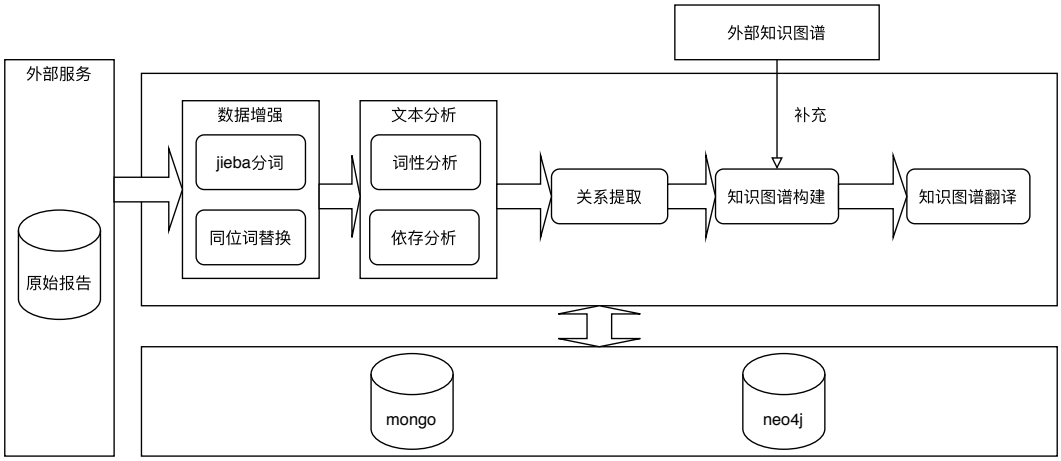

图 3.9: 知识图谱模块架构设计

息、报告点赞点踩数据、任务信息、三级页面信息、树状报告父子关系等结构化数据，结构化数据关系结构提取较为容易，因此仅对报告文本内容进行关系抽取。对报告内容关系抽取的流程为：首先使用 Jieba 对报告进行分词，并使用同位词文件对报告内容进行同位词替换进行数据增强。然后将报告内容进行分句，对每一句进行分词并进行词性分析及依存分析，生成句子的依存句法树，因报告中主要为短句且有较多的“点击”，“输入”，“进入”等操作，因此可以对报告的谓语词进行分析并抽取出对应的关系如报告 → 点击 → 按钮、报告 → 进入 → 首页等。针对报告的文本数据进行自然语言处理抽取出关系信息后将处理结果同系统中的用户、任务、点赞点踩等数据一同构建该次众包任务的知识图谱，同时对于提取出的实体，引入外部分类知识图谱对众测知识图谱进行补充。之后使用翻译模型将知识图谱内的实体和关系翻译成向量。对于翻译出来的报告实体向量进行比较，如果两个报告之间相似程度高于阈值，视为重复报告，对重复报告进行合并，聚合到同一报告簇中。

3.4.2　关系提取

关系提取主要针对报告文本描述内容，系统中的其他结构数据均为结构化数据。关系提取主要使用了 NLP 技术。报告文本内容首先经过同位词替换，同位词替换能够减少差异化描述对关系提取的影响。例如句子“首页展示异常”和“主页显示异常”描述的事件和意思相同，但是用词不同，同位词替换就是将描述相近的词语进行替换，从而提高文本描述风格的一致性，同位词规则中“首页”和“主页”统一为“首页”，“展示”和“显示”统一为“显示”，如上两句经过同位词替换后都为“首页显示异常”，从而提高了文本描述风格的一致性，达

到了数据增强的效果。

经过同位词替换后，针对句子的描述内容进行分词，本文提供了 Jieba 和 Pyltp 两种分词模型，默认使用 Jieba 对句子描述进行分析。分词即将句子拆分成单个的词语，这是进行 NLP 分析的第一步。句子经过分词拆分成词语后首先进行词性标注，即在给定的句子中判断句中词语的词性，确定词语在当前句中的词性并标注，如在“系统没有提示异常”一话中，分词模型会将该句划分为“系统”、“没有”、“提示”、“异常”这四个词，词性标注中会将这四个词分别标记为名词、副词、动词和名词。经过分词和词性分析后进行依存句法分析。如图3.10所示为“系统没有提示异常”这段话的依存句法分析树，可以提取出该句的主语为“系统”，宾语为“异常”，谓语是“提示”，“没有”为修饰谓语的副词。本文对谓语的副词进行了否定词分析，对于“没有”、“无法”、“不能”等词汇进行标记，如上句中可以提取的关系为“系统”→unshow→“异常”。

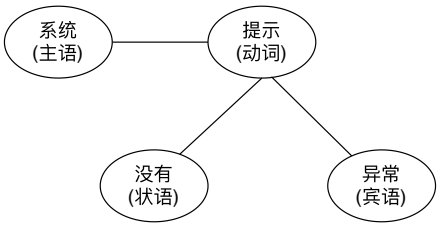

图 3.10: 依存句法树

3.4.3　知识图谱构建

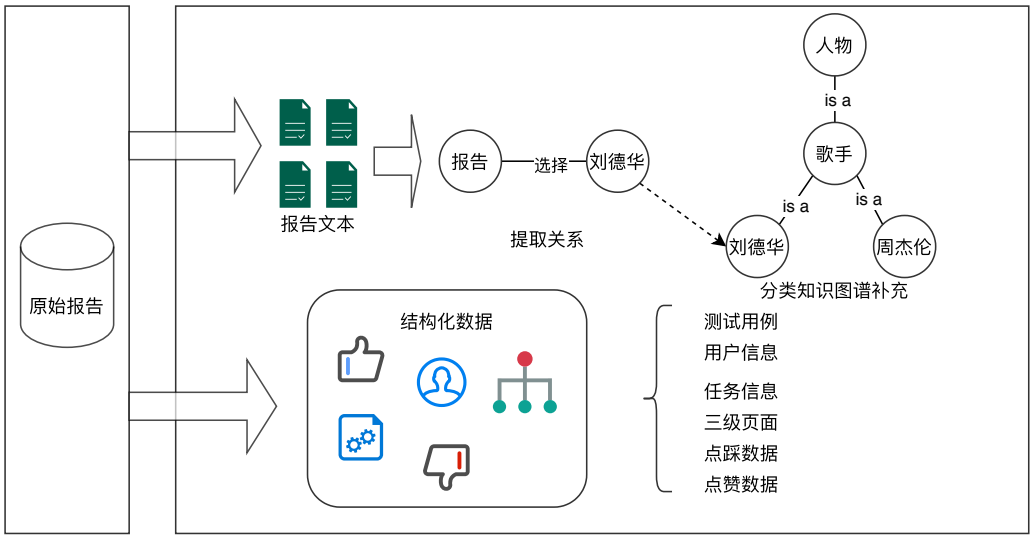

图 3.11: 知识图谱构建流程

知识图谱构建是将系统内的结构型数据库和经由关系提取阶段提取的关系信息加载到数据库中，并引入相关的分类知识图谱进行补全。知识图谱构建流程如图3.11所示。系统内众测任务原始报告的结构化数据主要包含：用户信息、缺陷报告、测试用例、任务信息、工人互动信息等。对于结构化数据，直接将关系保存在知识图谱中，如“任务”→包含→“报告”，“用户 1”→点赞→“报告 2”等。缺陷报告文本描述内容经过关系提取后，关系本身直接存储到知识图谱中，针对所提取的实体引入外界分类知识图谱进行补全。在中文概念知识图谱中，CNProbase 是目前规模最大的开放领域中文概念图谱及概念分类体系 [40]。如音乐类 App 测试中，Bug1 描述为“歌手选择刘德华”。所提取的用户操作“选择”→“刘德华”，其中“刘德华”实体在 CNProbase 中上级为歌手、演员、人物等。Bug2 描述“歌手选择周杰伦”，可以建立起“周杰伦”→“歌手”←“刘德华”的联系。从而在知识图谱翻译阶段获得较高的实体相似度。如图3.12所示为系统构建的知识图谱示例。

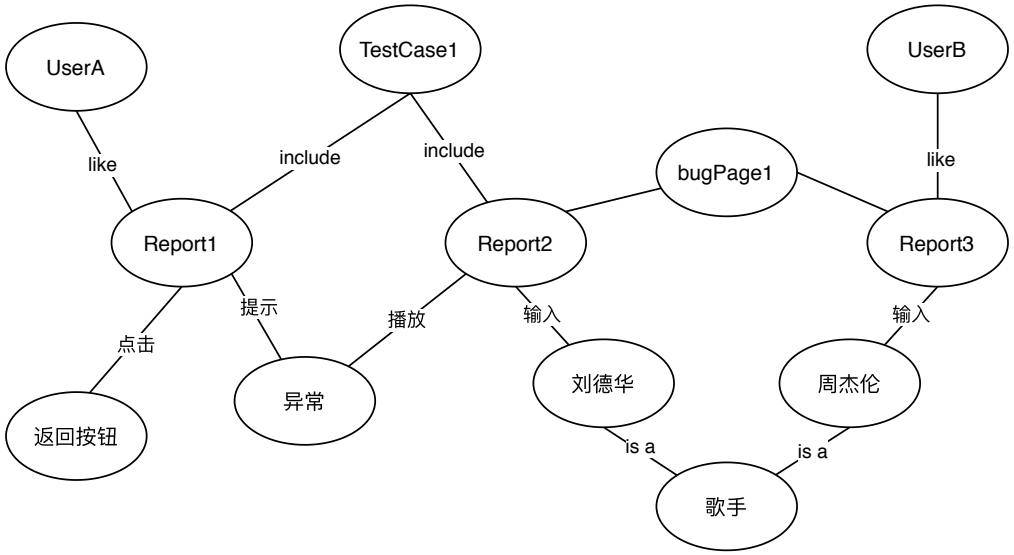

图 3.12: 知识图谱图结构示例

3.4.4　知识图谱翻译

如图3.13所示是 TransE 向量翻译算法的运行过程。TransE 算法对知识图谱中的实体和关系进行分布式向量表示，其将图谱中的每个三元组信息 (头节点, 关系, 尾节点) 中的“关系”看成是头节点到尾节点的一次翻译。在初始阶段，实体和关系向量会生成随机值，通过不断的调整实体和关系向量的值，经过多次迭代后得到最接近知识图谱图结构的关系和实体的向量表示。

TransE 算法流程

输入　训练集合 $S=\{(h,\ell,t)\}$ 实体集合 E，关系集合 L，间距 γ，向量维度 k.

1: 初始化 $\ell \leftarrow$ 向量取值$\left(-\frac{6}{\sqrt{k}},\frac{6}{\sqrt{k}}\right)\ell \in L$

2: $\ell \leftarrow \ell/\| \ell \| \ell \in L$

3 $e \leftarrow$ 向量取值$\left(-\frac{6}{\sqrt{k}},\frac{6}{\sqrt{k}}\right) e \in E$

4：进入循环

5: $e \leftarrow e/\| e \|, e \in E$

6: 取出大小为 b 的样本 S_{batch}

7: 初始化三元组关系 T_{batch}

8: 对于 $(h,\ell,t) \in S_{\text{batch}}$ 执行:

9: $(h',\ell,t') = \left(S'_{(h,\ell,t)}\right)$// 使用迭代后的三元组进行更新

10: 更新$T_{\text{batch}} = T_{\text{batch}} \cup \{((h,\ell,t),(h',\ell,t'))\}$

11: 结束循环

12: 更新向量 w.r.t $\sum_{((h,\ell,t),(h',\ell,t'))\in T_{\text{batch}}} \nabla[\gamma + d(h+\ell,t) - d(h'+\ell,t')]_+$

13：结束循环

图 3.13: TransE 算法流程

3.4.5　详细设计

如图3.14所示是知识图谱模块的核心类图设计。

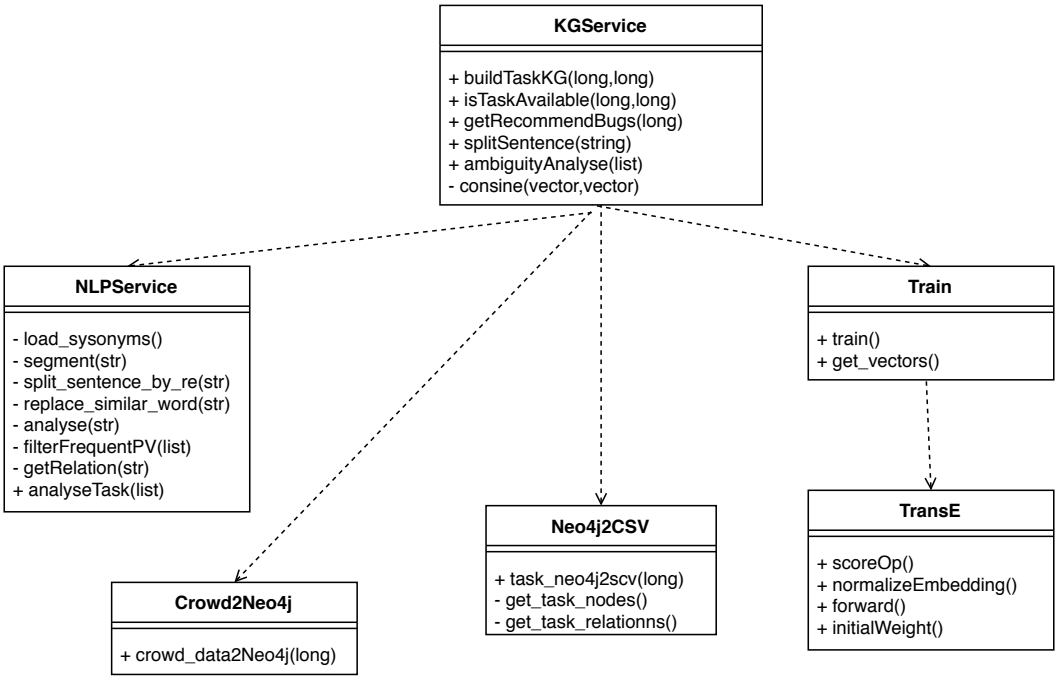

图 3.14: 知识图谱模块核心类图

该模块使用 Python 进行编写，与系统主模块之间使用 RPC 框架 Thrift 进行

交互。其中 KGService 负责与其他模块进行交互，是系统的入口，也负责了模块内部的交互，是该模块的控制中心。NLPService 主要是对报告中的描述文本进行自然语言处理。模块提供了分词、词性识别、依存关系分析等功能，主要目的是提取报告文本内容中的关系。Crowd2Neo4j 主要是使用系统内的用户信息、点赞点踩数据、任务信息及 NLPService 中分析得到的报告内部关系进行知识图谱的构建，并引入外部分类知识图谱进行补充，其中实体和关系存储在图数据库 Neo4j 中。Neo4j2CSV 主要是读取该场任务所构建的知识图谱内的实体和关系并将数据导出为知识图谱翻译模块中的所需要的 CSV 格式。Train 类主要负责将知识图谱使用翻译模型翻译成向量，其中使用的翻译模型是 TransE 模型，模型翻译的结果由 KGService 进行比较，判断未在同一树状结构下的报告之间是否具备较高的实体相似度，将实体相似度高的报告合并到同一报告簇中。

3.4.6　数据库设计

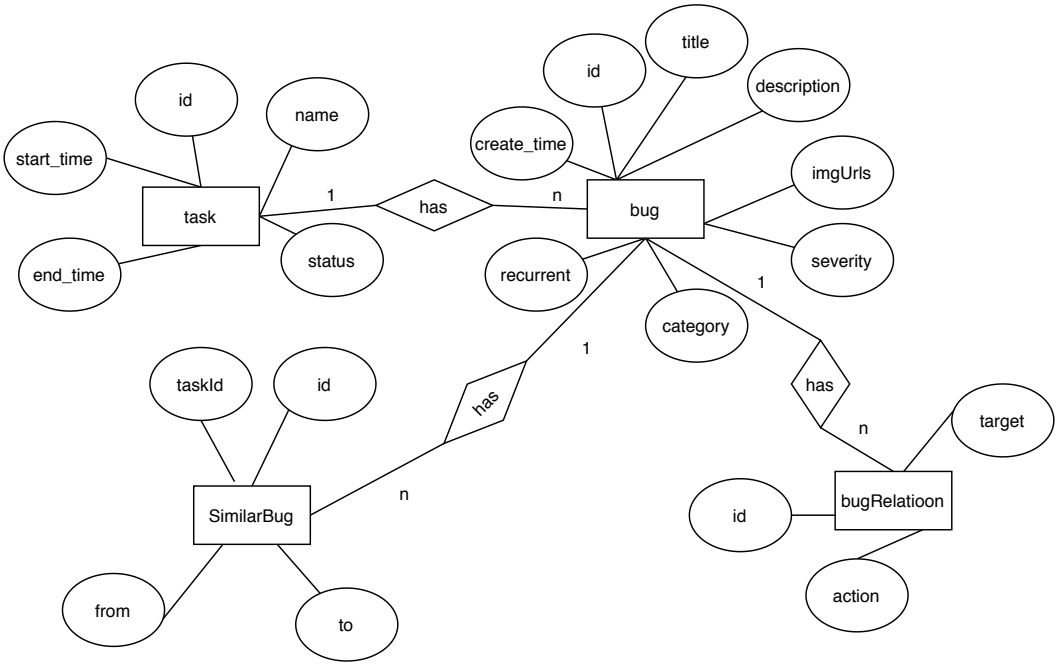

图 3.15: 知识图谱模块数据库 ER 图

如图3.15所示是知识图谱模块的 ER 图设计，主要描述了和该模块相关的数据库实体及实体间关系，task 表为系统中的众包任务，该任务数据是从外部服务处所获得的。bug 表中内容为用户在众测任务中提交的原始报告，同样是外部数据处所获得的。BugRelation 表中存储的是 NLP 处理中在报告内容中所提取的

关系信息，主要描述了操作的类型和操作的对象，包含用户操作和系统的反馈等。SimilarReport 表描述的是经过向量翻译计算后的报告相似信息，记录的是报告之间实体相似度。其中 task 表和 bug 表是一对多的关系，一个 task 中有多条 bug 记录，bug 表和 BugRelation 表是一对多关系，一个 bug 可以提取出多条关系，bug 表和 SimilarBug 表也是一对多关系，一个 bug 可以和多个 bug 实体进行相似度比较。

如表3.13所示是 BugRelation 表字段的详细说明，该表主要记录了 NLP 阶段所在报告中挖掘的关系信息，主要记录的是报告中所描述的用户的操作和系统的反馈。包含了 id、taskId、reportId、action 和 target 字段。下面对表中的字段进行详细说明。

表 3.13: BugRelation 表

字段	含义	类型	说明
id	唯一 id	BIGINT	id 为该表的主键
taskId	该记录所处的任务的 id	BIGINT	不可为空
reportId	该关系所处的报告的 id	BIGINT	不可为空
action	报告中所描述的用户的操作或系统的表现	VARCHAR	举例：“click”，“show”
target	action 操作所对应的目标	VARCHAR	举例：“首页”，“按钮”

如表3.14所示是 SimilarBug 表字段的详细说明，该表主要记录了经过知识图谱向量翻译后的报告实体相似度信息。主要包含了 id、taskId、from 和 to 字段。下面对表中的字段进行详细说明。

表 3.14: SimilarReport 表

字段	含义	类型	说明
id	该表唯一 id	BIGINT	id 为该表的主键
taskId	该表中报告所属的任务的 id	BIGINT	不可为空
from	相似报告之一，对应的是树状报告中的非根节点报告，此处记录的为报告的 id	BIGINT	不可为空
to	相似报告之一，对应的是树状报告中的根节点报告，此处为记录的为报告的 id	BIGINT	不可为空

3.5　图片计算模块设计

3.5.1　架构设计

图片计算模块的架构设计如图3.16所示，图片计算模块的输入为用户在任务中产生的原始报告，因为原始报告中存储的用户上传的图片为图片 URL 信息，

因此报告中的图片数据需要从服务器进行下载。该模块将图片数据加载到本地后对图片特征进行提取，图片特征选用的是 JCD 特征，该特征是对 CEDD 特征和 FCTH 的一个综合，考虑了图片的纹理、色彩和色块边界。根据提取之后的照片特征得到图片向量，对图片向量进行余弦相似度计算可以得到相似图片集合，对相似比例高于特定数值（目前是 0.8）且所属报告三级页面相同的图片进行标记，对于此类报告查询其报告实体相似度，在实体相似度不低阈值的情况下，将相似报告合并到同一报告簇中。

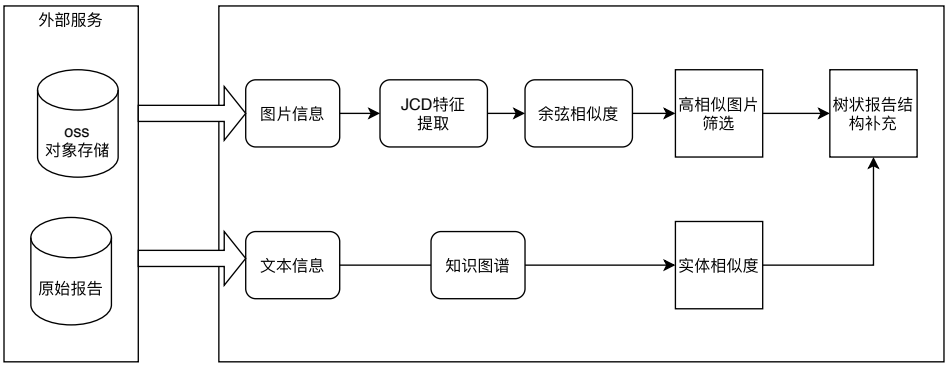

图 3.16: 图片计算模块架构图

3.5.2　详细设计

如图3.17所示是计算模块的类图，ImgCalService 为该模块的入口，calBugImg-Similarity() 方法提供了计算两份报告中图片相似度的方法。filterTaskSimilarBugs() 方法是模块的入口，该方法对指定众包任务内的报告进行图片相似度比较，并就图片相似度高的报告进行报告实体相似度计算，在图片相似度高且报告实体相似度不低的情况下，将报告进行合并，合并到同一报告簇。其中，报告实体相似度计算由 KGService 计算得出，KGService 调用知识图谱模块进行计算。

ImgService 提供了图片特征提取和相似度计算等功能，原始报告从 Source-BugService 中进行获取，原始报告中图片格式为 URL 链接，存储在阿里云对象存储 OSS 中，需要从 OSS 下载图片到本地进行处理。ImgDownloader 负责将图片从 OSS 下载到本地。ImgFeatureExtractor 负责对图片的特征进行提取，图片特征使用的是 JCD 特征，图片特征提取使用 LIRe 框架提供的工具包完成。图片特征提取完成后使用余弦相似度来计算图片相似度。

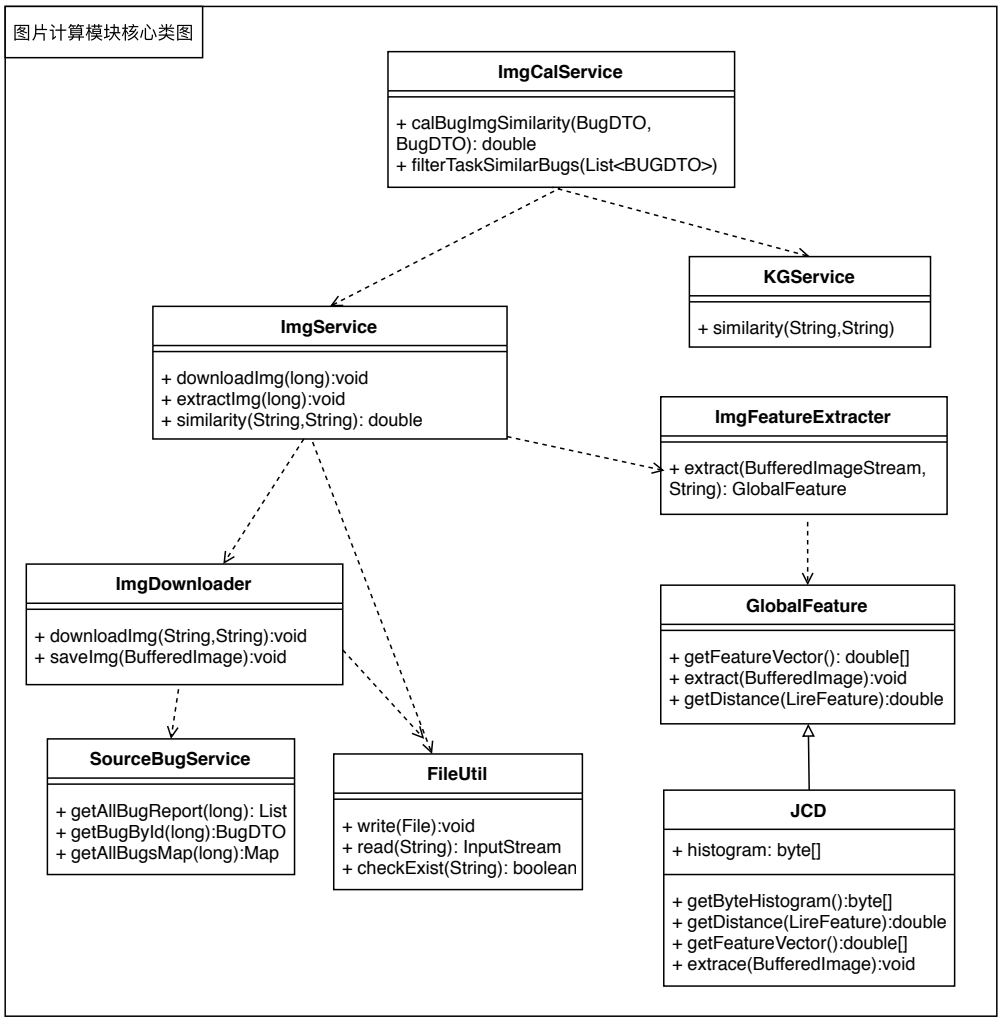

图 3.17: 图片计算模块类图

3.6　知识图谱报告融合模块设计

3.6.1　架构设计

如图3.18所示是知识图谱报告融合模块的模块架构图，模块中所用到的用户提交的原始报告通过外部服务获取。该模块调用知识图谱模块和图片计算模块获取报告合并信息，并依照合并信息对树状报告进行合并，报告合并结果持久化到数据库中。

得到报告合并结果后，首先对每个报告簇执行 PageRank 算法计算出报告簇中的主报告，其中涉及到图的构建，主要是在知识图谱结构中提取报告簇内报告相关的子图结构，报告节点的质量默认相同，报告节点之间的边的权重默认相同，图构建完成后使用 PageRank 算法计算每个节点在报告簇中的权重，并读取

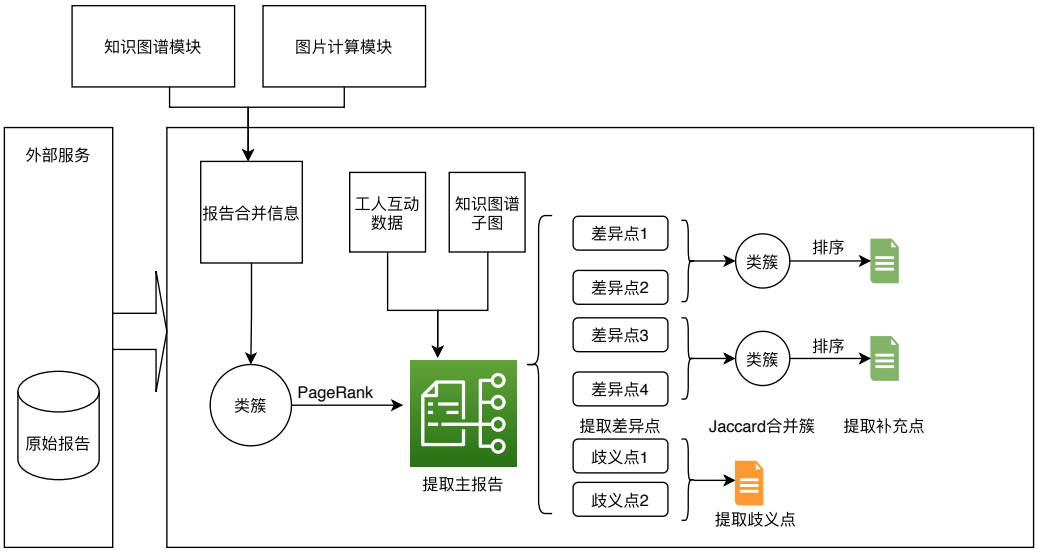

图 3.18: 融合模块架构设计

协作式众包测试中的用户点赞点踩数据，报告每被点赞一次分数提高 1%，每被点踩一次降低 1%。最后选择权重最高的报告作为该报告簇的主报告。之后在将主报告和同一类簇中的其他报告进行比较，提取出报告之间的差异点信息，差异点信息包括句子和图片信息。和报告一样，报告差异点之间也存在着重复情况，需要对报告之间的差异点进行聚类，此处使用的聚类算法为凝聚层次聚类算法。聚类后的报告补充点信息依照补充点所属报告在报告簇中的排序情况进行排序，从而得到报告簇的主报告和补充点报告数据。针对 NLP 过程中所提取的关系构成的知识图谱，查询是否存在歧义点，如果有歧义点信息，生成歧义项。模块运行过程中产生的报告合并信息、主报告、补充点和歧义点信息持久化到数据库中。

3.6.2　详细设计

如图3.19所示是该模块的类图，在该模块中，FusionService 是模块的核心。其他模块调用该模块的 fusion() 方法来对指定的任务进行融合。fusion() 方法将调用 KGService 和 ImgCalService 分别获取报告合并数据，并对报告进行合并，合并后的结果通过 MergeReportService 持久化到系统中。

融合所使用到的原始报告信息通过 SourceBugService 进行获取，系统中会对获取过的报告进行持久化操作，如果该任务中本地报告数量和众测服务提供的报告数量不同，那么系统将对本地的报告进行更新，确保和众测平台原始报告的一致性。

MainReportService 主要负责了对报告簇进行主报告提取的相关逻辑处理。

其中图结构由该类调用 KgService 中的 getClusterChildGraph() 获得。KgService 中提供的图结构是通过查询知识图谱所获得的。

GraphService 提供了融合过程中涉及图构建方面的功能，其中 buildGraph() 能够将输入的报告构造成图结构，从而在后续的主报告提取过程中能够使用 PageRank 算法进行主报告提取，PageRank 类提供了对 PageRank 算法的实现。图构建过程中，图节点数据是依照报告簇中报告在知识图谱中有 2 层关系的子图所构建的。节点及边的默认质量相同。此外由 PageRank 算法计算的报告在报告簇中的得分在通过点赞点踩数据进行加权，规则为如该报告获得一次点赞得分提高 1%，获得一次点踩报告得分降低 1%。

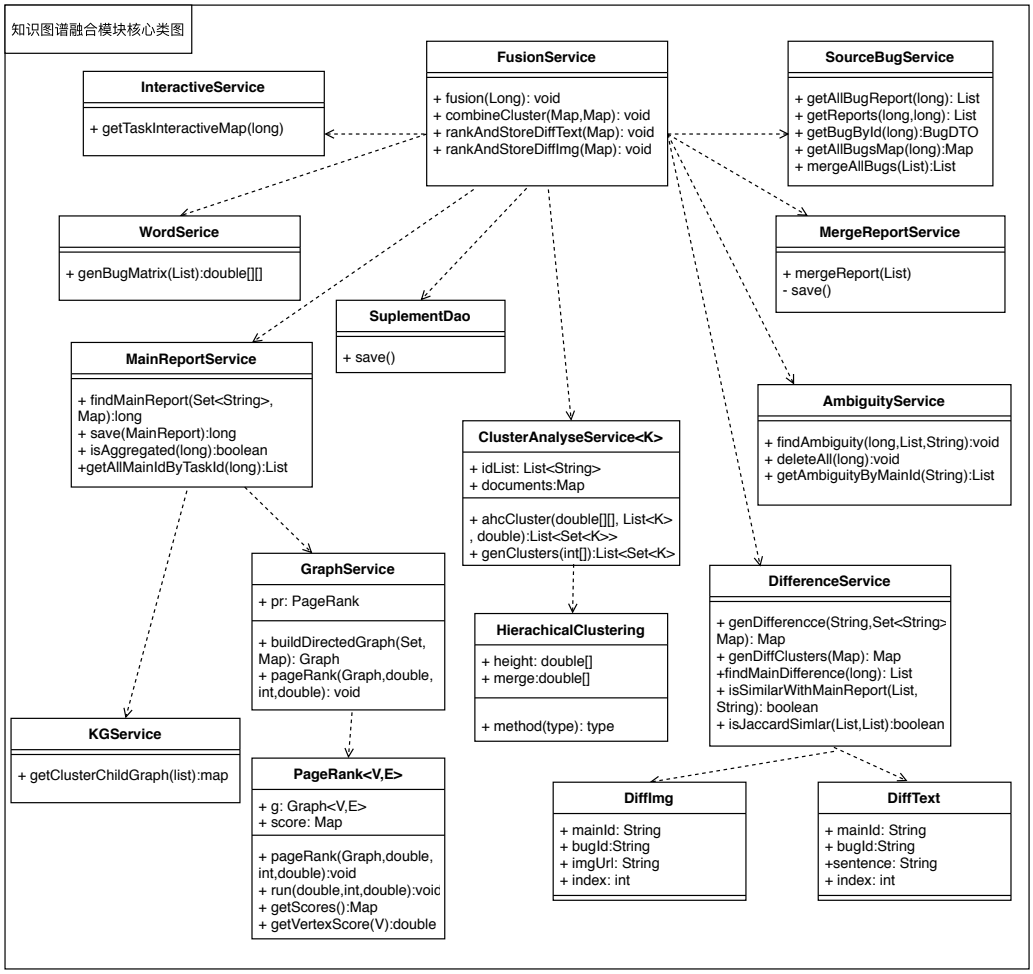

图 3.19: 知识图谱报告融合模块核心类图

ClusterAnalyseService 的功能是对补充点进行聚类分析，本系统使用凝聚层次聚类算法对报告的补充点数据进行聚类，执行的方法为 ahcCluster。

融合过程中产生的补充点信息通过 SupplementService 进行存储，歧义点信

息通过 AmbiguityService 进行提取并存储。DifferenceService 提供了在一个报告簇中识别与主报告不同的文字差异点信息以及不同的图片差异点信息的功能。将文本以句子为单位进行拆解，将图片以单张图片为单位进行拆解，然后将这些拆解出来的信息同报告簇中的主报告进行比较，计算相似度信息，提取出相似度不高的补充点进行聚类，形成补充点报告。AmbiguityService 提取歧义点则是比较报告内容对同一实体是否存在歧义描述项，“显示异常”和“没有显示异常”在知识图谱中对应“show→ 异常”和“unshow→ 异常”，通过比较报告簇所提取的实体和关系中是否存在对于同一实体同时具有互斥关系，如果存在则提取报告的歧义点形成歧义项。

3.6.3　数据库设计

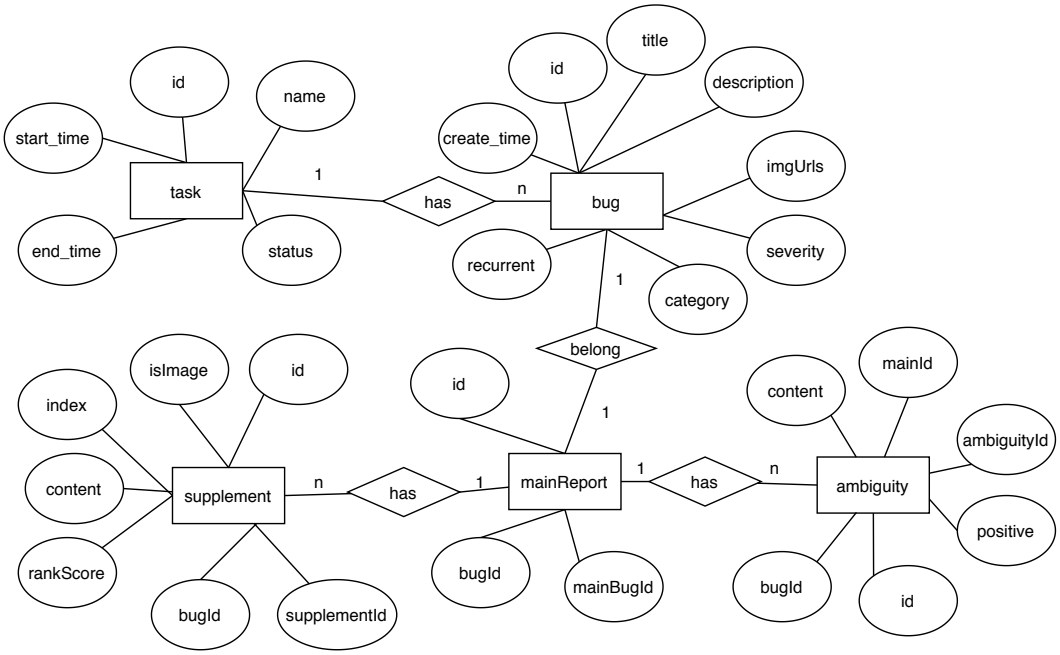

图 3.20: 知识图谱报告融合模块 ER 图

如图3.20所示是知识图谱报告融合模块的 ER 图，描述了和该模块相关的主要实体，task 表为系统中的众包任务，该任务数据是从外部服务处所获得的。bug 表是用户提交的原始报告，同样是从外部服务处所获得的。mainReport 表为报告簇中的主报告，supplement 表为报告簇中的补充点报告。ambiguity 表为报告簇中的歧义点报告，task 表和 bug 表为一对多的关系，一个 task 中有多条 bug 数据。bug 和 mainReport 表为一对一关系，每个报告都有所属的报告簇。mainReport 与 supplement 表为一对多关系，一份主报告可以拥有多条补充点报告，mainReport

与 ambiguity 表为一对多关系，一份主报告可以拥有多个歧义点报告。

如表3.15所示是 task 表字段的详细说明，该表主要记录了系统中的众包任务信息，主要包含 id、name、create_time、end_time 和 status 等字段。表3.15对表中字段进行了详细说明。

表 3.15: task 表

字段	含义	类型	说明
id	众包任务唯一 id	BIGINT	id 为该表的主键
name	该任务的名称	BIGINT	不可为空
status	任务的审核状态	INT	1 表示审核中，0 表示审核结束
create_time	任务开始时间	TIMESTAMP	无
end_time	任务结束时间	TIMESTAMP	无

如表3.16所示是 bug 表字段的详细说明，该表主要记录了系统中的众包任务信息，主要包含 id、name、create_time、end_time 和 status 等字段。表3.16对表中字段进行了详细说明。

表 3.16: bug 表

字段	含义	类型	说明
id	缺陷报告唯一 id	BIGINT	id 为该表的主键
title	缺陷报告的标题	VARCHAR	无
description	缺陷报告的描述	VARCHAR	无
create_time	报告创建时间	TIMESTAMP	无
taskId	报告所属的任务 id	BIGINT	不可为空
imgUrls	报告中用户提交的图片 url 列表	VARCHAR	使用分号进行分割
recurrent	缺陷复现程度	INT	取值范围为从 1 到 5 的整数，分别是其他、无规律复现、小概率复现、大概率复现、必现
category	缺陷所属类别	INT	取值范围为从 1 到 7 的整数，分别是安全、功能不完善、性能、页面布局缺陷、用户体验、不正常退出和其他
severity	缺陷的严重程度	INT	取值范围为从 1 到 5 的整数，分别为待定、较轻、一般、严重和紧急。

如表3.17所示是 ambiguity 表中字段的详细说明，该表主要记录了融合过程中产生的报告簇信息，主要包含 id、mainId 和 taskId 字段。表3.17对表中字段进行了详细说明。

表 3.17: ambiguity 表

字段	含义	类型	说明
id	歧义点唯一标识 id	BIGINT	id 为该表的主键
content	歧义点内容	VARCHAR	无
mainId	歧义点所属报告簇主报告 id	BIGINT	其取值为报告表的报告 id
bugId	该补充点所属 Bug 报告的报告 id	BIGINT	无
positive	是否为正向语义	INT	例如，从“没有提示异常”中所提取的“unshow→ 异常”对应的 positive 为 0，“提示异常”中提取的“show→ 异常”对应的 positive 为 1

如表3.18所示是 supplement 表字段的详细说明，该表主要记录了系统中的报告簇的补充点信息，主要包含 id、isImage、index、content、rankScore、bugId 和 mainId 字段。表3.18对表中字段进行了详细说明。

表 3.18: supplement 表

字段	含义	类型	说明
id	补充点唯一 id	BIGINT	id 为该表的主键
isImage	用于标记该补充点信息是否为图片	INT	1 表示为是图片，0 表示不是图片
index	序号，补充点在报告中的序列号，可以是句子在报告中的序号或者图片在报告图片中的序号	INT	取值为自然数
content	补充点内容	VARCHAR	无
rankScore	该补充点所属报告在报告簇中 PageRank 的得分	FLOAT	无
bugId	该补充点所属 Bug 报告的报告 id	BIGINT	无
mainId	该补充点所属的主报告的报告 id	BIGINT	无

3.7　报告整编模块设计

3.7.1　架构设计

报告整编模块的设计架构图如图3.21所示，该模块的主要承担了系统中报告整编系统业务逻辑的具体实现，报告整编系统是一个由 Java 语言编写的 Java Web 项目。

该模块采用 MVC[41] 的理念进行设计和开发，MVC（Model View Controller）框架又被称之为模型视图控制器模型，MVC 框架的目的是通过控制器 C 将模型 M（代表的是业务数据逻辑）和视图 V（人机交互界面）实现代码分离。该模块的前端部分只负责展示，并不需要进行逻辑处理，前端方面主要使用了 Thymeleaf 这个模版引擎进行开发，展示框架使用的是成熟的 Bootstrap，同后端交互方面使用 Ajax 进行数据传输。Controller 层负责处理来前端的请求，是系统的入口，用户可以使用浏览器通过 Controller 与系统进行交互，Controller 层负责了页面数据的装载，在这一层接受到的请求会被指定逻辑层进行处理。逻辑层对底层的数据进行了封装，是主要进行逻辑处理的地方。存储层主要是面向数据库，提供数据的持久化功能。系统在后端模块中还使用了缓存来提高系统的性能，接入日志组件进行运行日志的收集工作。该模块还需要和部分外部服务进行通讯，比如获取任务信息、报告信息及工人互动数据等。

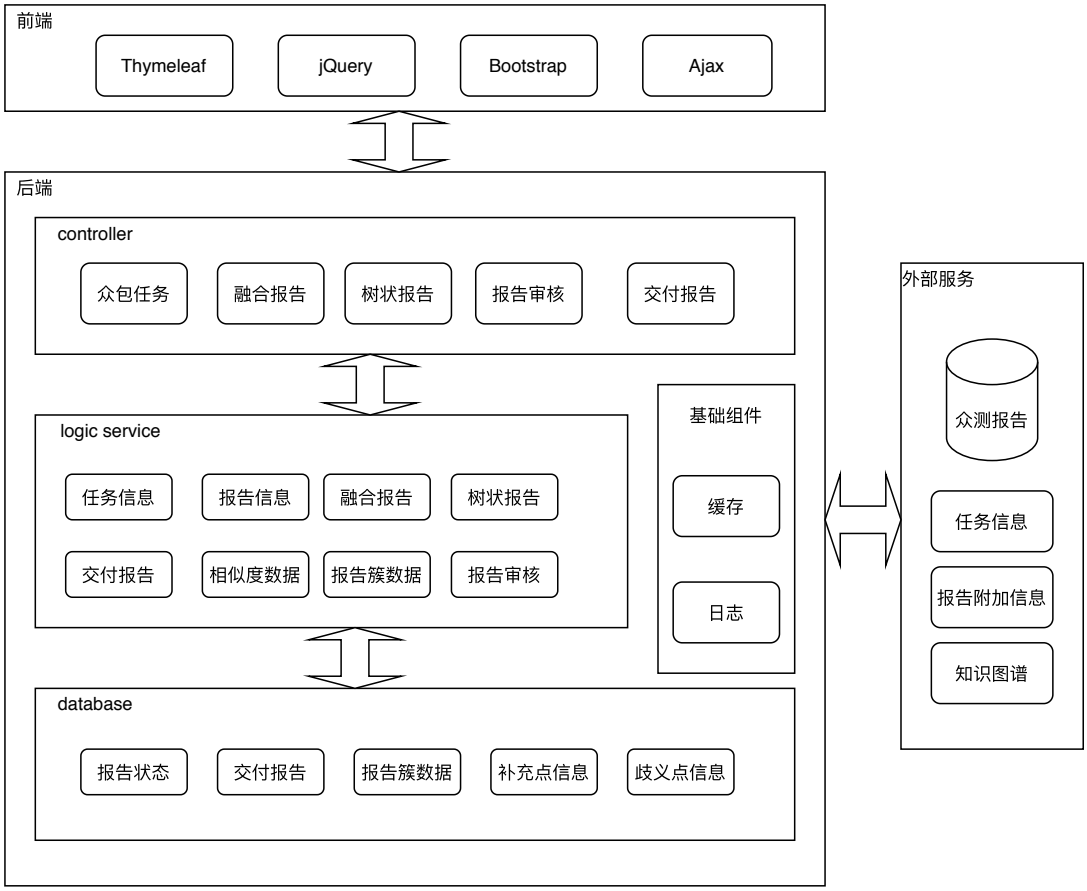

图 3.21: 报告整编模块架构图

3.7.2　流程设计

该模块的流程图如图3.22所示，该流程图描述了整编人员进入到系统进行报告整编的流程:

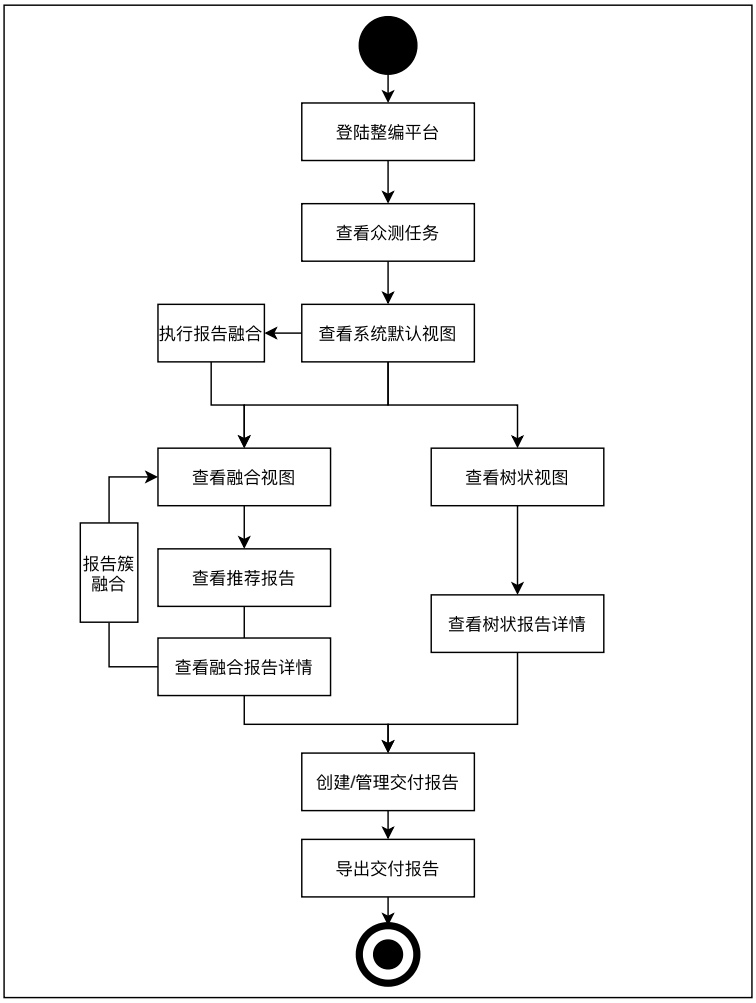

图 3.22: 报告整编模块流程图

（1）整编人员登陆到平台。

（2）查看系统内的众测任务。

（3）选择一个任务进行查看。

（4）系统向整编人员展示默认视图。

（5）整编人员对系统执行报告融合操作，融合完成后可以查看系统的融合报告，该操作在一份任务中只能被执行一次。

（6）整编人员查看该任务的融合/树状视图。

（7）选择一份融合/树状报告进行查看。

（8）对于融合报告，整编人员查看系统推荐的其他相似报告，选择将描述同一缺陷的报告加入到当前报告簇，并点击“融合当前报告簇”。

（9）对于该融合/树状报告进行整编，生成交付报告，设置该报告的审核状态。整编人员可以对已经提交的交付报告进行修改或删除。

重复步骤 6-9 直到完成所有报告的整编工作。

3.7.3　详细设计

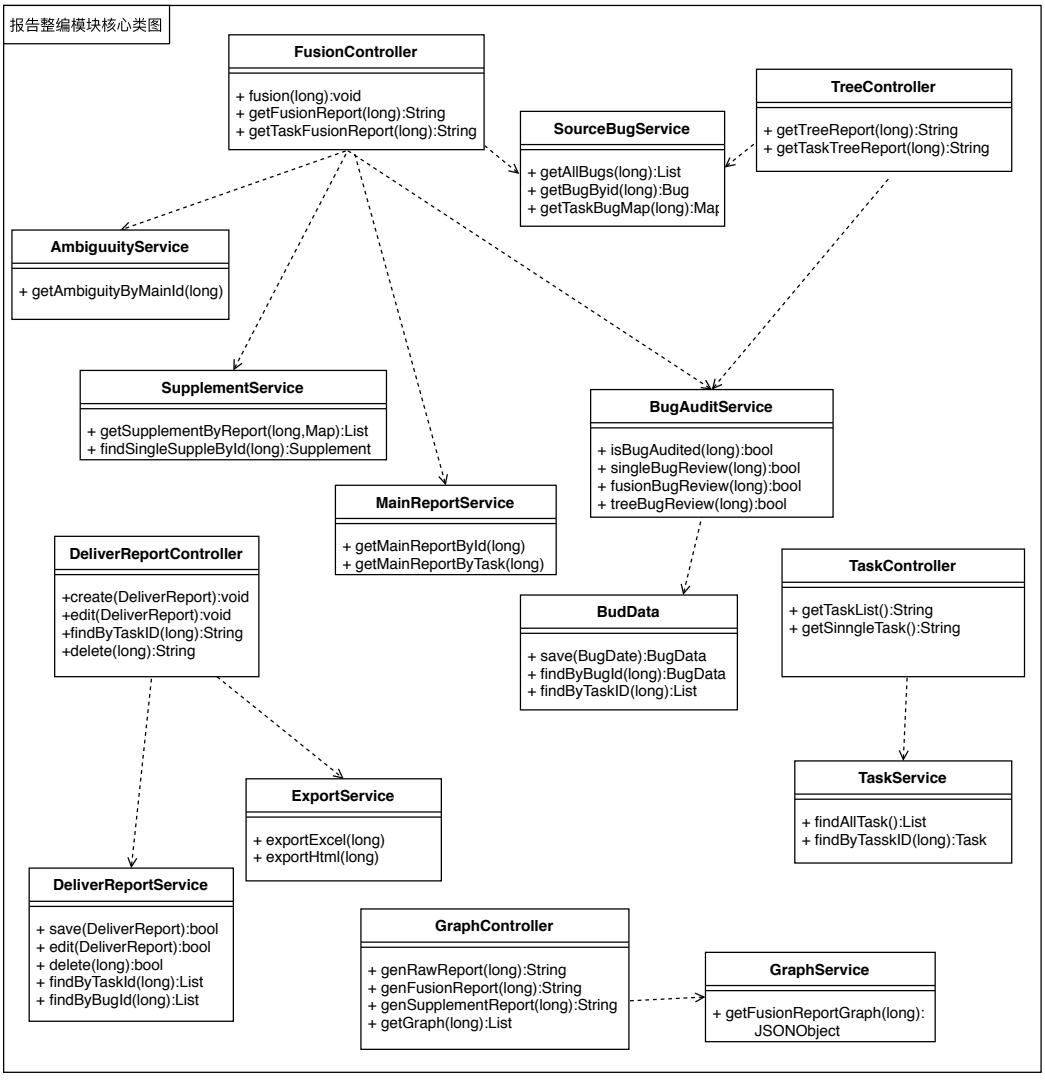

图 3.23: 报告整编模块核心类图

报告整编模块的核心类图如图3.23所示，核心类图主要描述了该模块的具体逻辑实现。

TaskController 主要负责处理和众包任务相关的请求，包括众包任务列表页和众包详情页面，TaskService 和数据库及外部服务进行交互，支持查询系统中所有的众包任务和指定的单个众包任务。

TreeController 主要负责处理树状报告相关的请求，包括树状报告详情页和树状报告列表页，TreeService 和数据库及外部服务进行交互，支持查询指定任务所有树状报告以及指定报告的树状报告。

FusionController 负责处理融合报告相关的请求。包括融合报告列表页、融合报告详情页及触发报告融合操作。触发报告融合后系统会自动化进行融合报告的处理，处理完成异步进程会回调调用函数通知融合完成。SupplementService 对报告融合过程中产生的补充点信息进行了封装和处理并对外提供查询接口。AmbiguityService 对报告融合过程中产生的歧义点信息进行了封装并对外提供查询接口。BugAuditService 是负责报告审核相关的类，提供了对树状报告、融合报告和单一报告的报告审核功能。GraphController 为融合报告结构可视化的展示提供了支持。

DeliverReportController 负责处理交付报告相关的请求。包括对交付报告的增删改查及对交付报告的导出，其中交付报告的导出支持 HTML 网页格式和 Excel 表格格式，导出功能是由 ExportService 所实现的。

3.7.4　数据库设计

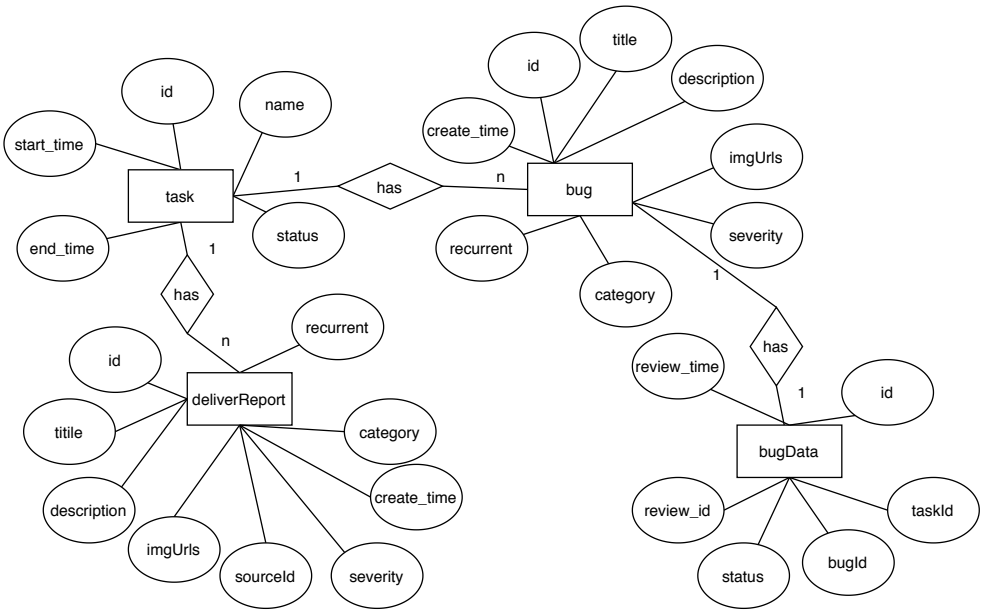

图 3.24: 报告整编模块 ER 图

如图3.24所示是报告整编模块的 ER 图。task 表示系统中的众包任务，deliverReport 表示交付报告，bug 表示用户提交的原始报告，bugData 表用于存储报告的审核状态。task 和 bug 的关系是一对多的关系，task 和 deliverReport 的关系也是一对多的关系，bug 和 bugData 是一对一的关系。

表 3.19: bugData 表

字段	含义	类型	说明
id	报告数据主键 id	BIGINT	id 为该表的主键
taskId	所对应的报告所属的任务 id	BIGINT	不可为空
bugId	所对应的报告的 id	BIGINT	不可为空
status	审核状态	INT	取值为 0 或 1，0 对应未审核，1 对应已审核。
review_id	整编人员 id	BIGINT	无
review_time	报告审核时间	TIMESTAMP	无

如表3.19所示是 bugData 表字段的详细说明，该表主要记录了系统中报告的审核情况，每一份 Bug 报告在 bugData 表中都有一条字段与之对应，主要记录了审核状态、审核人和审核时间。该表主要包含 id、taskId、bugId、status、review_id 及 review_time 共计 6 个字段。

表 3.20: deliverReport 表

字段	含义	类型	说明
id	交付报告唯一 id	BIGINT	id 为该表的主键
taskId	该报告所属的任务 id	BIGINT	不可为空
title	报告标题，该 Bug 报告的标题	VARCHAR	不可为空
description	报告描述，报告中对出现 Bug 的描述	VARCHAR	不可为空
imgUrls	报告图片，报告中提交的对 Bug 的截图	VARCHAR	可为空，存储的是图片的 URL 信息，一份报告可以提交多份图片，多份图片之间使用分号进行分隔
sourceId	原始报告 id，记录创建该交付报告的页面是原始报告 id	BIGINT	无
severity	Bug 严重等级	INT	主要分为 1-5 共 5 个等级，不可为空
create_time	报告创建时间	TIMESTAMP	无
category	Bug 所属的类型	INT	主要分为 1-7 共 7 个类型，不可为空
recurrent	Bug 复现程度	INT	主要分为 1-5 共 5 个等级，不可为空

如表3.20所示是 deliverReport 表字段的详细说明，该表主要记录了系统中整编人员生成的交付报告的信息，主要包含 id、taskId、title、description、imgUrls、sourceId、severity、create_time、category 和 recurrent 共 10 个字段。

3.8　本章小结

本章首先对基于知识图谱的众测报告融合系统的功能需求、非功能需求进行了分析和说明。依据分析对系统进行了相应的技术选型并就系统的总体架构进行了分析，包括了对系统 4+1 视图和模块的划分进行了说明。最后系统被分解为知识图谱模块、图片计算模块、知识图谱报告融合模块、报告整编模块共计 4 个模块。每个模块就模块的总体设计、流程设计、类图体系结构设计和数据库设计进行了详细的描述。

第四章　基于知识图谱的众测报告融合系统的实现

第三章主要描述了系统的整体设计和模块的架构设计，本章节将就系统的具体实现进行阐述，主要通过顺序图和关键代码解析的方式对模块进行分析，对于有页面展示的模块，展示系统运行截图。

4.1　知识图谱模块的实现

该模块涉及了对报告描述进行关系提取、知识图谱构建及知识图谱翻译模型向量表示三个主要部分。其中关系提取部分主要使用了 NLP 技术对报告文本进行分析，系统提供了两种分词模型：Jieba 和 Pyltp 分词，Pyltp 是一款常用的中文 NLP 工具包，其提供了许多中文自然语言处理实现，包含了分词、依存句法分析和词性标注等。关系提取模块中主要使用 Pyltp 对报告进行文本分析。模块使用 Neo4j 存储任务中的图结构，主要实体包括：任务、测试用例、报告、用户及报告中的操作实体。主要关系如图4.1所示，此外，针对 NLP 提取关系组过程中从报告文本描述内容中提取的实体，本文还引入了第三方分类知识图谱进行补充。知识图谱翻译模块中使用的模型为 TransE 翻译模型，对知识图谱中的实体生成实体向量。

表 4.1: 知识图谱关系列表

关系名称	含义
good	点赞，指的是用户对报告的点赞
bad	点踩，指的是用户对报告的点踩
include	包含，指的是测试用例包含缺陷报告
inpage	所处三级页面，指的是缺陷报告所属的三级页面，三级页面指的是填写报告时需要选择的 Bug 出现页面
click	点击，指的是用户对系统的操作，如：点击按钮
show	提示，指的是系统反应，如：系统提示用户未授权
switch	切换，指的是系统页面变化，如：系统跳转到登陆页面
input	输入，指的是用户进行的输入操作，如：在搜索框中输入“苹果”
choose	选择，指的是用户进行的选择操作，如：选择使用当前位置
search	搜索，指的是用户进行的搜索操作，如：搜索汉堡
任务通用关系	指的是在某场众测任务中频繁出现的谓语词汇，但是不在上述通用动作列表中，如测试购物 App 时的“加入”关键字
un*	语义相反关系，unclick、unshow、unswitch 等。如：没有跳转到登陆页面

4.1.1　知识图谱模块流程

如图4.1所示是知识图谱模块的顺序图，KGService 是该模块的入口，负责该模块内各类之间的协作，同时该类对外提供知识图谱模块的接口，其他模块可以通过 Thrift 对该类进行方法调用。SourceBugService 主要用于获取外部服务提供的原始报告数据。NLPService 负责对原始报告数据进行语言处理，主要目的是从报告描述中提取用户操作及系统表现，其运用了分词、词性分析和依存分析等功能。Crowd2Neo4j 主要负责对系统内存在的结构性数据及 NLPService 在报告文本描述中提取的关系进行知识图谱构建，并引入第三方分类知识图谱进行补充，构建的实体和关系数据保存在 Neo4j 图数据库中。Neo4j2CSV 主要负责将 Neo4j 中存储的关系及实体数据导出成知识图谱翻译模型所需要的 CSV 结构。Train 负责了模型的训练及实体及关系向量的获取。TransE 类是 TransE 知识图谱翻译算法的具体实现。

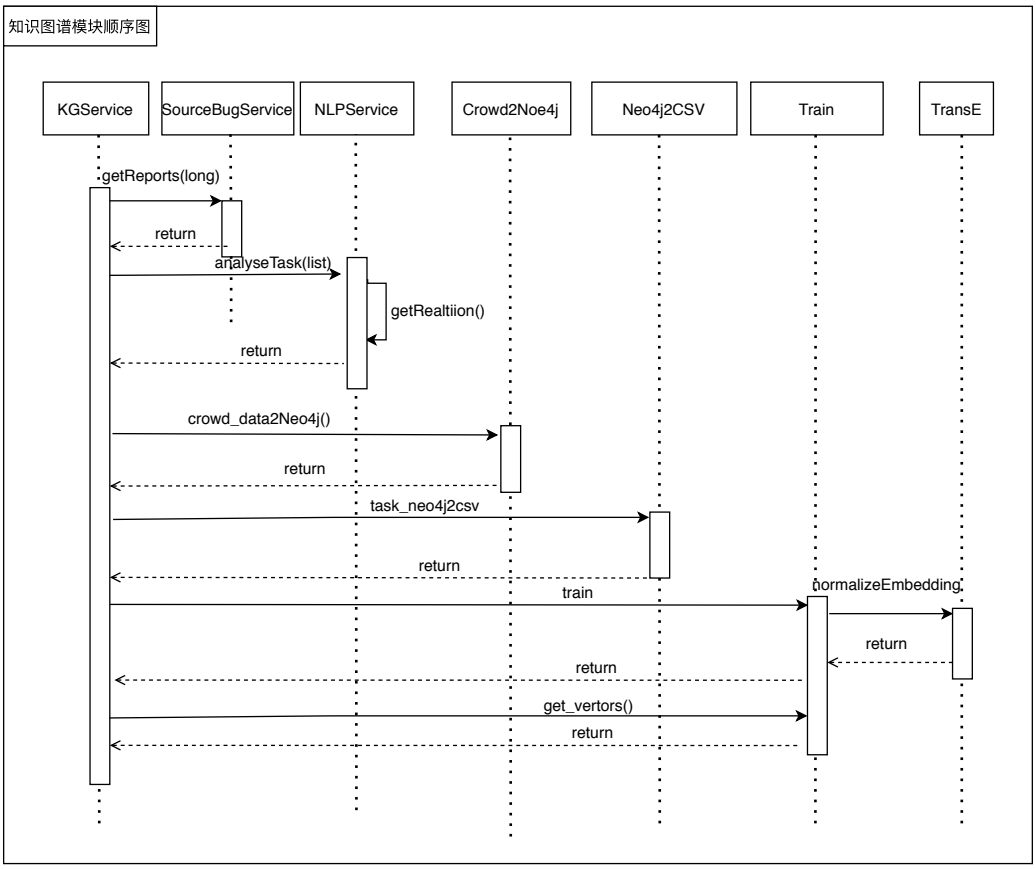

图 4.1: 知识图谱模块顺序图

4.1.2　NLPService 类详细实现

NLPService 的关键代码如下图所示。filterFreqqentPV 方法主要是对报告内的谓语动词进行统计，除了“点击”、“搜索”、“跳出”、“提示”、“进入”等常见的用户或系统操作，其他出现频率高的谓语词会被任务是该任务专有的谓语动词，如测试购物商城是“购买”可能是一个常见动作，如测试资源类应用时，“上传”可能是一个常见动作，目前统计的任务频繁谓语动词的规则是：出现频率超过 15 次或者占报告比例超过 10%。getRelation 方法主要是在单句中提取关系。

```
def analyseTask(self,bugs):
    self.filterFreqqentPV(len(bugs),bugs)# 筛选除”点击”,”跳转”等以外的高频谓词
    for bug in bugs:
        sentences = self.splitSentenceByRe(bug[' description '])
        for single in sentences:
            words = self.segment(single)# 进行分词
            yicun = self.yicun(words)# 进行依存分析，需调用词性分析
            relation = self.getRelation(words,yicun)# 关系提取，
            self.saveRelation(bug, relation)# 持久化
def filterFreqqentPV(self,reportNum):
    self.freqquentPV=sorted(self.taskPV.items(), key=lambda kv:(kv[1],kv[0]))
    for item in self.freqquentPV:
        if (int(item[1])>15 or int(item[1])*1.0/reportNum > 0.1):
            self.frequentWei.append(item[0])
            #将出现>15次或>10%的谓词加入到任务频繁谓词中
        ...
def getRelation(self,words,yicun):
    sentence = self.analyse(words,yicun)# 对句子进行分析，并提取反语义识别
    elif '输入' in sentence.wei:
    #针对”输入”、”点击”、”跳转”、”搜索”等常见动作进行关系提取
        for single_bin in sentence.bin:
            relations.append([sentence.reverse + 'input', single_bin])
    ...
    else:
        sameWord = [i for i in sentence.wei if i in self.frequentWei]
        if len(sameWord)!=0:# 对任务其他常见谓词进行提取
            …关系提取，同上
    return relations
```

图 4.2: NLPService 关键代码

4.1.3　Crowd2Neo4j 类详细实现

本系统中所提取的实体和关系保存在 Neo4j 图数据库中，其中关系实体主要包括：报告、测试用例、用户、三级页面、报告内容中抽取的实体和引入第三方知识图谱补充的实体。系统内结构化数据所提取的关系及报告文本内容中所提取的关系如表4.1所示，Crowd2Neo4j 类主要负责对系统内的实体和关系进行提取，并将其持久化到 Neo4j 数据库中。图4.3所示是该类执行过程。图4.4所示是系统所构建的知识图谱示意图。

```
for 任务中的测试报告 do:
    for 测试报告中的缺陷报告 do:
        存储众测系统内的结构化关系，包括树状报告parent关系，缺陷发生页面inpage关
        系，测试用例包含缺陷报告include关系等。
        for 报告描述中提取的关系 do:
            保存关系到知识图谱
            if 分类知识图谱中包含提取的实体 and 实体未导入当前知识图谱:
                提取分类知识图谱中实体向上追溯的实体，导入当前知识图谱
        end
    end
end
```

图 4.3: Crowd2Neo4j 执行代码

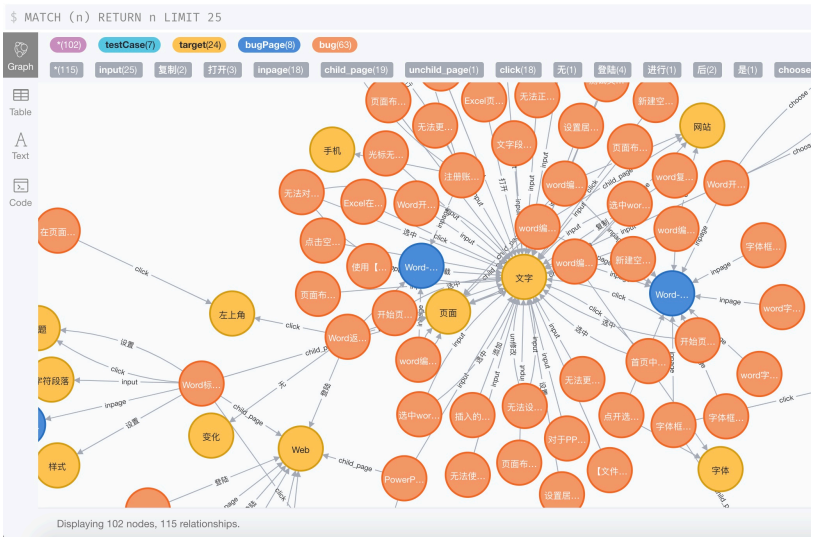

图 4.4: 系统构建知识图谱示意图

4.1.4　TransE 详细实现

TransE 模块主要对 TransE 算法进行了实现。其目的是将知识图谱中的实体和关系翻译成具体的向量。TransE 模块首先对系统内的实体和关系初始化向量，初始化时向量随机取值，然后对向量进行归一化。然后每次随机选取系统内 150 个向量进行更新，同时获取向量原始值和负三元组即错误的三元组信息。更新时通过梯度下降法求解损失函数的最小值，从而得到最优向量解。通过不断的迭代，最终实现所有的向量得到更新。模型翻译得到的向量保存在本地。该模块实现如图4.5所示。

```
def transE(self, times = 1000):
    for index in range(times):#迭代times次
        Sbatch = self.getSample(150) #随机获取150个三元组
        Tbatch = []#元组对（原三元组，负三元组）列表
        for sbatch in Sbatch:#遍历元组，并获取它们的负三元组
            tripletWithCorruptedTriplet = (sbatch, self.getCorruptedTriplet(sbatch))
            #将sbatch传入，获取负三元组，构成一个元组对
            if(tripletWithCorruptedTriplet not in Tbatch):
                Tbatch.append(tripletWithCorruptedTriplet)
        self.update(Tbatch) # 对整个集合进行更新
        if cycleIndex % 100 == 0:
            print("第%d次循环"%index)
            print(self.loss)
            self.saveRelationVector(self.taskid+"relationVector.txt")
            self.saveEntilyVector(self.taskid+"entityVector.txt")
            self.loss = 0
```

图 4.5: TransE 实现关键代码

4.2　图片计算模块的实现

4.2.1　图片计算模块流程

图片计算模块的顺序图如图4.6所示，ImgCalService 类是该模块的入口，该类调用 ImageService 获取报告内图片的相似度情况。因为用户提交的原始报告中图片的保存形式是 URL 链接，因此需要把报告图片下载到本地，用户提交的原始报告通过 SourceBugService 获取，ImgDownloader 负责图片的下载，ImageService 调用文件工具类 FileUtil 将图片保存到本地。

ImgFeatuerExtractor 类负责对图片特征进行提取，其使用 Lire 框架的算子提取了图片的 JCD 特征，该特征综合考虑了图片的颜色、对比度和图片内色块的边界。针对图片特征提取后得到图片的向量表示，对图片特征向量与其他报告计算余弦相似度作为图片之间的相似度。

对于图片相似度高的报告，ImgCalService 调用 KGService 计算其报告实体的相似度信息，并比较相似报告所处的三级页面数据，如果报告实体相似度也较高的情况下，将两个报告合并到同一报告簇中。

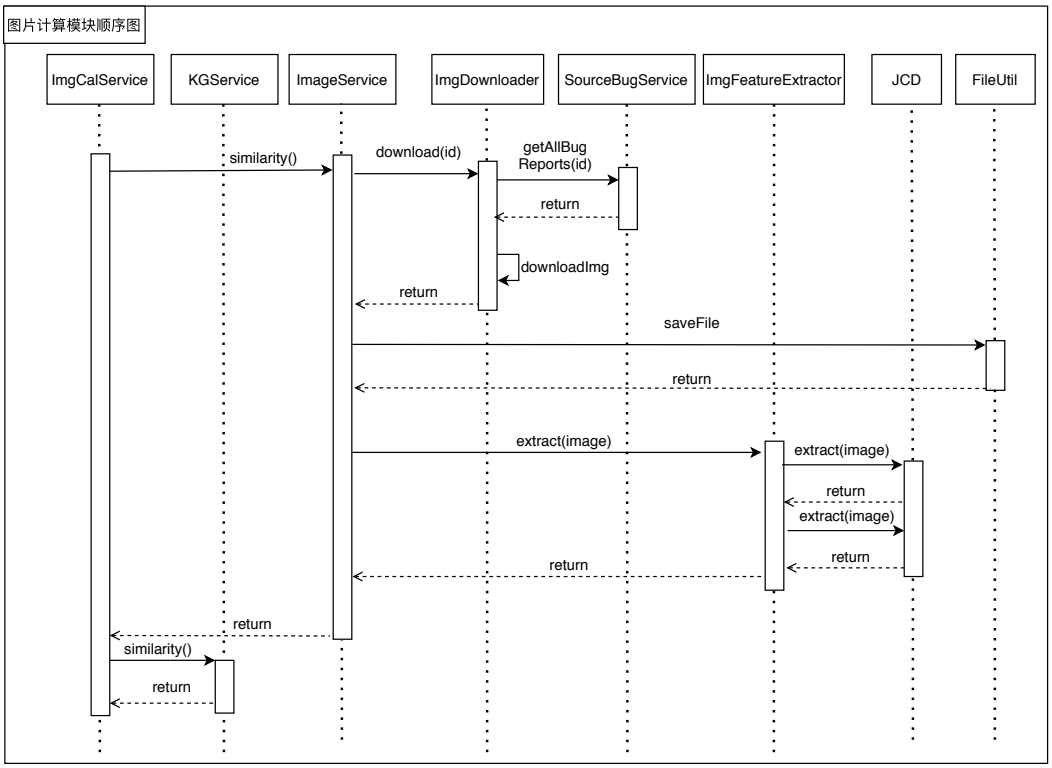

图 4.6: 图片计算模块顺序图

4.2.2　关键代码

如图4.7所示是图片计算模块的关键代码。该模块首先使用 JCD 算子对图片特征进行提取，得到单个图片的特征向量。然后对于报告之间图片进行比较，报告图片比较使用余弦相似度计算图片相似度，对于图片相似度高于 0.8 的图片比较其所属报告在知识图谱中的实体相似度，如果实体相似度高于 0.8，则对图片所属报告进行合并。

```
public List<String[]> filterTaskSimilarBugs (List<BugDTO> bugs){
    Map<String,List<float[]>> bug2imgVec;//bug对应图片特征向量
    Map<String,BugDTO> bug2DTO ;//bug对应报告实体
    for (BugDTO bug : bugs){
        List<float[]> from = bug2imgVec.get(bug.getId ());
        for (Map.Entry<String, List<float[]>> entry : bug2imgVec.entrySet()){
            if (! entry .getKey(). equals(bug. getId ())){
                List<float[]> to = entry . getValue ();
                if ( calImgsSimilarity (from,to )>0.8){// 比较图片相似度
                    BugDTO compare = bug2DTO.get(entry.getKey());
                    if(kgService. similarity (bug. getId (), compare.getId())>0.8&&
                    bug.getBug_page(). equals( compare.getBug_page())){
                    // 比较实体相似度及三级页面
    后续‘}’及保存代码省略
}
```

图 4.7: 图片计算模块关键代码

4.3 知识图谱报告融合模块的实现

4.3.1 知识图谱报告融合模块流程

知识图谱报告融合模块的顺序图如图4.8所示。FusionService 是该模块的核心类，是该模块的调用入口。

FusionService 调用 SourceBugService 首先获取系统中众测任务的原始报告。之后调用 ImgCalService 和 KGService 获取树状报告合并信息。并由 MergeReportService 类完成对报告簇的合并。对于已经合并的报告簇，系统调用 MainReportService 进行主报告提取，MainReportService 调用 InteractiveService 获取工人点赞点踩数据，调用 KGService 获取报告簇子图结构，图的构建由 GraphService 完成，之后调用 pageRank() 方法计算报告簇中各报告的得分，取最高分报告作为报告簇的主报告。调用 AmbiguityService 提取报告簇内的歧义点信息，歧义点信息的提取需要调用知识图谱服务完成。对于已经提取主报告的报告簇，调用 DifferentService 获取报告的差异点信息，差异点信息可以是报告中的单一句子和单一图片。对于报告簇内的差异点，使用聚类算法进行聚合，其中凝聚层次聚类算法的具体实现是由 HierachicalClustering 类所实现的。对于报告差异点合并的报告簇，使用 FusionService 中的 mergeCluster() 对补充点报告簇进行合并。合并之后的报告簇调用 MainReportService 获取补充点报告所属主报告在报告簇内

的排序，从而确定差异点报告簇的顺序，作为报告簇中的补充点信息。

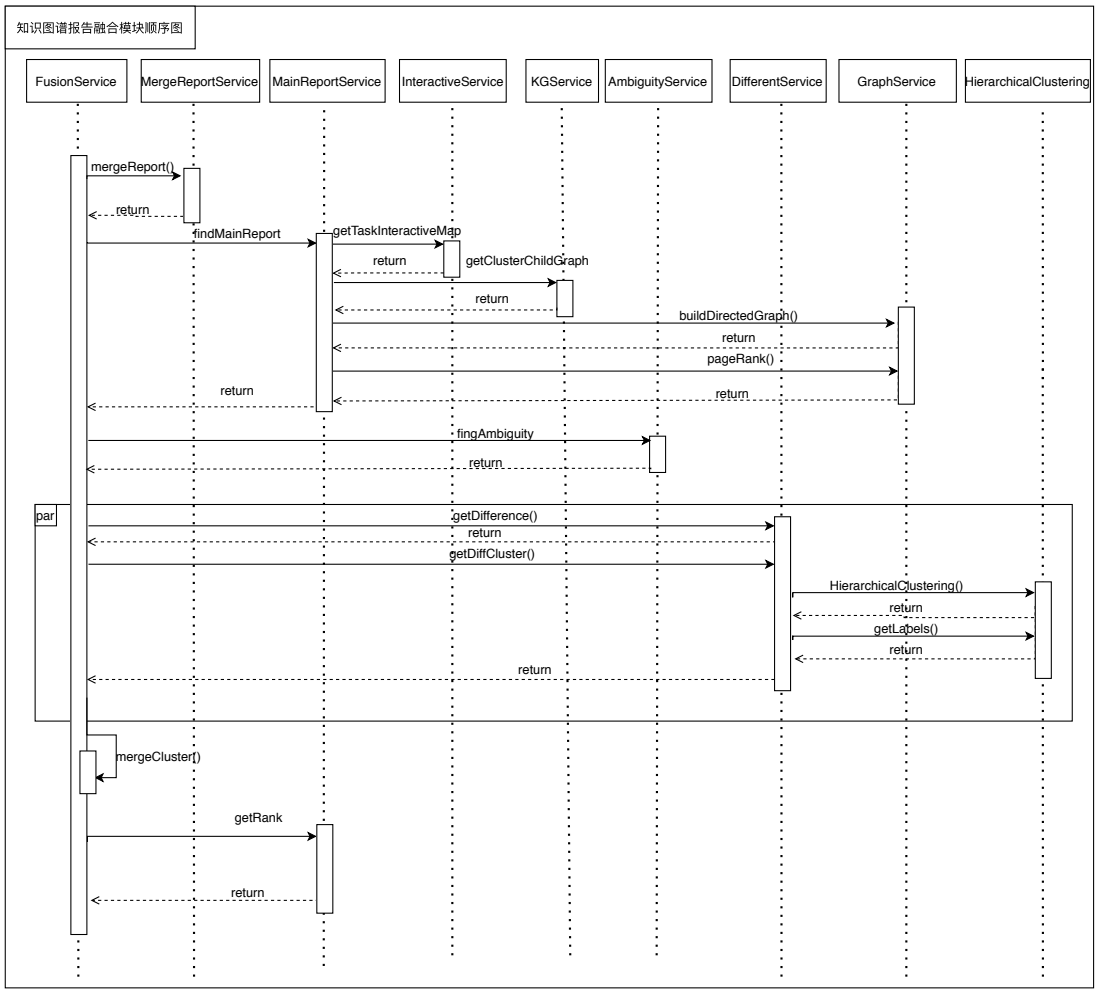

图 4.8: 知识图谱报告融合模块顺序图

4.3.2　关键代码

报告融合模块的主要方法 fusion() 如图4.9所示。该方法首先获得了所有原始报告的信息，然后分别调用 KGSerivice 及 ImgCalService 中的 filterTaskSimilarBugs 方法获取报告合并信息，并交由 MergeReportService 进行报告合并。合并后的报告由 MainReportService 进行主报告提取，主报告提取过程中涉及到图的构建，图结构调用 KGService 从知识图谱中获取，之后使用 PageRank 算法进行主报告提取。对于合并的报告簇调用 AmbiguityService 获取报告的歧义点，主报告提取完成后分别进行文字差异点和图片差异点提取，并就报告簇中的文字及图片差异点进行聚类，聚类之后对于形成的报告补充点类簇进行合并。

```
public void fusion(long taskId) {
    List<Bug> bugs = sourceBugService.getAllBugs( taskId );// 获取所有bug
    //分别获取知识图谱模块和图片计算模块的累簇合并信息
    List<String[]> simBugByImg = imgCalService.filterTaskSimilarBugs(bugs)
    List<String[]> simBugByKg = kgService.filterTaskSimilarBugs(bugs)
    ClusterAnalyzer<String> clusterCal = new ClusterAnalyzer<>();
    List<String> bugIds = bugs.stream().map(bug->bug.getId()).collect(
        Collectors.toList());// 获取报告的id列表
    List<Set<String>> clusters = mergeService.mergeReport(simBugByImg,simBugByKG);
    Map<String, Bug> bugInfo = bugs.stream(). collect(toMap(Bug::getId,
    Function.identity()));// 构建map
    Map<String, Set<String>> mainInfo = new HashMap<>();//构建map
    for (Set<String> cluster : clusters) {
        //持久化部分代码省略
        //提取主报告
        String mainReport = mainReportService.findMainReport( cluster , bugInfo);
        mainInfo.put(mainReport, cluster );
        ambiguityService.findAmbiguity( cluster ,mainReport);
        //获取报告的歧义点信息，并存储
    }
    //获取报告簇片差异点
    Map<String, List<DiffItem>> diffMap = diffService.getDifference(mainInfo, bugMap);
    Map<String, List<Group<String, DiffItem>>> diffMap = diffService.
     getDiffClusters(diffMap );// 对差异点进行聚类
    mergeCluster(diffMap );// 对类簇进行合并
    //存储
}
```

图 4.9: 知识图谱报告融合模块关键代码

4.4 报告整编模块的实现

4.4.1 任务列表与详情页实现

如图4.10是系统的任务列表页面。主要展示了任务的名称、任务当前的审核状态及进度。考虑到系统中的任务数量众多，对任务列表进行了分页处理，同时该页面还提供了对任务的检索功能，可以输入关键字来查询系统中的任务。任务列表默认按照任务开始时间进行排序。

图 4.10: 任务列表页

如图4.11所示是系统的任务详情页，主要展示了该任务的审核情况以及任务中的报告信息，当任务尚未进行报告融合操作时，该页面会显示“报告融合”按钮，可以触发报告融合操作，对该任务进行处理。当报告融合按钮触发后，该页面会进入等待阶段，融合过程结束后，页面会进行刷新，显示各报告所属的融合报告。考虑到每个任务中的报告数量众多，系统对该页面进行了分页处理，且支持关键字检索。报告列表中的每一列还支持排序操作。在该页面点击报告所属的融合报告会进入到融合报告详情页，点击报告所属的树状报告会进入到树状报告详情页面。

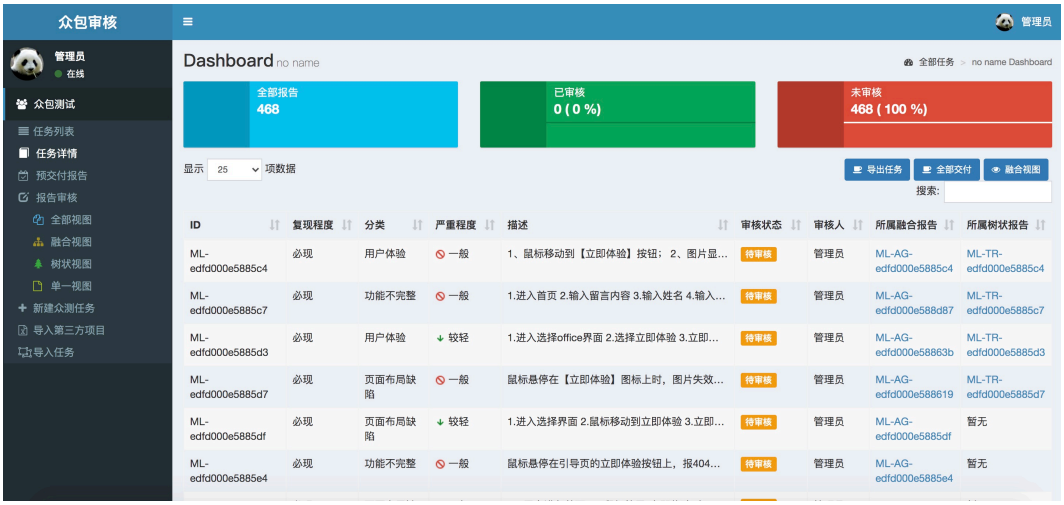

图 4.11: 任务详情页

4.4.2　融合报告的实现

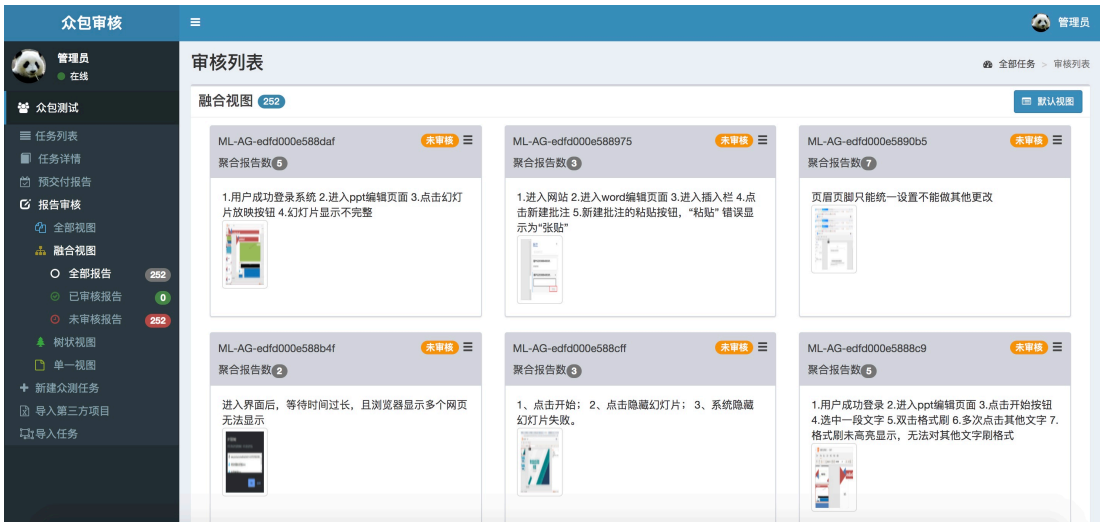

图 4.12: 融合报告列表页

如图4.12所示是系统的融合报告列表页，在任务详情页点击融合视图可以进入到该页面。众包任务经过报告融合之后会生成许多报告簇，该页面中的每一项就代表一个报告簇，点击每一项中的报告 id 可以进入到融合报告详情页。卡片右上角的图标表明了该报告簇的审核状态，报告簇在该页面显示的主要是报告簇中主报告的信息，点击右上角的展开按钮可以查看报告簇的补充点信息和歧义点信息，点击左侧导航栏可以对不同审核状态的融合报告进行筛选。

图 4.13: 融合详情页

如图4.13所示是系统的融合报告详情页面，可以由任务详情页及系统的融合视图页面进入。融合详情页主要展示了报告簇中的具体内容，例如三级页面、复现程度等统计信息，此外还包括报告簇的主报告及其他报告生成的补充点信息。补充点报告中的重要语句会通过高亮的方式显示，如果报告簇中报告间存在描述歧义项，将会有歧义点，歧义点报告中的重要语句也将通过高亮的方式展示，方便整编人员查看。

如图4.14展示的是报告簇中所提取的歧义点信息，报告 2 为对报告 1 的补充，报告 2 中未出现报告 1 中按钮图片不显示的缺陷，但是报告 2 中存在背景图片不显示的缺陷。

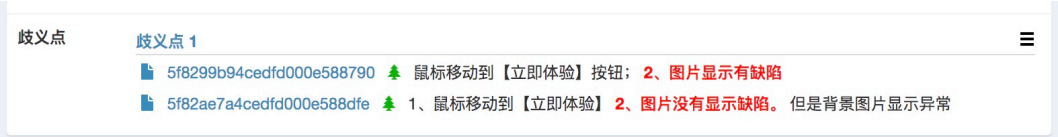

图 4.14: 歧义点展示

如图4.15是融合详情页面显示的知识图谱关系图，知识图谱关系图将展示该报告簇在众测知识图谱中的相关节点及结构，点击节点可以查看节点内容。

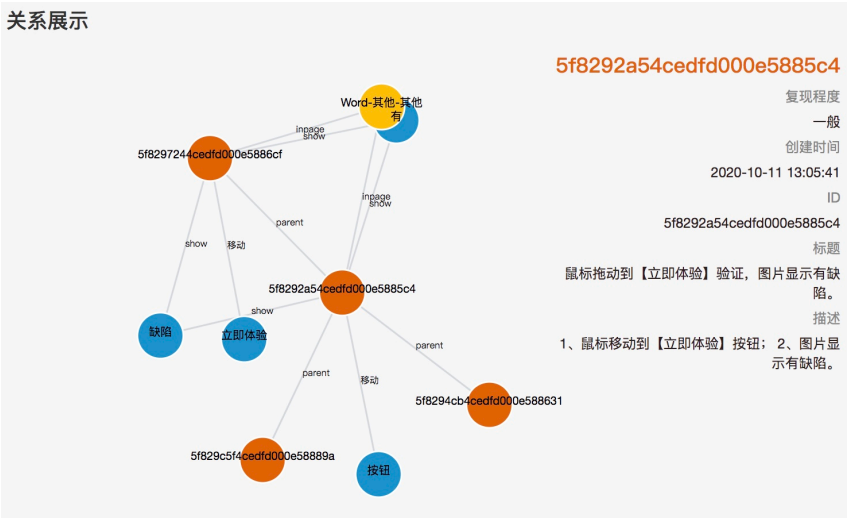

图 4.15: 关系展示

如图4.16所示是详情页报告再融合相关功能的实现，整编人员可以在该页面查看与当前报告簇相似的报告，推荐的报告主要是知识图谱中报告实体相似度高的报告及文本、图片相似度高的报告，系统将会推荐这些相似报告并给出推荐理由，整编人员可以选择将推荐报告加入到当前报告簇，并对加入报告后的报告簇再次进行报告融合。

图 4.16: 详情页报告推荐与再融合展示

4.4.3　树状报告的实现

如图4.17所示是系统的树状报告详情页面，可以由任务详情页和系统的树状视图页面进入，树状报告详情页主要展示了用户在填写报告中生成的树状报告信息，其中包括复现程度、严重等级等统计信息。该页面还原了树状报告的原始树状结构，方便整编人员查看。

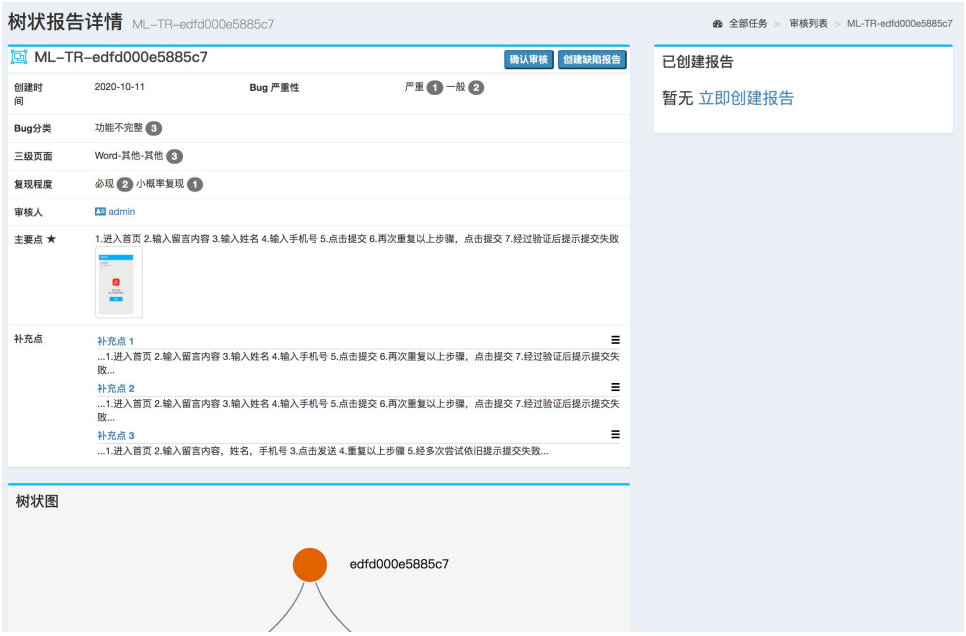

图 4.17: 树状报告详情页

4.4.4　报告整编的实现

如图4.18所示是编辑报告页面，用户可以在单一报告、树状报告和融合报告页面进行报告整编。整编人员通过在报告页面查看系统提取的主要观点、差异点和歧义点。整理出新的报告进行交付，其中 Bug 复现程度、严重性和分类是通过下拉框的方式进行选取，Bug 描述通过键盘进行输入，Bug 截图通过拖拽其他报告中的图片进行上传，点击保存可以把编辑好的报告持久化到系统中。

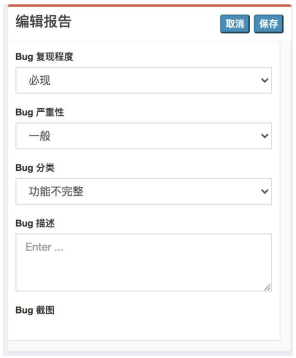

图 4.18: 报告整编页面

4.4.5　交付报告的实现

如图4.19所示是交付报告管理页面，展示的内容是整编人员在本次任务中编辑提交的交付报告。主要展示了交付报告的类别、复现程度、严重等级、具体描述和来源信息。该页面可以对交付报告进行编辑和删除操作。交付报告支持两种格式的报告导出，一种是 HTML 网页格式，另一种是 Excel 表格格式。

图 4.19: 交付报告管理

4.5　本章小结

本章主要对基于知识图谱的众测报告融合系统进行了实现细节的详细描述，首先介绍了系统知识图谱部分的流程和对应代码实现，其次对图片计算模块就顺序图和对应代码实现进行了分析，然后在知识图谱报告融合模块就整体流程进行了阐述，最后对报告整编模块进行了系统运行截图的展示。

第五章　基于知识图谱的众测报告融合系统的测试

根据第三章对系统功能需求和非功能需求的分析，本系统首先要保证功能的完整可用，其次是要保证非功能指标达到要求。本章节首先对该系统的功能测试进行测试，之后对非功能测试中的可移植性、可用性及性能进行测试，并进行了效果测试以验证效果。每次测试都先描述测试设计，之后展示测试结果。

5.1　测试准备

5.1.1　测试目标

本文对基于知识图谱的众测报告融合系统主要进行三方面的测试，首先是对系统进行功能测试，功能测试主要是针对第三章中描述的系统测试用例，针对每个用例来验证用例是否达到要求，然后对系统非功能需求进行测试，主要是可移植性测试、可用性测试和性能测试。可用性测试主要是测试系统所要求的宕机不超过 2 分钟的要求。可移植性测试主要是为了响应信创国产化号召，测试该系统在不同操作系统、不同系统架构上能否正常运行。性能测试则是为了测试系统在高并发场景下的性能表现。最后对系统进行效果测试以验证效果。

5.1.2　测试环境

表 5.1: 系统测试服务器

系统模块	服务器型号	服务器系统
报告融合服务	Aliyun ecs.c1.large	Ubuntu
Redis 服务	Aliyun ecs.c1.large	Ubuntu
数据库	Aliyun ecs.c1.large	Ubuntu

如表5.1所示是用于系统测试的服务器列表，其中报告融合系统、数据库服务与用于做系统缓存的 Redis 服务均部署在不同的服务器上，用于测试系统可移植性的服务器列表将在可移植性测试章节详细说明。

如图5.1所示为在服务器上配置的服务监测代码配置，其主要功能为监测系统 Docker 进程的运行情况，如果 Docker 进程因为系统出现异常造成宕机，则该脚本负责系统的重启。

```
#! /bin/bash
while true
do
    crowd_review=`docker ps |grep "crowd_review"|grep -v grep|wc -l`

    if [ $crowd_review -eq 0 ]
    then
        echo "start crowd_review"
        sh build_run.sh
    else
        #do nothing
    fi
    sleep 45 #每 45 秒检查一轮
done
```

图 5.1: 服务监测代码配置

5.2　功能测试

本章节根据第三章的功能需求进行分析，对系统用例进行功能测试设计，主要做法是按照用例描述的步骤对指定用例进行操作执行，以验证系统是否符合预期，是否出现功能上的异常，从而判断系统功能是否完善。

表5.2描述了查看任务列表的测试用例，主要对任务列表页面进行测试，以测试系统功能是否完善。

表 5.2: 查看任务列表测试用例

测试 ID	TC1
测试名称	查看任务列表
待测功能	整编人员查看系统内的众测任务列表，并对众测任务进行搜索筛选和对指定属性进行排序
测试步骤	1. 整编人员登陆到系统 2. 整编人员点击下一页 3. 整编人员点击未审核数进行排序 4. 整编人员输入“信创”进行筛选 5. 整编人员点击任务名称
预期结果	1. 系统显示众包任务列表，并按照发布时间逆序排序。 2. 系统显示第二页众包任务 3. 众包任务列表按照未审核数量进行降序排序 4. 系统仅显示任务名称中包含“信创”的众包任务 5. 系统跳转到众测任务详情页
实际结果	与预期相符合

表5.3描述了用例查看任务详情的测试用例，主要查看该页面的审核情况，及对用户提交的原始报告进行查看。

表 5.3: 查看任务详情测试用例

测试 ID	TC2
测试名称	查看任务详情
待测功能	整编人员进入到任务详情页，查看任务的审核情况，查看用户提交的缺陷报告情况，并对报告进行排序和关键字检索
测试步骤	1. 整编人员进入到报告详情页 2. 整编人员点击表头，对报告 id 进行排序 3. 整编人员输入关键字“崩溃”进行检索 4. 整编人员点击某报告的超链接进行查看 5. 整编人员点击某报告所属的融合报告进行查看 6. 整编人员点击某报告所属的树状报告进行查看
预期结果	1. 系统显示任务详情页，并对任务的审核情况和原始报告进行展示 2. 系统按照报告 id 对用户的原始报告进行升序排序并展示 3. 系统仅显示报告描述中带有“崩溃”的报告 4. 系统跳转到单一报告页面 5. 系统跳转到融合报告详情页 6. 系统跳转到树状报告详情页
实际结果	与预期相符合

表5.4描述了报告融合用例的测试用例，主要是在任务详情页触发报告融合操作，并在页面等待融合结束。

表 5.4: 报告融合测试用例

测试 ID	TC3
测试名称	报告融合
待测功能	整编人员进入到报告详情页，并点击报告融合按钮，在页面等待融合结束查看报告的融合信息
测试步骤	1. 用户进入到未触发报告融合操作的任务的详情页 2. 用户点击“报告融合”按钮 3. 用户等待报告融合直至结束 4. 用户刷新系统，重新进入该页面 5. 用户点击某报告所属的融合报告
预期结果	1. 系统显示任务详情页，并显示报告融合按钮 2. 系统进行报告融合，并设置报告融合按钮不可点 3. 系统显示融合中，等待融合结束后刷新表格，逐份报告提示融合完成，并显示每份报告所属的融合报告 4. 系统不再显示报告融合按钮 5. 系统跳转到融合报告详情页面
实际结果	与预期相符合

表5.5描述了查看融合报告列表用例的测试用例，查看融合报告列表页面主

要展示了该任务报告的融合信息，展示了融合过后的报告簇。

表 5.5: 融合报告列表测试用例

测试 ID	TC4
测试名称	融合报告列表
待测功能	整编人员进入到报告融合列表页，浏览融合后系统的报告簇
测试步骤	1. 整编人员进入到已触发报告融合操作任务的融合报告列表页面 2. 整编人员选择“未审核”进行筛选 3. 整编人员点击图片进行查看 4. 整编人员点击某个报告簇右上角的展开按钮 5. 整编人员点击报告簇上的超链接
预期结果	1. 系统用卡片方式显示融合报告报告簇， 2. 系统仅展示审核状态为未审核的融合报告 3. 系统弹窗展示大图详情 4. 系统展示报告簇的补充点和歧义点信息 5. 系统跳转到融合报告详情页
实际结果	与预期相符合

表5.6描述了查看融合报告详情的测试用例，主要是在融合报告详情页，整编人员可以在该页面查看融合报告详细信息，包括融合报告簇主报告、补充点和歧义点，用户还可以在该页面查看可视化的融合树及知识图谱结构，整编人员可以在该页面生成交付报告，并设置该融合报告为已审核状态。

表 5.6: 融合报告详情测试用例

测试 ID	TC5
测试名称	融合报告详情
待测功能	整编人员进入到报告融合详情页面，查看融合详细信息，生成交付报告并设置审核状态。
测试步骤	1. 整编人员进入到融合报告详情页 2. 整编人员查看报告树并点击子节点 3. 整编人员展开补充点报告 4. 整编人员展开歧义点报告 5. 整编人员在页面点击编辑报告并拖拽图片 6. 整编人员点击审核
预期结果	1. 系统展示报告融合详细信息，包括主报告、补充点及歧义点 2. 系统展示子节点报告的基本信息，包括图片和文字等 3. 系统显示该融合报告的补充点信息 4. 系统展示该融合报告的歧义点信息 5. 系统展示报告编辑页面，并将图片添加到编辑框中 6. 系统将该报告设计为已审核状态并提示用户
实际结果	与预期相符合

表5.7描述了详情页报告融合的测试用例，主要是在融合报告详情页，整编人员可以在报告详情页查看系统所推荐的相似报告及推荐原因，并且能够将推荐报告加入到当前报告簇，再次进行报告融合操作。

表 5.7: 详情页报告融合测试用例

测试 ID	TC6
测试名称	详情页报告融合
待测功能	整编人员进入到报告融合详情页面，查看融合详细信息，查看系统推荐缺陷，选择推荐报告加入当前报告簇并融合。
测试步骤	1. 整编人员进入到融合报告详情页 2. 整编人员查看当前报告簇推荐报告 3. 整编人员选中一个报告点击“加入当前报告簇” 4. 整编人员点击“融合当前报告簇”
预期结果	1. 系统展示报告融合详细信息 2. 系统展示当前报告簇推荐报告并展示推荐原因 3. 系统将该报告加入当前报告簇 4. 系统开启融合，提示用户等待，融合完成后页面刷新
实际结果	与预期相符合

表5.8描述了查看树状报告列表的测试用例，查看树状报告列表页主要展示了该任务中的树状报告信息。

表 5.8: 树状报告列表测试用例

测试 ID	TC7
测试名称	树状报告列表
待测功能	整编人员进入到树状报告列表页，浏览任务中存在的树状报告
测试步骤	1. 整编人员待任务结束后进入到该任务的树状列表页 2. 整编人员选择“未审核”进行筛选 3. 整编人员点击图片进行查看 4. 整编人员点击某个树状报告右上角的展开按钮 5. 整编人员点击树状报告卡片上的超链接
预期结果	1. 系统用卡片的方式显示一颗报告树 2. 系统仅对审核状态为未审核的报告进行展示 3. 系统弹窗展示图片详情 4. 系统展示树状报告非根结点的报告信息 5. 系统跳转到树状报告详情页
实际结果	与预期相符合

表5.9描述了查看树状报告详情的测试，主要是在树状报告详情页面。整编人员可以在该页面查看树状报告的详细信息，报告内容包括图片和文字，整编

人员还可以在该页面查看该树状报告的可视化树状结构，整编人员可以在该页面进行报告整编，并设置该树状报告的审核状态。

表 5.9: 树状报告详情测试用例

测试 ID	TC8
测试名称	树状报告详情
待测功能	整编人员进入到树状报告详情页面，查看树状报告详细信息，生成交付报告并设置报告的审核状态
测试步骤	1. 整编人员进入到树状报告详情页 2. 整编人员展开非根节点信息 3. 整编人员点击报告上的链接 4. 整编人员点在该页面编辑交付报告并拖拽图片 5. 整编人员点击审核
预期结果	1. 系统展示树状报告的详细信息，包括各节点报告及可视化报告树 2. 系统显示该树状报告的所有节点报告信息 3. 系统跳转到单一报告页面 4. 系统展示交付报告编辑页面，并将图片添加到编辑框中 5. 系统将该报告状态设置为已审核并提示用户
实际结果	与预期相符合

表5.10描述了报告整编用例的测试用例，整编人员可以在融合报告详情页、树状报告详情页及单一报告页进行报告整编。

表 5.10: 报告整编测试用例

测试 ID	TC9
测试名称	报告整编
待测功能	整编人员进入报告详情页，对页面信息进行整合并生成交付报告，以及对交付报告的删改
测试步骤	1. 整编人员进入报告详情页 2. 整编人员填写交付报告信息并提交 3. 整编人员点击编辑 4. 整编人员点击删除 5. 整编人员拖拽图片 6. 整编人员设置报告审核状态为已审核
预期结果	1. 系统展示报告详情页 2. 系统保存交付报告并显示在报告列表中 3. 系统显示交付报告编辑框 4. 系统删除该交付报告 5. 系统将该图片添加到交付报告编辑框 6. 系统变更当前报告审核状态并提示给用户
实际结果	与预期相符合

表5.11描述了交付报告管理用例的功能测试，整编人员可以在交付报告页面进行交付报告的管理功能，主要是编辑和修改。并且整编人员可以对交付报告下载导出。

表 5.11: 交付报告管理测试用例

测试 ID	TC10
测试名称	交付报告管理
待测功能	整编人员进入交付报告管理页面并输入关键字进行搜索，整编人员点击两种格式的报告进行下载
测试步骤	1. 整编人员进入到交付报告页 2. 整编人员输入关键字“崩溃”进行筛选 3. 整编人员点击导出 Html 格式的交付报告 4. 整编人员点击导出 Excel 格式的交付报告
预期结果	1. 系统展示交付报告页的基本信息，包括该任务下的所有交付报告 2. 系统仅展示报告描述中包含“崩溃”关键字的交付报告 3. 系统提示下载 Html 格式的报告压缩包，一段时间后下载到本地 4. 系统提示下载 Excel 格式的报告压缩包，一段时间后下载到本地
实际结果	与预期相符合

5.3　可用性测试

本小节旨在对该系统的可用性进行测试，因该系统使用用户主要为企业内部员工，系统使用频率较低，因此该系统无需要求高可用性，但仍然需要系统宕机时间不超过 2 分钟，故对系统进行可用性测试，以证明该系统具备符合要求的可用性。

5.3.1　测试设计

本次系统可用性测试的服务器环境与功能测试相同，测试思路为在服务成功部署的服务器上，人为制造服务器宕机的情况，随后通过不断调用接口的方式，观察系统恢复时间，并进行记录，将记录结果同人为制造宕机时间进行比较，从而判断系统是否在规定时间内完成自启，是否满足可用性指标。

5.3.2　测试执行

如图5.2所示为系统的可用情况记录。10:01:23 秒人为将服务进程关闭，10:01:29 秒监测到服务进程关闭并进行重启。系统启动耗时 40 秒，监控服务于 10:02:20 秒再次请求冲程，服务一共宕机 40s，满足系统宕机不超过 2 分钟的要求。

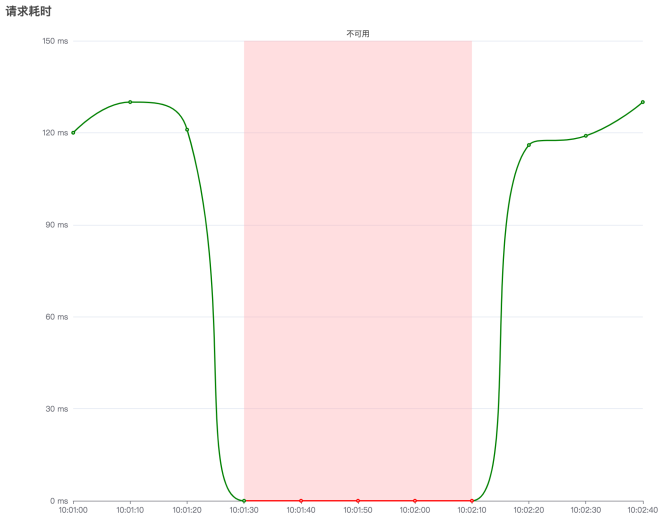

图 5.2: 系统可用性记录

5.4　可移植性测试

本小节旨在对该系统的可移植性进行测试。为响应信创国产操作系统的号召，支持国产操作系统，系统需要运行在主流国产操作系统上，故对系统在不同款国产操作系统上进行运行测试，以证明系统具有优良的可移植性。

5.4.1　测试设计

表 5.12: 可移植性测试服务器列表

操作系统名称	系统指令集
银河麒麟操作系统 V10	x86_64
银河麒麟操作系统 V10	arm
统信 UOS 操作系统	arm
统信 UOS 操作系统	mips
统信 UOS 操作系统	x86
阿里云 centos	x86_64

本次测试的思路是在不同操作系统的服务器行运行该系统，观察系统运行情况，并对系统的功能按照上一章节中的功能测试进行测试。具体的测试服务器基本信息如表5.12所示。测试服务器包含括 Mips、Arm、x86 架构的国产操作系统服务器，并使用主流的 x86_64 架构的 CentOS 服务器进行对比，参与测试的服务器一共为 5 台，用于对比的服务器 1 台。测试内容为测试系统在上述服务器上的启动情况及 TC1-TC10 共 10 个测试用例的通过情况。

5.4.2　测试执行

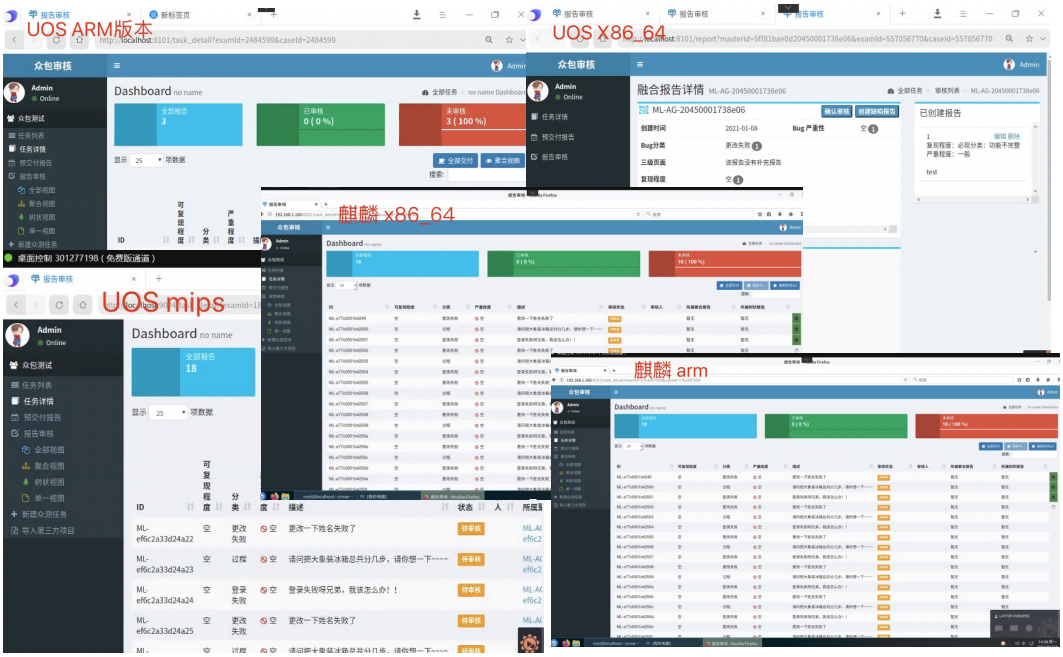

图 5.3: 可移植性运行截图

如图5.3所示，该系统在上述国产操作系统的服务器中均能正常运行，其中主要包括麒麟和 UOS 这两种操作系统，包含 x86_64、Arm 和 Mips 这三种系统指令集。测试结果表明，基于知识图谱的众测报告融合系统在上述国产操作系统中皆运行正常并通过系统功能性测试用例。除 Mips 指令集下的统信 UOS 操作系统因不支持 Docker 而使用 Java 部署外，其他系统均使用 Docker 部署完成，下表展示了本系统在以上操作系统中的运行情况。

表 5.13: 可移植性通过情况表

操作系统名称	系统指令集	是否正常启动	通过测试用例
银河麒麟操作系统 V10	x86_64	是	10/10
银河麒麟操作系统 V10	Arm	是	10/10
统信 UOS 操作系统	Arm	是	10/10
统信 UOS 操作系统	Mips	是	10/10
统信 UOS 操作系统	x86	是	10/10

5.5　性能测试

性能测试可以使用专用的性能测试工具来对系统接口正常、异常及峰值性能进行测试，从而对系统的性能指标进行判定是否符合预期，本节使用常用的性能测试工具 JMeter 来对系统关键接口进行性能测试。

5.5.1　测试设计

本节拟模拟 100 个用户来对系统接口进行测试，选用的接口为获取融合报告详情页数据。每个用户各发送 10 个请求，所有的线程在 10 秒内建立完毕。使用 JMeter 的“Response Time Graph”观察接口响应时间变化，JMeter 部分配置如下图5.4所示。

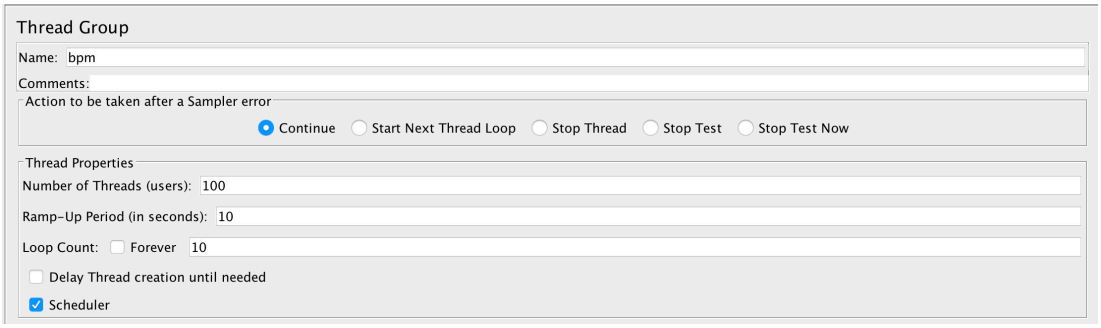

图 5.4: JMeter 配置

5.5.2　测试执行

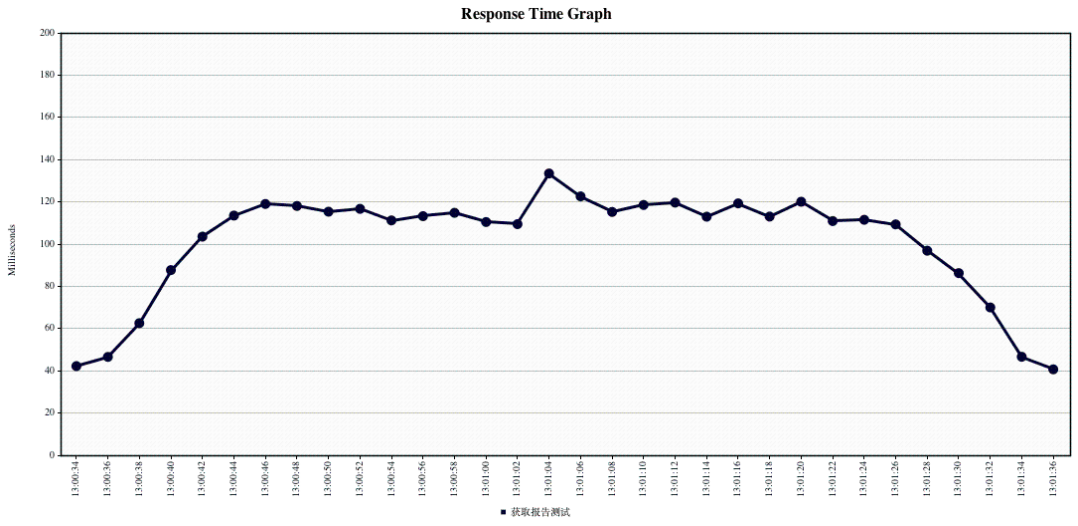

图 5.5: 获取报告数据接口响应时间图

图5.5展示了每秒系统平均响应时间变化图。结果显示，前十秒随着线程逐渐启动，接口响应时间不断增加，之后整体较为平稳，后十秒随着线程逐渐关闭，响应时间不断降低，整体不超过 140ms，展示了该接口具备良好的性能。

5.6　效果测试

效果测试可以通过进行对比实验的方式进行效果的验证，采用基于知识图谱的报告融合系统和基于词向量的报告融合系统进行对比，检测基于知识图谱的众测报告融合系统在众测报告整编过程中的效果，主要考虑对比的指标包括：生成报告簇数量、报告簇错误融合率、报告整编时间、交付报告数量和交付报告重复率。

5.6.1　测试设计

选取“McCafe 麦咖啡”为测试目标进行众包测试，所生成的缺陷报告分别进行报告整编工作。整编人员平均分为 A、B 两组，A 组使用基于知识图谱的众测报告融合系统进行报告整编，B 组使用基于词向量的报告融合系统进行整编。记录各项评测指标。

5.6.2　测试结果

共有 10 名报告整编人员参与测试，A、B 组各 5 名。任务总缺陷报告数为 468 份。以下指标皆为每组数据平均值，其中有效报告簇指的是簇大小大于 1 的报告簇，错误融合率指的是簇内包含明显不同 Bug 的报告簇比例，测试结果如表5.14所示。

表 5.14: A/B 测试执行结果

指标（平均）	A 组（实验组）	B 组（对照组）
有效报告簇数量	55	87
错误融合率	27.3%	56.3%
报告整编耗时	2 小时 13 分钟	2 小时 54 分钟
交付报告数量	98.4 份	104.2 份
交付报告重复率	2.03%	13.44%

根据测试结果可以看出，使用基于知识图谱的众测报告融合系统的 A 组虽然生成的有效报告簇的数量少于对照组，但是报告簇错误融合率大幅下降，从 56.3% 下降到 27.3%，报告整编总耗时也从 2 小时 54 分钟降低到 2 小时 13 分钟。整编过程生成的交付报告中，交付报告的重复率从 13.44% 降低到 2.03%。结果

显示 A 组耗时更短，效率更高，交付报告质量更优，整编人员的效率得到了明显提升。

5.7　本章小结

本章节主要对基于知识图谱的众测报告融合系统进行了测试，测试指标为第三章中描述的系统功能需求、非功能需求及效果测试。从功能需求测试、可移植性测试、可用性测试、性能测试及效果测试的角度对系统进行了严格的测试。测试结果表明，系统功能测试已经达到规格要求，系统功能运行正常。可用性测试显示系统具备一定程度的可用性，在可移植性测试方面，测试表明系统可以在包括 Arm、x86、Mips 及 x86_64 的国产操作系统（主要是统信 UOS 和银河麒麟）上正常运行，具备良好的可移植性，性能测试展示系统在高并发场景下具备良好的性能。效果测试结果展示系统能够有效提升整编人员的使用效率。

第六章　总结与展望

6.1　总结

线上开展的众包测试具有生成的报告数量多、报告质量参差不齐且重复率高的特点，这些特点为整编人员整编原始报告生成交付报告带来了极大的困难。而相似报告检测中，基于词向量的文本相似度检测不能挖掘文本间的语义关系，重复报告识别准确度不高，故本文设计了基于知识图谱的众测报告融合系统，通过引入知识图谱技术来改善重复报告检测。在众包测试报告融合系统中，创新性的引进了知识图谱技术。此外，本系统完善了报告整编的流程，使得报告整编过程更加高效。其次，对于重复报告组成的报告簇使用 PageRank 算法计算报告在报告簇中的排序，进而提取重复报告簇中的主报告，并对报告描述文本进行了补充点和歧义点提取，提取了报告簇中除主报告外的关键信息。众测报告经过报告融合后，系统中的重复报告大多被聚合在同一报告簇中，并且就报告簇提取了主要报告、补充点和歧义点供整编人员查看，这极大的提高了整编人员识别重复报告、获取报告内容的效率，交付报告的质量也得到了极大提升。

本文的主要工作如下：首先就知识图谱、报告相似检测和报告摘要算法的研究现状进行了阐述。本文对缺陷报告相似性度量进行了较为深入的探索，并就报告融合系统现状进行了分析，在考虑到融合系统当前痛点的基础上引入知识图谱提高重复报告识别的准确度，使用 PageRank 算法和自然语言处理技术对报告关键内容进行了提取。其次，本文完成了众测报告融合系统的需求分析并进行了主要设计，经过分析，众测报告融合系统分为知识图谱模块、图片计算模块、知识图谱报告融合模块和报告整编模块共 4 个模块。为了保证系统的性能，本文引入了 Redis 作为系统缓存；为了保证系统的可用性，本文使用 Nginx 进行负载均衡。最后，本文完成了系统的具体实现，并就系统实现进行了详细的测试，本文对系统的功能和非功能需求均进行了测试，测试结果证明系统功能完备，具备优良的性能。

6.2　展望

目前，本系统已经开发完成并线上部署，系统已初步完善，但是还有许多需要改进的地方，主要包括以下几个方面。

（1）自动化生成交付报告。目前系统中主要对报告进行了报告融合操作，其中知识图谱用于计算报告之间的相似度，融合结束之后需要整编人员手动进行

整编，以后的报告融合系统可以就自动化生成测试报告进行探索，彻底解放整编人员，使得系统更加的智能化。

（2）使用基于位置匹配的图片相似度算法，目前系统中使用的图片相似度算法基于整张图片进行特征提取并进行比较。在移动 App 众包测试中，由于用户提交的截屏比例相似，该算法在移动 App 众测中具有很好的适应性。但是在 Web 网站的众包测试中，由于用户通常不是全屏截图，有些用户提交的截图所截的面积大，有些用户提交的截图所截的面积小，该图片相似度算法在这种众包任务中，所识别的图片相似度通常差异较大，是未来一大改进方向。

参考文献

[1] H. J, The rise of crowdsourcing, Wired Magazine 16 (6) (2006) 1–4.

[2] 冯剑红, 李国良, 冯建华, 众包技术研究综述, 计算机学报 38 (9) (2015) 1713–1726.

[3] Z. Q.-H. Zhao Yu-Xiang, Evaluation on crowdsourcing research:current status and future direction.information systems frontiers 11 (1) (2012) 1–18.

[4] H. M. J. Y. Mao K, Capra L, A survey of the use of crowdsourcing in software engineering.journal of systems and software 126 (2017) 57–84.

[5] 章晓芳, 冯洋, 刘頔, 陈振宇, 徐宝文, 众包软件测试技术研究进展, Journal of Software 29 (1).

[6] H. A. Latoza T, Crowdsourcing in software engineering: Models, motivations, and challenges, IEEE software 33 (1) (2016) 74–80.

[7] J. M. L. S. Zogaj, U. Bretschneider, Managing crowdsourced software testing: a case study based insight on the challenges of a crowdsourcing intermediary, Journal of Business Economics 84 (3) (2014) 375–405.

[8] Y. Zhao, Q. Zhu, Evaluation on crowdsourcing research: Current status and future direction, Information Systems Frontiers 16 (3) (2014) 417–434.

[9] A. H. E. S. D. M.Allahbakhsh, B.Benatallah, Quality control in crowdsourcing systems: Issues and directions, ieee internet computing, Journal of Business Economics 17 (2) (2013) 76–81.

[10] H. L.-r. W. Y.-f. XU Zeng-lin, SHENG Yong-pan, Review on knowledge graph techniques, Journal of University of Electronic Science and Technology of China 45 (4) (2016) 589–606.

[11] J. Webber, A programmatic introduction to neo4j, in: Proceedings of the 3rd annual conference on Systems, programming, and applications: software for humanity, 2012, pp. 217–218.

[12] A. G.-D. J. W. O. Y. Antoine Bordes, Nicolas Usunier, Translating embeddings for modeling multi-relational data, Neural Information Processing Systems (NIPS).

[13] J. F.-Z. C. Zhen Wang, Jianwen Zhang, Knowledge graph embedding by translating on hyperplanes, Department of Information Science and Technology.

[14] M. S.-Y. L. X. Z. Yankai Lin, Zhiyuan Liu, Learning entity and relation embeddings for knowledge graph completion, Proceedings of the Twenty-Ninth AAAI Conference on Artificial IntelligenceJanuary (2015) 2181–2187.

[15] O. N. P. Runeson, M. Alexandersson, Detection of duplicate defect reports using natural language processing, In Proceedings of International Conference on Software Engineering (2007) 499–510.

[16] W. W. JALBERT N, Automated duplicate detection for bug tracking systems[c], IEEE International Conference on Dependable Systems and Networks with Ftcs and DCC (2008) 52–61.

[17] X. T.-e. a. WANG X, ZHANG L, An approach to detecting duplicate bug reports using natural language and execution information[c], IEEE International Conference on Software Engineering (2008) 461–470.

[18] M. G. C. SOMASUNDARAM K, Automatic categorization of bug reports using latent dirichlet allocation, India Software Engineering Conference (2012) 125–130.

[19] A. T. Nguyen, T. T. Nguyen, T. N. Nguyen, D. Lo, C. Sun, Duplicate bug report detection with a combination of information retrieval and topic modeling, in: 2012 Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering, IEEE, 2012, pp. 70–79.

[20] A. A. A. Hindle, E. Stroulia, A contextual approach towards more accurate duplicate bug report detection and ranking, Empirical Software Engineering 21 (2) (2016) 368–410.

[21] X. X.-L. B. X. Yang, D. Lo, J. Sun, Combining word embedding with information retrieval to recommend similar bug reports, In Proceedings of International Symposium on Software Reliability Engineering (2016) 127–137.

[22] Z. C. Y. Feng, J. A. Jones, C. Fang, Multi-objective test report prioritization using image understanding, In Proceedings of Automated Software Engineering (2016) 202–213.

[23] S. W. T. M. J. Wang, M. Li, Q. Wang, Cutting away the confusion from crowdtesting, arXiv preprint.

[24] 余笙, 李斌, 孙小兵, 薄莉莉, 周澄, 知识驱动的相似缺陷报告推荐方法 [j/ol], 计算机科学 (2021) 1–14.

[25] L. H. P, The automatic creation of literature abstracts, IBM Journal Research and Development 2 (2) (1958) 159–165.

[26] G. Salton, C. Buckley, Term-weighting approaches in automatic text retrieval, Information processing & management 24 (5) (1988) 513–523.

[27] R. A. EL-Beltagy S R, Kp-miner: participation in semeval-2, The International Workshop on Semantic Evaluation (2010) 190–193.

[28] R. M. T. W. L. Page, S. Brin, The pagerank citation ranking: Bringing order to the web, Stanford InfoLab.

[29] R. Mihalcea, P. Tarau, Textrank: Bringing order into text, in: Proceedings of the 2004 conference on empirical methods in natural language processing, 2004, pp. 404–411.

[30] M. R, Graph-based ranking algorithms for sentence extraction, applied to text summarization, Association for Computational Linguistics (2004) 1–20.

[31] F. H. e. a. Khan A, Salim N, Abstractive text summarization based on improved sematic graph approach, International Journal of Parallel Programming (2018) 1–25.

[32] J. A. J. Y. L. R. Hao, Y. Feng, Z. Chen, Ctras: Crowdsourced test report aggregation and summarization, in proceedings of international conference on software engineering.

[33] T. Y. W. Hinton G E, Osindero S, A fast learning algorithm for deep beliefets, Neural Computation 18 (7) (2006) 1527–1554.

[34] Z. B. Nallapati R, Xiang B, Sequence-to-sequence rnns for text summarization, arXiv preprint arXiv (2016) 1–13.

[35] L. O. BERNERS-LEE T, HENDLER J, Scientific american magazine, Journal of University of Electronic Science and Technology of China 23 (1) (2008) 1–4.

[36] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, arXiv preprint arXiv:1301.3781.

[37] 赵京胜, 宋梦雪, 高祥, 自然语言处理发展及应用综述, 信息技术与信息化 (2019) 143–146.

[38] L. Page, S. Brin, R. Motwani, T. Winograd, The pagerank citation ranking: Bringing order to the web., Tech. rep., Stanford InfoLab (1999).

[39] M. Slee, A. Agarwal, M. Kwiatkowski, Thrift: Scalable cross-language services implementation, Facebook white paper 5 (8) (2007) 127.

[40] J. Chen, A. Wang, J. Chen, Y. Xiao, Z. Chu, J. Liu, J. Liang, W. Wang, Cn-probase: a data-driven approach for large-scale chinese taxonomy construction, in: 2019 IEEE 35th International Conference on Data Engineering (ICDE), IEEE, 2019, pp. 1706–1709.

[41] J. Deacon, Model-view-controller (mvc) architecture, Online][Citado em: 10 de março de 2006.] http://www. jdl. co. uk/briefings/MVC. pdf.

简历与科研成果

基本情况 李文龙，男，汉族，1997 年 4 月出生，安徽省安庆市人。

教育背景

2019.09 ~ 2021.06 南京大学软件学院 硕士

2015.09 ~ 2019.06 南京大学软件学院 本科

致　　谢

在本文完成之际，我要向在项目实现、论文编写过程中，对我进行指导和帮助的人，表达我最诚挚的感谢和最珍贵的祝福。

首先我要感谢我的导师陈振宇教授，感谢他在我研究生这两年时间内对我的指导和照顾。在我大四时就指导我完成本科毕业论文的撰写，引导我进入慕测平台完成部分开发工作，这使我收益匪浅，将我过去几年中在书本上学习到的知识成功得以实践。感谢冯洋老师，感谢他在论文撰写过程中给了我许多建议。

在 ise 实验室相互了解、相互成长的这三年时间里，我对软件工程这个学科本身有了更加深入的理解，对于软件开发这件事有了更加深刻的认识，这几年中宝贵的经历将使我有了更深层次的思考和对未来有了更加热切的期盼。

同时我还要感谢我的同学们，特别是徐佳炜和段梦洋，自从进入实验室以来我们就是一个小组的成员，大家相互激励、互相学习。感谢我的舍友李伟民、田贵松和黄国成，感谢他们在学习和生活上对我的无私的帮助，正是同学们的帮助，才让我有了一个难忘的、充满回忆的研究生生涯。

这里需要特别感谢我的家人和女朋友。他们是我强大的后盾与精神的港湾，这是我十几年来努力学习认真工作的动力源泉。感谢他们给予的爱和关怀。

感谢 Joel Hanson 先生，您创作的《Traveling Light》陪伴了我无数个夜晚。每当压力袭来的时候，这支优美的歌总能使我拨云见日。

最后，对能在百忙之中抽出时间审阅我论文的各位老师、专家们表示由衷的感谢!