



南京大學

研究生畢業論文 (申請工程碩士學位)

論 文 題 目 面向可信眾測激勵的
區塊鏈服務系統設計與實現

作 者 姓 名 張皓明

學 科、專 業 名 稱 工程碩士（軟件工程領域）

研 究 方 向 軟件工程

指 導 教 師 陳振宇 教授

2021 年 5 月 25 日

学 号：MF1932238

论文答辩日期：2021 年 5 月 20 日

指 导 教 师： (签字)

The Design and Implementation of a Blockchain Service System Oriented to Credible Crowdsourced Testing Incentives

by

Haoming Zhang

Supervised by

Professor Zhenyu Chen

A dissertation submitted to
the graduate school of Nanjing University
in partial fulfilment of the requirements for the degree of

MASTER OF ENGINEERING

in

Software Engineering



Software Institute
Nanjing University

May 25, 2021

学位论文原创性声明

任何收存和保管本论文的单位和个人，未经作者本人授权，不得将本论文转借他人并复印、抄录、拍照或以任何方式传播，否则，引起有碍作者著作权益的问题，将可能承担法律责任。

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含其他个人或集体已经发表或撰写的作品成果。本文所引用的重要文献，均已在文中以明确方式标明。本声明的法律结果由本人承担。

论文作者签名：_____

日期：_____年____月____日

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：面向可信众测激励的区块链服务系统设计与实现

工程硕士（软件工程领域） 专业 2019 级硕士生姓名：张皓明
指导教师（姓名、职称）：陈振宇 教授

摘 要

众包测试是一种以众包方式进行的软件测试活动，由需求方、工人方和平台方三方共同参与并协同合作，凭借用户广泛与反馈快速等特点以实现测试能力与效率方面的提升。在众包测试中，为了吸引更多工人报名参与任务，同时也为了鼓励工人提高测试质量从而更好地完成任务，平台常常会对工人方进行激励。然而，现有众测平台的激励主要存在着两大问题：一是当前平台常见的激励机制往往效果有限，劳动力评估方式不科学或是分配方式不公平，导致无法保障工人方权益；二是奖励分配所依据的数据往往由第三方平台中心化地存储和管理，易受到单点攻击或者出现平台数据欺诈的现象。

本文结合反向拍卖模型设计了可信评分在线众测激励机制来实现激励的正向效果，以解决众测激励中有效性不足的问题；采用了与区块链相结合的存储方式来实现数据的真实可靠性与流程的透明公开性，以解决众测激励中可信性不足的问题。本文设计与实现了面向可信众测激励的区块链服务系统，联合需求方、工人方、众测平台方三方作为区块链节点搭建联盟链集群平台。本文通过智能合约将众测流程基本信息实时接入区块链平台作为分配依据，当执行奖励分配时先通过数据溯源从账本中获取依据的数据信息，再由合约脚本自动化根据在线评分激励机制完成积分分配并将结果存入区块链平台中以记录过程，从而实现众包测试的可信众测激励；当执行奖励查询时则直接通过账本数据获取激励结果信息。技术实现层面，本文采用 BaaS 区块链即服务架构进行开发，数据层构建区块链基础平台采用 Hyperledger Fabric 框架提供区块链服务，众测激励服务层采用 Springboot 框架，前端采用 Vue 框架结合 Fabric Explorer 展示。

本文在功能、性能与安全性方面对系统进行了测试。系统功能上实现了可信众测激励；性能上并发量、响应时间、吞吐量等指标均满足真实环境下的业

务需求；安全性上区块链智能合约检验结果良好，并无漏洞。同时，本文接入众测月度赛的真实案例数据来模拟众测流程，结果同样可证明系统在大量负载下能保持良好性能，并且经检验确实所有工人均依据报告评分获得的奖励分配，激励数据无误且公开透明。本系统实现了众测激励流程的有效性与可信性，进而推动众包测试继续发展。

关键词： 众包测试，激励机制，区块链，智能合约

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 a Blockchain Service
System Oriented to Credible Crowdsourced Testing Incentives
SPECIALIZATION: Software Engineering
POSTGRADUATE: Haoming Zhang
MENTOR: Professor Zhenyu Chen

Abstract

Crowdsourced testing is a software testing activity by crowdsourcing, which is jointly participated and cooperated by requesters, workers and the platform. With the characteristics of wide range of users and rapid feedback, the testing ability and efficiency can be improved. In order to attract more workers to sign up and participate in testing tasks, and to encourage workers to improve the quality of the test so as to better complete the task, the platform often provides incentives to them. However, there are two major problems in the incentives of existing crowdsourced testing platforms: For one thing, the data on which the rewards are distributed is often stored and managed centrally by third-party platforms, which is vulnerable to single-point attacks or data fraud; For another, the common incentive mechanisms used in current platforms often have limited effect, and the labor force evaluation method is unscientific or the distribution mode is unfair, which makes it difficult to protect the interests of workers.

In this thesis, we design an online scoring incentive mechanism to achieve the positive effect of incentives, so as to solve the problem of insufficient effectiveness in crowdsourced testing incentives; we adopt a storage method combined with blockchain to achieve data authenticity and process transparency, so as to solve the problem of insufficient credibility in crowdsourced testing incentives. This thesis designs and implements a blockchain service system oriented to credible crowdsourced testing incentives, which unites requesters, workers and the platform as blockchain nodes to build a consortium blockchain cluster platform. This thesis uses smart contracts to access the basic data of the crowdsourced testing process on the blockchain platform as the basis for distribution in real time. When the reward distribution is executed, the data information

is obtained through data provenance from the ledger, and then the contract script is automated to complete the reward distribution according to the online scoring incentive mechanism. The result of distribution is also stored in the blockchain platform to record the process and to realize the credible crowdsourced testing incentives. When the reward query is executed, the incentive result information is obtained directly through the ledger data. In technical implementation, this thesis adopts the BaaS (Blockchain as a Service) architecture for development. The data layer builds the blockchain basic platform using the Hyperledger Fabric to provide blockchain services, the crowdsourced testing incentive service layer uses Springboot, and the front-end uses Vue combined with Fabirc Explorer to display.

This thesis has carried on the function, the performance and the safety test to the system. In terms of system functions, the system has realized such functions as blockchain data storage and query, crowdsourced testing incentive, etc. In terms of system performance, concurrency, response time, throughput and other indicators all meet the business requirements in the real environment. In terms of system security, the system blockchain smart contract has been static analyzed and there are no common security vulnerabilities and hidden dangers. At the same time, this thesis uses the real case data of the crowdsourced testing monthly competition to simulate the process. The results also prove that the system can maintain good performance under a large load. Moreover, it is verified that all workers are really allocated rewards according to the report score, and the incentive data is correct, open and transparent. The system ensures the validity and credibility of the crowdsourced testing incentive process, which promotes the continued development of crowdsourced testing.

keywords: Crowdsourced Testing, Incentive Mechanism, Blockchain, Smart Contract

目 录

目 录	v
插图清单	ix
附表清单	xiii
第一章 引言	1
1.1 项目背景与意义	1
1.2 国内外研究现状	2
1.2.1 众测激励技术	2
1.2.2 区块链技术	4
1.3 本文的主要工作	5
1.4 本文的组织结构	6
第二章 技术综述	7
2.1 众测激励技术	7
2.1.1 众包测试概述	7
2.1.2 众测激励	9
2.2 区块链技术	10
2.2.1 区块链概述	10
2.2.2 区块链架构	10
2.2.3 区块链分类	13
2.2.4 Hyperledger Fabric	14
2.3 本章小结	14
第三章 可信众测激励方法	15
3.1 引言	15
3.2 可信众测激励设计	16
3.2.1 激励有效性设计	16
3.2.2 激励可信性设计	18
3.3 可信众测激励方法	19
3.3.1 需求提交	19

3.3.2 报告提交	20
3.3.3 报告聚合	21
3.3.4 奖励分配	22
3.4 本章小结	24
第四章 系统需求分析与总体设计	25
4.1 系统整体概述	25
4.2 系统需求分析	25
4.2.1 系统涉众分析	25
4.2.2 功能性需求分析	26
4.2.3 非功能性需求分析	30
4.3 系统总体设计	31
4.3.1 系统架构	31
4.3.2 4+1 视图	33
4.4 持久化模型设计	36
4.5 本章小结	39
第五章 系统详细设计与实现	41
5.1 众测激励系统服务设计与实现	41
5.1.1 众测任务服务	41
5.1.2 任务需求服务	43
5.1.3 测试报告服务	46
5.1.4 最终报告服务	49
5.1.5 积分奖励服务	51
5.2 区块链服务设计与实现	54
5.2.1 对外服务层	55
5.2.2 基础设施层	59
5.3 系统界面展示	61
5.4 本章小结	64
第六章 系统测试与案例分析	65
6.1 测试环境	65
6.2 测试指标	66
6.3 测试设计	66
6.3.1 功能测试设计	66

目 录	vii
6.3.2 性能测试设计	70
6.3.3 安全测试设计	71
6.4 测试结果分析	72
6.4.1 功能测试结果分析	72
6.4.2 性能测试结果分析	72
6.4.3 安全测试结果分析	74
6.5 案例分析	75
6.6 本章小结	78
第七章 总结与展望	79
7.1 总结	79
7.2 展望	80
致 谢	81
参考文献	83
简历与科研成果	91
《学位论文出版授权书》	93

插图清单

2-1	众包测试流程图.....	7
2-2	众包测试报告实例图.....	8
2-3	离线与在线激励机制对比图.....	9
2-4	区块链基础架构图.....	11
2-5	Farbic 工作流程图.....	14
3-1	激励有效性设计路线图.....	16
3-2	激励可信性设计路线图.....	18
3-3	可信众测激励流程图.....	19
3-4	需求提交流程图.....	20
3-5	报告提交流程图.....	20
3-6	报告聚合流程图.....	21
3-7	奖励分配流程图.....	22
4-1	系统用例图.....	26
4-2	系统架构图.....	31
4-3	系统逻辑视图.....	33
4-4	系统开发视图.....	34
4-5	系统进程视图.....	34
4-6	系统物理视图.....	35
5-1	众测任务服务核心类图.....	41
5-2	众测任务溯源顺序图.....	42
5-3	众测任务溯源关键代码.....	43
5-4	任务需求服务核心类图.....	44
5-5	任务需求提交顺序图.....	44
5-6	任务需求溯源顺序图.....	45
5-7	任务需求提交关键代码.....	45

5-8 任务需求溯源关键代码	45
5-9 测试报告服务核心类图	46
5-10 测试报告提交顺序图	47
5-11 测试报告溯源顺序图	47
5-12 测试报告提交关键代码	48
5-13 测试报告溯源关键代码	48
5-14 最终报告服务核心类图	49
5-15 最终报告提交顺序图	49
5-16 最终报告数据溯源顺序图	50
5-17 最终报告提交关键代码	50
5-18 最终报告溯源关键代码	51
5-19 积分奖励服务核心类图	52
5-20 积分奖励提交与分配顺序图	52
5-21 积分奖励溯源顺序图	53
5-22 积分奖励提交与分配关键代码	53
5-23 积分奖励溯源关键代码	54
5-24 区块链基础平台核心类图	55
5-25 区块链平台服务层数据溯源顺序图	55
5-26 区块链平台服务层数据上链顺序图	56
5-27 区块链平台服务层数据溯源关键代码	56
5-28 区块链平台服务层数据上链关键代码	57
5-29 智能合约关键代码	57
5-30 智能合约 save 方法关键代码	58
5-31 CouchDB 与 Fabric CA 组件部署关键代码	59
5-32 Orderer 节点部署关键代码	60
5-33 Peer 节点部署关键代码	61
5-34 需求方主页界面	61
5-35 需求方报告聚合溯源界面	62
5-36 工人方主页界面	62
5-37 工人方测试报告溯源界面	63
5-38 工人方激励结果溯源界面	63
6-1 区块链平台智能合约高并发性能测试结果	74

6-2 众测月度赛数据示例	75
6-3 平台方数据统计界面	76
6-4 平台方奖励分配界面	76
6-5 众测月度赛响应时间性能统计	77
6-6 众测月度赛 CPU 占用性能统计	78

附表清单

2-1	共识机制 PoW、PoS 与 PBFT 对比	12
2-2	公有链、联盟链与私有链对比	13
4-1	众测数据传输用例描述	27
4-2	众测流程溯源用例描述	28
4-3	众测任务溯源用例描述	28
4-4	众测报告溯源用例描述	29
4-5	众测奖励分配用例描述	29
4-6	众测奖励查询用例描述	30
4-7	众测任务数据模型	36
4-8	需求提交数据模型	36
4-9	需求审核数据模型	37
4-10	报告提交数据模型	37
4-11	报告审核数据模型	38
4-12	报告聚合数据模型	38
4-13	奖励分配数据模型	39
6-1	系统测试环境	65
6-2	系统测试评价指标说明	66
6-3	众测数据传输测试用例	67
6-4	众测任务溯源测试用例	67
6-5	众测报告溯源测试用例	68
6-6	众测流程溯源测试用例	68
6-7	众测奖励分配测试用例	69
6-8	众测奖励查询测试用例	70
6-9	系统服务端关键接口	70
6-10	区块链平台智能合约关键方法	71
6-11	区块链平台待检测智能合约漏洞	72

6-12 功能测试用例结果	72
6-13 服务端接口性能测试结果	73
6-14 区块链平台智能合约方法性能测试结果.....	73
6-15 区块链平台资源占用测试结果	73
6-16 区块链平台智能合约安全测试结果	74
6-17 众测月度赛数据统计	75

第一章 引言

1.1 项目背景与意义

众包方式是指借助互联网聚集众人，通过群体的努力来共同完成此前由于人数不足以致无法进行的任务 [1]。众包测试是指一种多方参与的、以众包方式进行的软件测试活动 [2]。众包测试不同于传统测试，通过其更多用户群体的规模与更接近真实的场景，以达到测试能力与效率的改进。作为任务中实时的数字资产，众测数据分别由三类参与方提供，包括需求方提交的需求文档、工人方提交的测试报告和平台方交付的最终报告等资产。

众测平台随着规模的扩大，对于参与者的数量和质量的要求越来越高。在众包测试中，为了吸引更多工人报名参与任务，同时也为了鼓励工人提高测试质量，平台常常会对其进行激励。然而，现有的众包测试平台的激励主要存在着一定问题。一方面，当前平台常用的激励机制普遍效果有限，由于劳动力评估或分配方式不公平进而难以达到各方的利益平衡，需要设计一种合适的激励机制以保障参与方权益。另一方面，众测通常将报告及评分等数据作为激励的依据，然而这些关键性的参照数据往往由第三方平台使用传统数据库进行中心化的存储和管理，很可能造成单点攻击等安全性问题或者是数据欺诈等真实性问题，因此需要通过公开透明的激励流程来保证真实性。

区块链是一种分布式账本，按时间顺序对交易记录进行存储，具有去中心化、可追溯、难以篡改的特性 [3]。在区块链中，账本数据由网络中所有验证节点共同维护，仅当通过共识后才能作为区块插入链尾，此前的数据难以篡改，进而保障了数据的安全性。针对上述众测激励问题，我们将区块链与众测激励相结合，利用区块链多节点的特性将众测流程中各个参与方视为节点，设计了结合区块链的可信在线激励机制与可信众测激励流程，通过结合反向拍卖模型设计的可信评分在线众测激励机制进行劳动力公正评估与分配，解决了分配方式不公平导致激励效果不佳的问题；通过区块链的去中心化特性与不可篡改性进行分布式可靠存储，解决了第三方数据真实性存疑的问题；通过区块链的溯源特性进行劳动力资产的全链路确权，解决了激励过程难以公开透明的问题。

本文将需求方、工人方、平台方三方视为节点，构建区块链基础平台并应用于可信众测激励。本文通过智能合约将众测流程中数据实时导入区块链平台作为分配的依据，当执行奖励分配时，先通过溯源从账本中获取激励所依据的数据信息，再通过在线评分激励机制完成激励并存入区块链平台以记录结果；当执行奖励查询时，则直接通过账本来获取激励结果数据。本文在理论上，深入研究了区块链与众测激励技术，给出了基于区块链的可信众测激励方法；在实践上，将上述方法实际应用于众包测试，进一步推动了众测激励的发展。

1.2 国内外研究现状

本文的研究集中在基于区块链的可信众测激励，针对上述技术，下文将从众测激励、区块链两个方面讨论国内外的研究现状。

1.2.1 众测激励技术

众包测试作为当前软件测试领域中的新分支，通过其云平台的优势 [4]，依赖于大量工人在多平台上进行更真实快速的测试，在传统软件测试方法的基础上实现了效率与质量方面的改造与升级。当前众包测试领域的研究聚焦于众包测试流程的完善与质量的提升方面 [5]，众测激励即为其中之一。激励，即以一定的奖惩性措施来激发与引导组织成员的行为，从而有效实现组织的目标 [6]。同理，众测激励则是众测平台通过奖励的方式鼓励用户积极地参与任务与高质量地完成任务 [7]。因此，如何形成良性的众测激励已成为当前众测激励研究的重点之一。在激励过程中需要考虑的因素有两方面，即激励方式以及激励机制的选择。

在激励方式方面，Lakhani [8] 等人的研究表明金钱等奖励手段是个体参与众包的关键动机。目前大部分众包平台正是通过设置高额的奖金来吸引广泛的参与者，比如猪八戒平台^① [9]。然而，还有很多众包平台是并没有任何奖励提供的，他们所吸引参与者的更多是内在的原因。Huberman [10] 等人发现，在 Youtube 网站上，当用户没有收到多少关注时会丧失继续上传的动力；而当用户持续受到关注时则会不断积极进行视频分享。夏恩君 [11] 等人总结到，个体参与众包活动的动因既包括内部动机也包括外部动机，而其中主要的内部动机就是当参与者完成任务时所获得的成就感。

^①猪八戒平台. <https://www.zbj.com/>

在激励机制方面,传统的激励机制包括离线和在线两种类型 [6]。离线激励机制是指平台在任务开始之前,利用拍卖、博弈等手段对所有任务进行分配,任务结束以后支付报酬 [12]。在线激励机制是根据用户的参与情况、任务完成情况等因素,由平台实时决策,对用户最终所付报酬进行动态调整的过程。Yang [12] 等人提出了一种基于 Stackelberg 游戏的离线激励机制与其两种模型:以平台为中心的模型,即平台主动提供了用户共享的奖励;以用户为中心的模型,即用户主动决定最终的支付金额。然而,离线激励机制假定所有用户从任务开始就出现,随后不能接受新的投标,并且默认工人一定会完成该次任务 [13],因此可能会出现工人未提交报告仍获得奖励,或是工人无法放弃已接收任务并死锁的情况。为了解决上述问题,Zhang [14] 等人基于拍卖法做出了改进,提出了一种基于阈值拍卖的在线激励机制。该机制设定了最低奖励,按照每个工人完成任务的表现进行累加并分配,较好地解决了离线激励机制中工人不作为的问题,根据任务完成的情况相应的调整奖励,使激励达到了相对公平,然而使用的拍卖法在用户数量上和工人效率上有着不足 [15]。在拍卖法中,定价是以工人中心,确实较为适合在移动众包等大量需要用户且专业性要求不高的场景 [16];然而在众包测试场景下,往往需要测试工人有一定的专业性且任务中需要的工人数量相对较少,拍卖法对于需求方来说可能会导致任务总开销成本过高。针对拍卖法中的问题, Lee [17] 等人提出了反向拍卖法,由需求方来定价,随后工人接收后按照任务获得奖励。反向拍卖法解决了工人逃单的问题并能够控制需求方成本,从而平衡各方权益。

在线激励机制中,对奖励进行合理且公平的分配是用户权益的重要保障。众测激励主要依据是通过绩效考核方式评价众包工人的任务完成情况 [18],类似于按劳分配。Chittilappilly [19] 等人提出了两种主要的绩效考核方式,一种是按照任务量评估,并结合任务的难易程度,比如 uTest^①、Testin^② 等平台;另一种是按照缺陷的检测结果评估,并结合缺陷严重程度,比如 MoocTest 慕测^③ 等平台。Xie 等人在 [20, 21] 中提出基于评分的激励机制,要求平台管理员依据对工人解决方案质量的评分来激励,其本质即为按照缺陷的检测结果评估的方式。Xie 等人也比较了需求方打分与平台打分两种方式。若由需求方打分,虽然完全符合其意愿,但是可能会由于私心、偏见等导致评分不公平;由平台打分,作为第三方打分较为公正,可通过盲审以避免上述情况的发生。

^①uTest. <https://www.utest.com/>

^②Testin. <https://www.testin.cn/>

^③MoocTest 慕测平台. <http://service.moocetest.net/>

在众包测试平台的激励中,除了激励机制的选择以外,激励过程的可信性也是当前众多众包平台需要着重考虑的一个因素。由于激励中分配过程的主要参照依据为众包平台此前所记录的数据,该数据的可靠与否将直接影响到工人的获得的利益,因此需要通过手段保证众测数据的可信性。然而,当前主流的众包平台,无论是国内的百度众测^①、阿里云众包^②、Testin,还是国外的Google Test Cloud^③、Amazon Mechanical Turk^④、Applause^⑤等,在设计上都并未有突出考虑到激励可信性的方面。大多数平台都依赖于中心化服务器,这会受到传统基于信任的模型的弱点,例如单点故障 [22];由于恶意用户的参与,它们还容易受到分布式拒绝服务 (DDoS) 和 Sybil 攻击 [23]。

综上所述,当前主流众包平台的激励存在着两大问题:一是激励有效性不足,即激励机制对于工人的劳动力评估不公平导致权益受损;二是激励可信性不足,即依据的众测数据由平台中心化存储从而存在数据故障或欺诈的隐患。

1.2.2 区块链技术

2008 年中本聪发表了《比特币:一种点对点的电子现金系统》[24],标志着比特币的诞生。2009 年随着创世区块与后继区块相连接形成了链,区块链也随之正式出现 [25]。2013 年 Buterin 提出了以太坊区块链平台 [26],不仅能够通过其定义的以太币来进行数字化货币交易,而且设置了可编辑的智能合约,基于图灵完备的编程语言,从而首次将智能合约应用到区块链 [27]。2015 年 Linux 基金会主导发起了名为 Hyperledger 的开源项目 [28],旨在推动区块链跨行业应用。Hyperledger 项目旗下有众多区块链平台,如 IBM 提供的 Fabric、Intel 提供的 Sawtooth 与其他来源的 Burrow、Iroha 等 [27]。Hyperledger Fabric^⑥是一个带有准入机制的企业级联盟链项目,目标是实现一个通用的权限区块链的底层基础框架 [29]。

区块链具有去中心化、数据难以伪造与篡改、时序性、分布式、高冗余存储、智能合约等特点 [30],不仅在金融领域 [31] 有着广泛的应用,如数字资产、支付和代币等 [32],还能涉及到其他领域,包括信息共享、鉴证证明、物流追溯、物联网、居民医疗、司法服务等 [33]。在众包应用领域,区块链也取

^①百度众测平台. <http://test.baidu.com/>

^②阿里云众包平台. <https://newjob.taobao.com/>

^③Google Test Cloud. <https://google.github.io/googletest/>

^④Amazon Mechanical Turk. <https://www.mturk.com/>

^⑤Applause. <https://www.applause.com/>

^⑥Hyperledger Fabric. <https://www.hyperledger.org/use/fabric>

得了一些技术进展。由于当前众包领域的中心化所导致的安全问题与数据可信性问题，去中心化的区块链技术与之结合能够提供良好的解决方案。Li [23] 等人提出的 CrowdBC 可用于众包的基于区块链的去中心化框架概念化，在该框架中可由大量的工人解决请求者的任务，而无需依赖任何第三方信任的机构，可以确保用户的隐私并且仅需低廉的交易费用需要。Zhu [34] 等人也利用结合区块链和众包的优势，提出了一个创新的混合区块链众包平台 zkCrowd，集成了混合区块链结构、智能合约、双重分类账和双重共识协议，可确保通信安全，验证交易并保护隐私。Lin [35] 等人研究了已有的具有激励机制的区块链众包系统 BCS 的安全性和隐私要求，并提出了基于 JUICE 的原型实现的 SecBCS 系统。Zhang [36] 等人定义基于区块链的众包方案的基本要求，包括公平性、保密性和完整性，然后使用哈希加密算法，提出了一种基于区块链的安全公平的众包平台 BFC。对于基于区块链的激励问题，Kadadha [37] 等人提出了一种基于拍卖法的区块链全分布式众包框架 ABCrowd，结合拍卖法，完全在以太坊区块链上运行众包平台，通过智能合约执行来确保可信的分配执行。然而，上述区块链与众包相结合的系统中仅仅将区块链用于众包的数据存储，对于众包测试领域而言，一方面是场景与众包拍卖的场景有较大差别 [38]，并不能完全适用；另一方面是已有系统均并未实现当前众包测试中所需要的激励可信化，无法保证激励流程的公正透明。

1.3 本文的主要工作

本文设计了一种基于区块链的可信众测激励方法，并以此实现了面向可信众测激励的区块链系统，主要工作如下：

在方法上，针对当前众测平台在激励中普遍存在的可信性与有效性问题，本文设计了一种基于区块链的可信众测激励方法，通过结合反向拍卖模型设计的可信评分在线众测激励机制来实现有效性，通过与区块链结合的存储方式来实现可信性，并根据该方案设计了对应的可信众测激励流程。本文通过智能合约将众测流程数据实时接入区块链平台作为分配依据，当执行奖励分配时，先通过溯源从账本获取依据数据，再由自动化通过在线评分激励机制完成激励并将结果存入区块链；当执行奖励查询时，则直接通过账本获取激励结果数据。

在系统上，本文根据上述可信众测激励方法，实现了面向可信众测激励的区块链系统。本文采用易于扩展的 BaaS 架构，共分为三层。首先是区块链基

础平台，分为服务层和基础设施层。服务层基于基础设施层封装对外提供的接口，包含数据上链、数据溯源和资产确权三大功能；基础设施层采用了联盟链 Hyperledger Fabric 框架，实现智能合约、CA 认证等功能。其次是可信众测激励服务端，主要分为两大类服务：用户服务和众测任务服务。用户服务包括用户登录与信息，而众测任务服务包含任务需求服务、测试报告服务、最终报告服务以及积分奖励服务。最后是系统客户端，包含前端展示页面即三方用户的系统界面，除此以外还提供了 Fabric Explorer 的区块链浏览器。

在实验上，本文对系统进行功能、性能与安全测试。在功能方面，系统实现了用例中的众测数据存储查询、众测激励等功能；在性能方面，系统通过 JMeter 与 Caliper 等工具的测试，吞吐量达到 250tps，可支持真实的生产场景；在安全方面，系统通过 Scanner 工具的测试，区块链中智能合约经过静态分析不存在常见安全漏洞。系统同时接入众测月度赛真实数据进行案例分析，经检验激励结果无误且公开可信，在高并发下仍能提供稳定的服务。

1.4 本文的组织结构

本文详细介绍了面向可信众测激励的区块链服务系统，其组织结构如下：

第一章，引言。概述本系统的研究背景与意义，对当前国内外在众测激励与区块链技术上的研究现状进行分析，最后对本文主要工作进行阐述。

第二章，技术综述。结合研究内容，具体说明了相关的众测激励与区块链技术概念，先后分析了众测流程与激励机制、区块链的特性与结构。

第三章，众测激励流程的设计。分析当前众测激励主要问题，随后结合现有方案与研究成果进行讨论，设计可信众测激励流程。

第四章，众测激励系统的需求分析与总体设计。先分析需求并给出用例图，再给出架构图与 4+1 视图详细描述总体设计，最后定义账本持久化模型。

第五章，众测激励系统的详细设计与实现。首先介绍服务端众测任务、任务需求、测试报告、最终报告、积分奖励模块的实现，然后介绍区块链基础平台的服务层实现与设施层部署，最后展示系统界面。

第六章，众测激励系统的测试与案例分析。先测试功能、性能与安全性并分析结果，最后接入真实比赛数据模拟众测案例并分析结果。

第七章，总结与展望。首先概括本文当前研究内容与成果，随后针对本文当前有待改进之处，讨论并给出后续修改的方向。

第二章 技术综述

本章对系统涉及的相关概念和技术进行简要概述。本系统主要研究了区块链技术与众测激励技术并将其相结合以实现可信众测激励，下文将详细介绍相关的概念和技术，主要包括众测激励技术和区块链技术。

2.1 众测激励技术

2.1.1 众包测试概述

2006 年，Howe [39] 首次提出了众包的概念，即“公司承担员工的任务并将其以公开招募的形式外包给人们的行为”。众包测试正是一种众包技术在软件测试领域应用的的活动，交由数量众多且来源广泛的工作人员来进行测试，其中既包含专业人员与新人，也包含真实用户甚至相关专家 [40]，在传统测试基础上做到了改进，因此实现了高度并行化并显着提高了测试效率。然而，众包测试并不是传统测试的替代品，而是一种为传统测试提供的补充性质的解决方案。

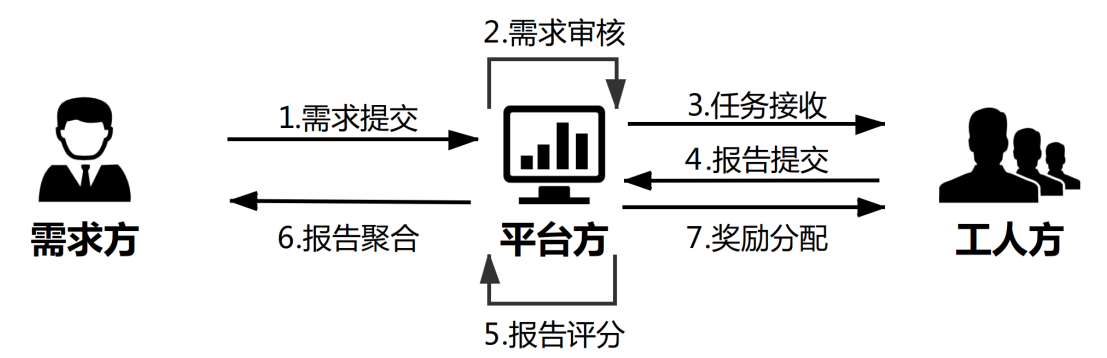


图 2-1: 众包测试流程图

以国内主流的众测平台慕测平台为参考，下文将介绍众包测试的基本流程。如图 2-1所示，众包测试的参与方主要分为为需求方 (Requester)、工人方 (Worker) 和平台方 (Platform) 三方 [41]：需求方是众测任务的发起者，借助平台

来获取对待测软件的专业性测试；工人方是服务于软件测试行业且具备较强专业性的个人或群体；平台方为需求方和工人方之间建立起联系，是众测任务的推进与助力者，需求方向其提出需求并收获专业测试结果，工人方通过其接收任务并获得劳动报酬。

众包测试的流程主要包含以下七个阶段：

1) 需求提交：需求方通过平台发布测试任务，将待测软件和测试任务提交至众测平台，并提供该次任务的奖励；

2) 需求审核：由平台方对需求方提交的任务需求进行评估审核，通过后正式发布任务，未通过则驳回需求并返还奖励；

3) 任务接收：任务发布后，由工人方自行选择接收自己感兴趣的任务；

4) 报告提交：工人方下载需求和对应软件后，根据文档内容进行测试，完成后按格式填写并提交测试报告，由若干份缺陷报告组成一份测试报告；

5) 报告评分：平台方根据任务需求对工人方提交的测试报告中每份缺陷报告进行评估审核；

6) 报告聚合：在完成所有缺陷报告的审核后，平台方需要选取部分优质缺陷报告，聚合其后形成最终报告，并将其交付给需求方；

7) 奖励分配：任务完成后，平台将需求方预设的奖励分配给每位参与该任务的工人。

测试报告名称# XXX-APP测试报告	
创建日期	2021-01-20 18:51
设备品牌	xxx
设备名称	xxx
操作系统	windows 10
测试用例列表 为了评分准确,请勿提交重复测试用例	
用例1	发布截图
所有Bug#缺陷列表 为了评分准确,请勿提交重复Bug	
Bug题目	无法发布截图
复现程度	小概率复现
严重程度	一般
缺陷分类	功能不完善

图 2-2: 众包测试报告实例图

众包测试报告的具体内容如图 2-2所示，与传统的软件测试报告基本相同，包含的字段主要有测试报告名称、创建时间、测试工人、测试环境、测试用例列表、缺陷报告列表、复现程度、严重程度、缺陷分类等。缺陷报告中对每个缺陷需提供截图以便后期调试。

2.1.2 众测激励

在众测流程中，需求方是众测任务的发起者，而测试工人们则是平台生产力的来源，是任务的执行者。一个健康的平台必须建立在高质量与数量参与者的基础上才能得到发展。若没有足够数量与质量的工人加入，在任务发布后可能面临着缺少工人劳动力或是尽管数量足够但参与度普遍不高的情况，进而导致任务停滞或报告质量参差不齐，造成各参与方的不满，最终形成用户流失的局面。因此，为了提高工人的工作质量，鼓励其更认真地完成任务，也为了提高其参与的积极性，鼓励更多工人来加入从而扩大众测的规模，平台会对工人方进行激励。然而，工人方作为被激励者，对于自己劳动的付出自然希望能够获取等价的报酬。为了确保激励过程的公平性与有效性，众测平台往往会对影响激励的关键因素进行研究，如激励形式与激励机制等。

激励形式可划分为内部激励与外部激励两种类型 [42]。外部激励是目前使用更多的方式，是指由外酬所触发的、与工作任务内容并无直接关系的激励。外酬是指当任务完成后或在工作场所以外工人所获得的满足感，它与工作任务并不是同步的。外部激励的方式有多种，包括金钱激励、娱乐性激励、虚拟激励等。金钱激励 [43] 是最常用的方式，以金钱补偿的形式回报工人参与众包测试活动。而内部激励是任务本身的激励作用与完成任务所额外产生的激励作用累加的效果，即工人本身的兴趣爱好或是成就等对其行为产生的影响，与外部激励有本质区别。如果任务本身能发挥工人的特长，或是其对该工作充满兴趣，那么任务本身就是一种激励，且能使工人维持长久的高积极性。

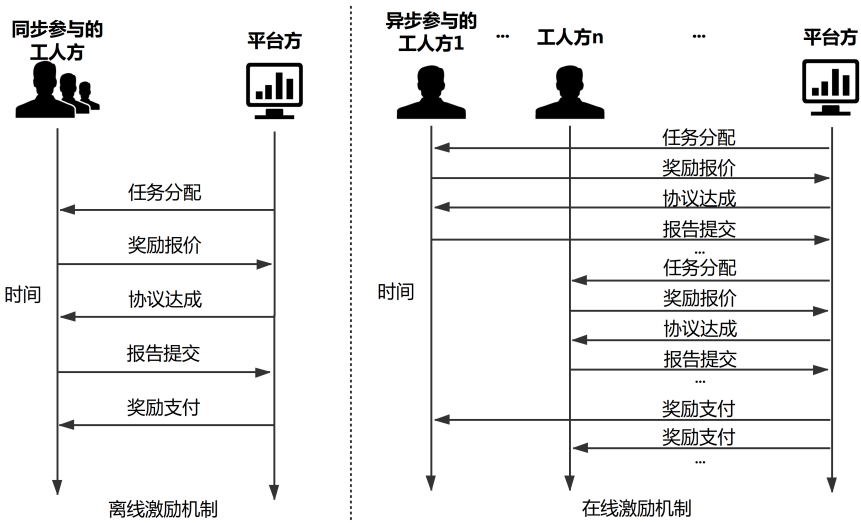


图 2-3: 离线与在线激励机制对比图

传统的激励机制包括离线和在线两种类型，图 2-3 给出两者的对比。离线激励机制中，在任务开始之前利用拍卖等手段分配所有任务，任务结束以后再支付报酬。离线的前提是所有用户都是值得信赖的，在接收任务后会积极且合格地完成任务并提交，不会中途未完成就离开 [12]。在线激励机制中，任务异步分配给不断加入的用户，用户陆续提交报告，平台根据用户的参与情况、任务完成情况、数据上传质量等因素进行实时决策，动态地调整工人方的最终所付报酬。其中，拍卖是一种解决参与者行为的有效方法，已被广泛用于设计激励机制 [44]。拍卖过程主要先由需求方发起请求，随后工人发起自己任务的定价，需求方收集定价后选择工人并发布任务，完成后支付相应的奖励 [45]。

2.2 区块链技术

2.2.1 区块链概述

区块链是一种链式结构的分布式数字账本 [46]，节点可以在网络中的公共账本上存储交易，每一笔交易在执行并公示后无法再次更改。区块链具有以下四个方面的特性 [47]：(1) 去中心化特性，指在区块链网络中，每次进行交易直接在任意两个对等节点之间进行即可，不需要经过中心机构的管理。去中心化的方法使得区块链能够降低服务器成本并提升性能，也能提高交易的效率。(2) 难以篡改特性，指在区块链网络中，发生的每一笔交易都会被记录在每个节点上，验证并记录于区块中，插入在此前区块的尾部。同时还使用分布式共识算法，因此交易数据难以伪造，且对于此前的数据进行篡改几乎是不可能的。(3) 可匿名特性，指在区块链网络中，每个用户使用的地址并不是公开的，而是通过非对称加密算法得到的密文地址，从而与其他用户进行交互，用户也可以生成多个地址来隐藏个人身份。(4) 可验证特性，指在区块链网络中，每个交易都必须通过各验证节点的验证并赋予时间戳，且链上每个区块都包含其前驱区块的加密哈希值，因此用户能够通过节点验证并且追溯以往的区块记录，从而使数据具有透明性与可验证性。

2.2.2 区块链架构

图 2-4 给出了区块链的基础架构，其按照分层模型划分为五个不同层级，下文将分别介绍各层的具体内容。

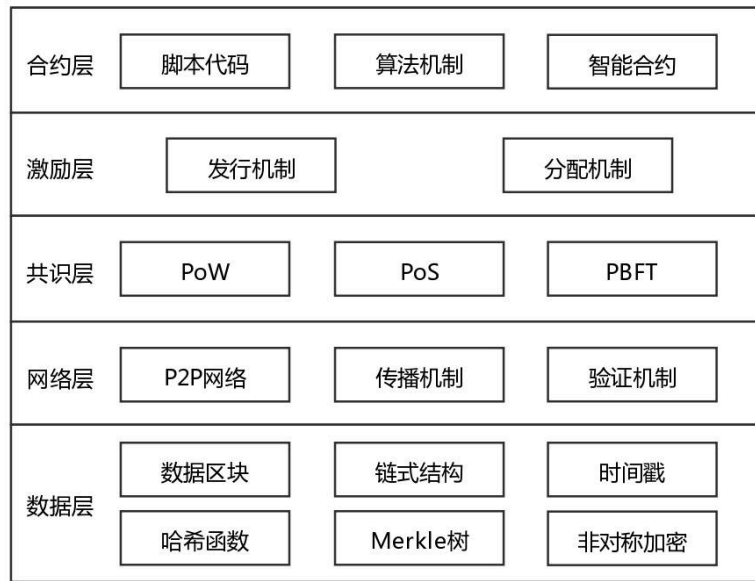


图 2-4: 区块链基础架构图

1) 数据层

数据层存放区块链所有数据信息，封装了哈希值、时间戳与公私钥等，同时通过非对称加密、哈希函数和带时间戳的 Merkle 树生成区块存储数据。其中，Merkle 树是为了快速地进行数据安全性验证，而时间戳是为了追溯数据。

2) 网络层

网络层包括 P2P、传播和验证机制。点对点网络不需要中心化服务器来操控，由各节点分配资源。传播机制是指当某节点创造出新区块后，会以广播形式公告其余节点；而验证机制是指收到消息的节点会先对区块进行验证，再去创造新的区块。网络中的所有节点都一直在监听网络，既会产生信息又能接收信息并验证，但是区块若想要被追加到链中必须通过大部分节点认可。

3) 共识层

共识层主要封装了网络节点的各类共识算法，后者有助于实现共享数据在分布式节点间的一致性和容错性 [48]。工作量证明 (PoW) 是比特币使用的一种共识策略 [49]。矿工是指计算哈希值的节点，挖矿是指 PoW 过程。PoW 要求每个节点都计算区块头的哈希值，当某个节点达到目标值时会广播到其余节点，同时其余节点需相互验证结果是否无误。当通过验证后，其余矿工会将该区块添加上链。权益证明 (PoS) 是 PoW 的能源改进版 [50]。需要先确定各个矿工的货币所有权，其最高者可参与共识。然后不同于挖矿过程，该方法采用随机加入股权者的方式，通过交易将初始股份分散到用户手中，并以利息的形式

新增货币，实现对节点的奖励 [51]。拜占庭一致性协议 (PBFT) 是一种容忍拜占庭故障的复制算法 [52]。每个新区块将在每轮共识中确定，并遵循规则选择节点进行负责。共识分为预准备、准备和提交阶段，若节点获得的投票比率超过三分之二，则进入下一阶段。

表 2-1: 共识机制 PoW、PoS 与 PBFT 对比

	PoW	PoS	PBFT
权限管理	开放	开放	授权
容错性	25% 算力	51% 权益	33% 错误节点
节约能源	否	部分	完全

上述共识机制间的对比如表 2-1所示。在权限管理方面，PoW 和 PoS 中节点可以随意自由加入网络；而 PBFT 在每个回合中选择主要矿工前需要知道每人的身份。在容错方面，PoW 的采矿策略 [53] 下矿工仅需 25% 的算力即可获利；PoS 则设定了 51% 的权益为阈值；PBFT 则需要处理三分之一的故障节点。在节约能源方面，PoW 要求矿工不断进行挖矿计算因此较为浪费能源；PoS 仅要求矿工对区块头搜索目标值，空间有限因此能源消耗较小；PBFT 过程中不存在挖矿，因此几乎完全不消耗多余能源。通过上述对比分析，PBFT 机制能源消耗少，且账本不用额外的计算量来达到一致性，具有良好性能。由于本文采用的是联盟链，授权节点数量较少且不存在失控现象，进而容错方面并无强烈要求，因此使用 PBFT 共识机制。

4) 激励层

激励层主要包括经济激励的发行机制和分配机制，为区块链挖矿提供奖励，通过部分数字资产来激励矿工去验证交易信息，从而维护挖矿活动以及账本更新持续进行。此处的激励层与前文的众测激励并不是同一概念。区块链中激励层的主要目的是为了鼓励矿工去进行挖矿等活动，为区块链的持续发展而服务，并且仅在公有链中才采用激励层，在联盟链与私有链中一般不会设置该层；而众测激励的主要目的是鼓励工人方积极参与任务并提高报告质量，为众测的持续发展而服务。由于本文采用的是联盟链，因此并未设计激励层，在第三章中单独设计了众测激励方法。

5) 合约层

合约层涵盖了智能合约与机制、算法等，是区块链可编程的基础。其中，智能合约可理解为一种区块链中自验证、自执行和状态自响应的协议，包括一

组可执行函数和状态变量等，状态变量随着函数的执行而进行相应改变。在区块链中所有用户均能对其请求交易以调用函数，而智能合约也会在各节点执行 [54]。智能合约的运行原理为，当一组团体同意一组预定义的协议，即可将其编成智能合并进行签名加密，随后公告至其余节点来检查。账本会组装验证过的合约，若触发了前提条件，这些协议无需人工干预，将会被激活并自动执行。本系统采用智能合约实现众测数据上链与查询以及奖励分配等功能，在执行功能时实时触发智能合约交易，经过节点共识后完成账本的数据更新，从而保证众测激励流程的可信性。

2.2.3 区块链分类

表 2-2: 公有链、联盟链与私有链对比

	公有链	联盟链	私有链
共识过程	不需许可	需许可	需许可
数据读取权限	公开	公开/受限	公开/受限
共识群体	所有节点	部分节点	某个组织
去中心化	是	部分	否
交易效率	低	高	高
数据变动性	不可变	可变	可变

区块链分为公有链、联盟链及私有链三类，表 2-2给出了区别 [27]：在共识过程方面，任意节点均可加入公有链，而节点想要加入联盟链或私有链必须经过联盟或组织的许可；数据读取权限方面，所有交易在公有链中对外公开，但需由联盟或组织决定联盟链和私有链信息的是否公开；在共识群体方面，公有链允许所有节点参与，联盟链仅预选的一部分节点能够参与，而私有链则完全由单个控制的组织来决定共识；在去中心化方面，完全实现去中心化的是公有链，部分实现的是联盟链，而完全中心化的是由组织控制的私有链；在交易效率方面，公有链由于节点数量众多导致传播时间长，由于安全性限制导致响应时间久且交易吞吐量不足，因此效率低于其余两类；在数据变动性方面，由于各分布式节点中均存有交易数据，因此公有链基本难以篡改数据，而由于是联盟或组织存储数据，联盟链或私有链存在篡改的可能。本文中由于各参与方整体联合协作，类似于联盟组织的形式，并且众测数据为数字资产不能对外公开，满足联盟链的场景，因此本文联合三方搭建联盟链以实现可信众测激励。

2.2.4 Hyperledger Fabric

2015 年由 Linux 基金会牵头、IBM 和 Digital Assets 开发了开源联盟链平台 Hyperledger Fabric [55]。Fabric 使用智能合约方案，采用模块化架构，可管理节点成员权限，其共识机制可进行配置。图 2-5展示了 Fabric 的工作流程。Fabric 由组织组成网络并维护节点，其分布式账本协议由节点运行，使用 Docker Compose 构建。开发者可通过官方提供的 Fabric-Java-SDK 来管理网络。

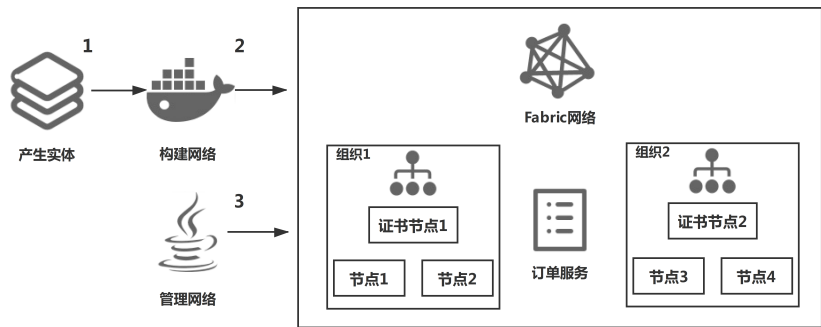


图 2-5: Fabric 工作流程图

Fabric 架构由区块链、会员和链码三大服务组成 [29]。Fabric 的架构核心是区块链服务。其中，交易执行时由智能合约即链码执行，通过分布式账本存储信息、执行交易并变更状态；共识算法接口由共识管理部分提供，其支持 PBFT 算法。会员服务可注册用户，并支持验证身份。用户权限按照角色来分配，Fabric CA 负责身份证书部分，颁发并管理用户的身份与交易的证书。用户需要通过受信任的成员服务提供者 (MSP) 进行注册，经过注册机构颁发有效身份证书、交易机构颁发交易证书后，用户才能够在网络上发送交易。Fabric 的智能合约链码 (Chaincode)，其在区块链节点上部署。其节点可划分为两种：一种是维护账本、负责共识与验证事务的验证节点；另一种是连接到验证节点作为代理将客户端、不执行事务的非验证节点。

2.3 本章小结

本章为系统实现提供理论依据与技术支持。首先介绍了众包测试流程，接着介绍众测激励的目的与方法。最后介绍了区块链技术，包括区块链基础架构、分类及联盟链 Fabric。

第三章 可信众测激励方法

本章概述系统的可信众测激励方法。首先，本章介绍当前众测激励的问题。其次，结合反向拍卖模型设计了可信评分在线众测激励机制以解决有效性问题，采用与区块链结合的方式以解决可信性问题。最后，通过建模方式详细阐述了可信众测激励方法的完整流程。

3.1 引言

目前，在众包测试的流程中，众测平台常常会对参与的工人方进行激励，原因主要有以下两点：一方面为了吸引更多工人来参与及报名，保证有足够数量的用户参与任务；另一方面为了督促工人更好地完成任务，提高其测试的质量。当前本文激励的对象仅为工人方，由于其激励结果与其权益直接挂钩，只有保障了工人方的利益，才能够使得激励有效，切实鼓励与吸引到工人，进而促进众测的良性发展。然而，现有众包测试平台在激励过程中存在着两大问题：有效性不足与可信性不足。

在有效性方面，在当前常见的众包平台中，对劳动力评估与对奖励分配的方式均部分存在着不科学与不公平的问题，容易出现诸如工人逃单骗取激励等恶性现象，进而导致众包平台的激励机制缺失公信力。目前的众包平台采用的各类激励机制各有优劣，需要对众多激励机制进行比较与选取，结合其长处和特点，设计出一种适用于众包测试的激励机制，以尽可能地使众测激励达到预期的效果。

在可信性方面，由于当前激励的依据是平台方存储的众测数据记录，平台作为第三方使用传统数据库中心化地存储和管理数据，一方面中心化节点可能会受到单点攻击等造成数据故障，另一方面对平台数据本身的真实性存疑，平台方数据权限过大，在一定程度上存在数据欺诈的可能。众测平台需要保障需求方的奖励开销完全分配到每个工人手中而没有被平台所贪污，同时保证每个工人获得的奖励确实是按照劳动力所分配的，因此必须通过公开透明的流程保证数据的真实可靠性。

3.2 可信众测激励设计

针对可信众测激励的方法设计，我们需要从激励有效性方面入手，设计激励机制后，结合当前的可信性问题的解决方案，进而设计具体详细的方法。

3.2.1 激励有效性设计

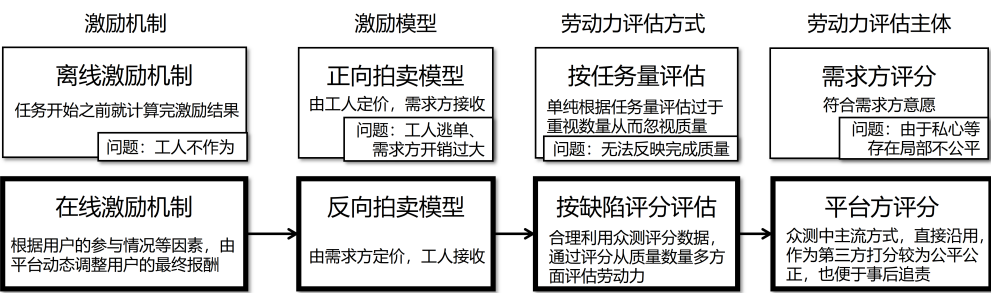


图 3-1: 激励有效性设计路线图

在激励过程中影响其结果评价的因素众多 [19]，包括激励机制、缺陷等级、任务优先级、评分标准等。由于本文研究重点为可信众测激励，在有效性方面，为简化过程不做深入讨论，仅着重于激励机制等有效性方面，并不考虑其余因素。本节激励有效性设计的总体路线如图 3-1所示，下文将分别从图中激励机制、激励模型、劳动力评估方式、劳动力评估主体四个方面按顺序阐述。

激励机制：传统激励机制包括离线设置和在线设置两种类型，两者各有利弊。在离线激励机制中，在所有工人上传竞标即接收任务之后，就确定了拍卖的赢家 [12]。该方案假定所有工人从任务一开始的分配时就出现，随后不能接受新的任务投标，并且默认工人一定会完成该任务。显然，当工人在完成测试报告的过程中出现了未提交的情况，该机制仍会对其进行奖励，因而造成需求方的开销严重损失；又或者是工人接收任务后后期需要放弃却不能接受新的任务，被迫死锁在该任务中。因此，离线机制并不适用于众包测试。在线激励机制根据用户的完成情况动态调整最终报酬，能够结合质量进行合理的评价，因此相对更适合众测。本文选取在线激励机制作为可信众测的激励机制。

激励模型：在线激励机制可结合激励模型作进一步的设计，常见的为拍卖模型 [14]。众包中的拍卖模型与真实拍卖场景并不完全一致，由于买家与卖家并非一对多而是多对多的关系，因此买家能同时报价多个商品并成交，卖家也

能够一次拍卖有多个买家。拍卖模型中，由工人方作为卖家最先开始提出最低酬劳的报价，需求方作为买家听取价格选择工人，随后根据每个工人完成任务的表现累加奖励，最后支付奖励。这种在线激励的方式较好地解决了离线机制中工人不作为的问题，根据任务完成的情况相应的调整奖励，使激励达到了相对公平，然而使用了拍卖法在用户数量上和工人效率上有着不足。拍卖模型定价以工人中心，确实较为适合在移动众包等大量需要用户且专业性要求不高的场景 [16]；然而在众包测试场景下，工人需要一定专业性且对于其参与的数量要求相对较小，并且对需求方可能会导致开销成本过高。针对该问题，反向拍卖模型是一种不错的解决思路：由需求方作为买家来最先提出最高报酬的定价，随后部分工人作为卖家接受，接着按照任务完成表现进行动态调整奖励。反向拍卖法解决了此前拍卖法中工人逃单与需求方开销过大的两大问题 [17]，很好地控制了成本并平衡各方权益，适配于众包测试。由于本文主要研究众测激励的有效性，激励机制的具体实现并非研究重点，为简化报价过程，本文结合上述反向拍卖模型，在此基础上进行了精简，由需求方在任务开始直接提供任务总报价，由工人自助进行选择，可即时查看任务的人均报酬定价，任务完成后根据表现进行奖励动态调整。

劳动力评估方式：在线激励机制中，任务完成后对奖励的动态调整分配直接决定工人最终报酬，因此需要对每个工人进行公正公平的绩效考核。当前常用的劳动力绩效评估方式包括按任务量评估与按缺陷评估的两种方式。按任务量评估的方法过于重视数量而忽视完成质量 [19]，一方面导致无法做到多维度的评估进而影响工人权益分配，另一方面会恶性引导工人，甚至出现刷单不作为的行为。按缺陷评估的方式显然更加合理，不仅能够有效从质量数量多维度来评估工人的任务完成绩效，还能督促工人保质保量地参与众测，进而提高众测本身的质量。缺陷评估法可以直接使用现有众测缺陷评分数据，合理利用数据进行二次加工评估从而量化劳动力，无需改动现有众测流程即可实现精准绩效评估。

劳动力评估主体：缺陷评分评估机制根据主体分为由平台方打分与由需求方打分两种方式。本文考虑到若交由需求方打分的情况下，可能会由需求方本身的主观因素影响如偏见等，导致打分不公正，进而影响到工人的积极性。因此，由第三方平台来评分是较为合适的，平台独立于各方，评分的结果相对中立；平台审核也有据可查，便于时候追责；同时这也是当前众测中主流的评分方式，无需改动众测流程，直接沿用即可。因此本文采用以平台为中心、结合

反向拍卖模型评分的在线激励机制。在需求方提供总奖励报价并由工人自主接受完成任务后，由平台对每位工人的报告进行评分，随后参照 [20] 中的方法，计算工人报告得分在总分中的占比即贡献比例，按照比例获取相应奖励。

3.2.2 激励可信性设计

在选定激励机制后，对于众包测试的激励有效性问题能够得到解决，然而仍存在激励可信性不足的问题。因此，我们需要对于当前的众测激励机制继续进行改进，以满足奖励分配的透明与可信。从用户的角度出发，奖励分配的全过程理应公开并完整记录形成一条透明的溯源链路，便于各参与方进行查阅与校验。工人方能够查看到相互的劳动力评估与对应的结果，通过公开的流程与数据避免争议；需求方能够查看到自己支付的开销报酬与其具体的最终分配结果，确保奖励切实落入工人手中。而从工程的角度出发，奖励分配的过程需要由平台自动化完成，且确保分配的内容准确无误。

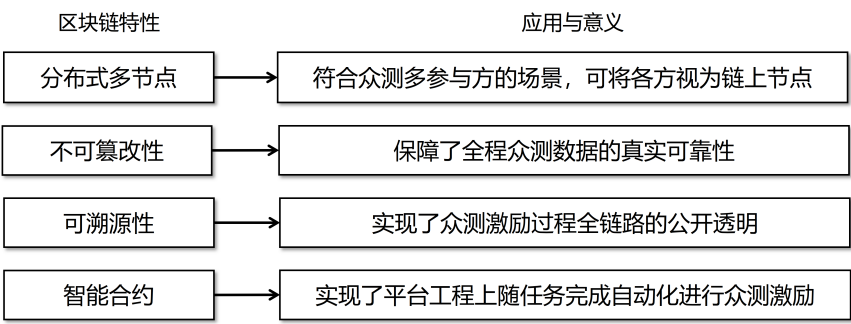


图 3-2: 激励可信性设计路线图

针对可信性问题，第一章提到区块链与众包相结合的方法 [37] 值得我们借鉴。如图 3-2所示，在两者的结合下，我们可以利用区块链的特性很好地解决以上的问题。区块链中分布式多节点参与的特性与当前众测多个参与方的场景相符合，可将参与方视为链上节点，通过在线评分激励机制进行劳动力公正评估与分配，解决了分配方式不公平以致激励效果不佳的问题；通过区块链的去中心化特性与不可篡改性进行分布式可靠存储，解决了第三方数据真实性存疑的问题；通过区块链的溯源特性进行贡献资产的全链路确权，解决了激励过程无法做到公开透明的问题；通过区块链的智能合约准确完成自动化奖励分配，解决了平台工程上自主激励带来的不便性的问题。因此，本文最终的方案即为基于区块链，采用结合反向拍卖模型评分的可信在线众测激励机制，从而实现众包测试的可信众测激励，该方法的具体内容在下一小节将详细介绍。

3.3 可信众测激励方法

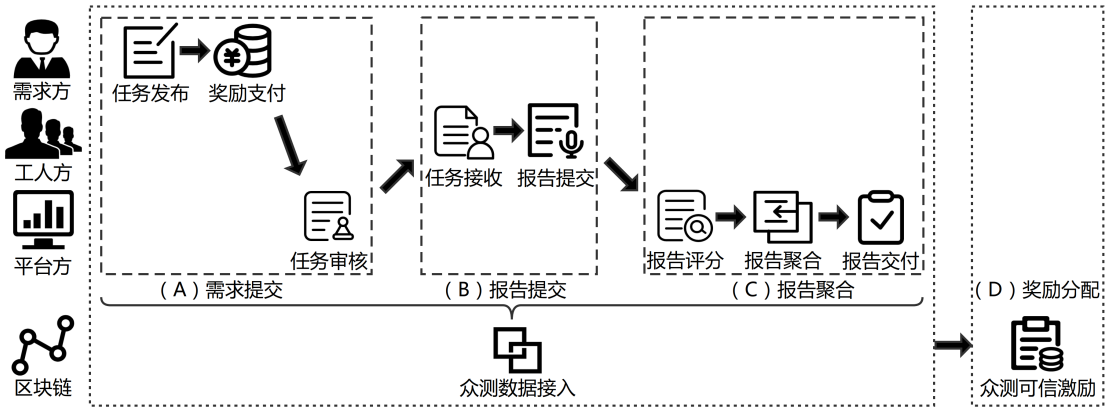


图 3-3: 可信众测激励流程图

根据第二章的介绍，在众包测试中一次完整的流程分为需求提交、需求审核、任务接收、报告提交、报告评分、报告聚合和奖励分配七个详细步骤。为简化过程，下文将其合并为需求提交、报告提交、报告聚合和奖励分配四个阶段。本文将这四个阶段与基于区块链的在线评分激励机制相结合，设计的激励方法流程如图 3-3所示。前三阶段可统称为众测数据接入阶段，根据在线激励机制的要求，需要先将上述众测数据进行接入后再实现激励，又由于基于区块链，因此先将数据接入区块链平台，再进入由其发起的众测可信激励阶段，即奖励分配阶段。下面将通过建模的方式详细介绍可信众测激励方法流程。

在一次完整的众包测试过程中，虽然在整个平台内需求方与众测工人的用户均有多人，但是一个任务中仅仅包含一位需求方与多名众测工人，因此参与者主要分为需求方（记为 R ），工人方（分别记为 $W_1, W_2, W_3, \dots, W_n$ ，这里 n 表示工人总数），以及平台方（记为 P ）。

3.3.1 需求提交

需求提交阶段由需求方与平台方共同参与，流程如图 3-4所示。在此阶段中，需求方先发布需求任务，然后对该任务支付相应的奖励；最后平台方对该需求进行审核，通过后才能发布任务，未通过则驳回需求并退回奖励，待需求方再次提交。需求提交与审核、奖励的数据都将存入区块链中作为凭证。

任务发布：一项任务由需求方进行发起，某位需求方 R 通过众测平台 P 发起一项测试任务 T ，并且提交任务相关信息（如需求文档等）与待测软件。

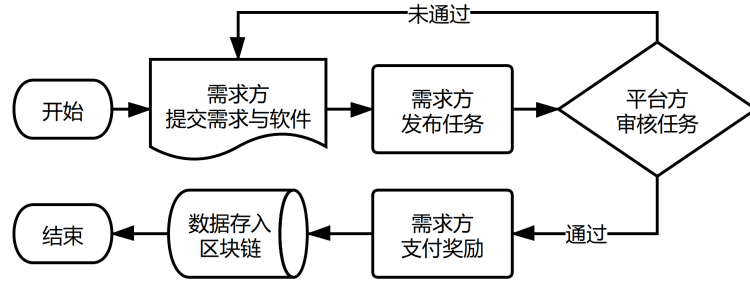


图 3-4: 需求提交流程图

奖励支付：由于本文结合反向拍卖模型，由需求方进行定价，在任务发布之后，需求方 R 需要对该任务支付对应的奖励报酬 S_R 。在本文中，由于系统暂时不涉及货币的交易，同时使用的技术联盟链不存在代币机制，因此采用积分的形式，代替实际场景中的交易与奖励，来实现众包测试的流程。 R 支付的积分 S_R 直接交由平台 P 暂时保管，此时任务已形成，并作为将发布阶段的所有基础信息包含积分 S_R 同时记录在区块链上以保证数据真实性。

任务审核：平台方 P 对于该任务提交的任务 T 与相应需求文档、待测软件等进行评估审核。如任务中需求内容与待测软件一致且无误后，则说明任务正常，审核通过，则 T 将通过 P 发布，由 W 选择接收与否。若未通过审核，则会回退到前一阶段，由 R 重新进行任务的提交，此前支付的奖励 S_T 也会相应退回。该阶段中任务需求的审核情况信息会存入区块链中。

3.3.2 报告提交

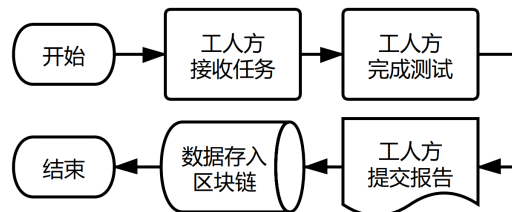


图 3-5: 报告提交流程图

报告提交阶段由工人方参与，流程如图 3-5 所示。在此阶段中，工人们需要在平台中先接收该测试任务，下载需求文档与待测软件，在该任务按要求完成测试后，填写并提交测试报告。报告提交的数据都将存入区块链。

任务接收：需求方 R 发布任务 T 后，由工人方 W 自主进行选择接收自己感兴趣的任務，假设本次任务最终有 n 名工人接收，工人的接收阶段的信息同

样会以区块的形式上链存储。由于本文采用的是在线激励机制而非离线机制，因此激励并不在任务接收后立即执行，而是需要等到任务完成后才能进行，根据工人实际完成的质量情况来决定激励的程度。

报告提交：测试工人 W_1, \dots, W_n 分别根据任务进行对应测试，完成后每人提交一份测试报告 TR ，每份 TR 中包含多份缺陷报告 BR ，该阶段提交的每个缺陷报告的信息都会存入区块链中。

3.3.3 报告聚合

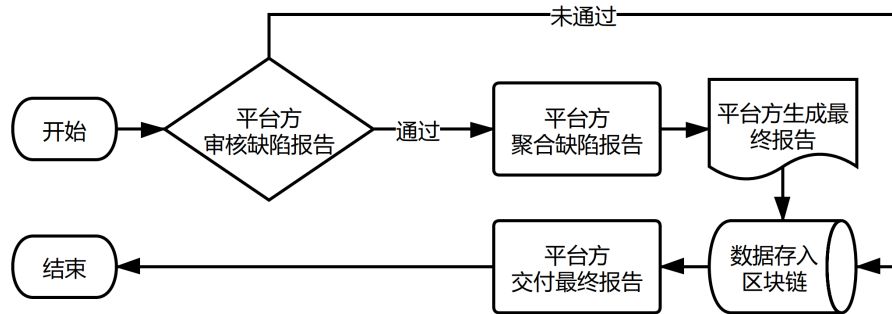


图 3-6: 报告聚合流程图

报告聚合阶段由平台方与需求方参与，流程如图 3-6 所示。在此阶段中，平台方需要先对该任务的所有缺陷报告审核评分，然后在所有缺陷报告中选取部分，并进行聚合后形成最终报告，最后将该最终报告交付给需求方。报告评分以及聚合的数据都将存入区块链中作为凭证。

报告评分：平台方 P 对于该任务的所有提交的缺陷报告 BR_1, BR_2, BR_3, \dots 进行评估审核。首先这里的粒度是到缺陷报告而非测试报告，其次由于本文采用的是基于评分的在线激励机制，因此这里的评估方式采用评分法，对于每份缺陷报告由平台进行打分。每份缺陷报告的得分分别记为 $S_{BR_1}, S_{BR_2}, S_{BR_3}, \dots$ 。该阶段中每份缺陷报告的评分情况信息都会存入区块链中。

报告聚合：在所有缺陷报告 BR 都完成评分后，平台将在所有 BR_1, BR_2, BR_3, \dots 中选取部分典型的报告聚合成最终报告 FR 。这里聚合所选取的依据以及算法并非本文研究重点，因此本文并不考虑，将使用慕测现有算法进行报告聚合。该阶段中每份缺陷报告的聚合情况信息与最终报告的信息都会存入区块链中。

报告交付：在缺陷报告聚合完成后，平台需要将最终报告 FR 交付给需求方 R ，一次众包测试的任务基本完成。

3.3.4 奖励分配

奖励分配阶段由平台方与工人方参与，流程如图 3-7所示。由于本文采用的是在线激励机制，因此奖励分配将在任务完成后进行。在此阶段中，平台方需要依据此前区块链中的记录对工人进行积分激励。由于按缺陷评分评估绩效，首先平台调用智能合约获取此前区块链中存储的可信众测缺陷评分数据，其次合约脚本根据数据自动化进行工人的任务最终积分结果的计算与分配，最后将积分奖励的分配过程与结果数据都将存入区块链中作为凭证。由于本文激励方法基于区块链，因此这里所有涉及计算的数据均来源于区块链平台，激励依据正是链上可信数据，通过数据真实性以确保激励的公平可信。

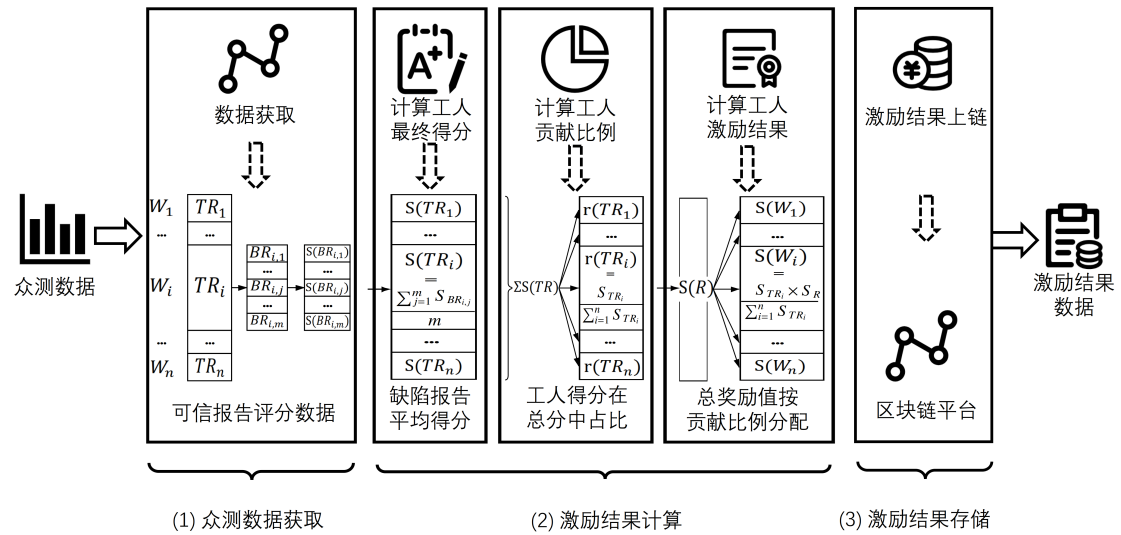


图 3-7: 奖励分配流程图

积分激励：在本次众测任务完成后，平台方 P 需要将需求方 R 预设的奖励 S_T 分配给 W_1, \dots, W_n 。在获取到各位众测工人在该任务下所有缺陷报告得分数据后，需要设计激励算法进行分配。由于此处的激励算法并非本文研究重点，为简化过程，此处参考前文提到的基于评分的在线激励机制 [20]，借鉴了已有研究中较为简便的计算方法作为本文奖励分配算法，本文需要对每次测试任务按照每位工人在其中的贡献程度进行激励，这里贡献程度就是该工人提交的测试报告的评分在所有报告的累计总分中的占比。然而因为此前的慕测的评分粒度是精确到缺陷报告，因此我们需要做的首先是计算出每份测试报告的最终得分，然后在计算贡献度。积分分配详细内容可参照算法 3.1，具体分为三个步骤：计算测试报告最终得分、计算工人任务贡献度与计算工人积分。

算法 3.1 可信众测激励积分分配算法

输入: n //工人总数
 S //各工人报告得分集合
 r //任务奖励总积分
 输出: R //各工人积分奖励集合
 1: **for** $i = 1$ to n **do**
 2: $S_i =$ 为所有缺陷报告的平均值
 3: $sum = sum + S_i$ //记录所有报告累加得分
 4: **end for**
 5: **for** $i = 1$ to n **do**
 6: $R_i = (S_i / sum) * r$ //计算每位工人获得的积分
 7: **end for**
 8: **return** R

第一步：计算测试报告最终得分。本步骤需要对该次任务中的所有工人提交的测试报告计算最终得分，一份测试报告 TR_i 由一位工人 W_i 提交，包含多份缺陷报告 $BR_{i,1}, BR_{i,2}, BR_{i,3}, \dots$ 。由于此处缺陷报告是同粒度且同权重的，参考慕测的做法，直接对多份缺陷报告计算平均得分即为测试报告的最终得分。接下来需要对 W_i 提交的 TR_i 计算平均得分，记为 S_{TR_i} ，其中 $i \leq n$ 。为保证数据真实性，从区块链平台获取当前该 TR_i 的所有缺陷报告信息。假设该 TR_i 包含 m 份缺陷报告 $BR_{i,1}, BR_{i,2}, BR_{i,3}, \dots, BR_{i,m}$ ，其对应得分分别为 $S_{BR_{i,1}}, S_{BR_{i,2}}, S_{BR_{i,3}}, \dots, S_{BR_{i,m}}$ ，则有

$$S_{TR_i} = \frac{\sum_{j=1}^m S_{BR_{i,j}}}{m} \quad (3-1)$$

第二步：计算工人任务贡献度。参考基于评分的在线激励机制 [20]，本步骤需要对该次任务中的所有工人计算任务贡献度，即该工人 W_i 提交的测试报告 TR_i 的最终得分 S_{TR_i} 在本次任务中所有 TR 的累加总分中的占比。为保证数据真实性，从区块链平台获取当前该任务中每位工人的测试报告最终得分 $S_{TR_1}, S_{TR_2}, S_{TR_3}, \dots, S_{TR_n}$ 。因此 W_i 的贡献度为

$$\frac{S_{TR_i}}{\sum_{i=1}^n S_{TR_i}} \quad (3-2)$$

第三步：计算工人积分。由于积分是对于 R 预设的奖励 S_R 进行分配，前一步已经计算出每位工人 W_i 的任务贡献度，因此只需按照 W_i 的任务贡献度给予相应权重的积分即可。因此，工人 W_i 应得的相应积分奖励 S_{W_i} 为 S_R 中的任

务贡献值，即

$$S_{w_i} = \frac{S_{TR_i}}{\sum_{i=1}^n S_{TR_i}} \times S_R \quad (3-3)$$

这里需要说明的是，基于评分的在线激励机制 [20] 中由于总积分是一开始需求方预设的，这里按照贡献比例进行分配时，若计算直接四舍五入会导致分配结果不够精确，所有工人获得奖励累加之和与实际总奖励或多或少有细微的出入。本文考虑到一方面本身研究重点并非具体的激励机制，而是应用机制与区块链技术实现可信与有效的众测激励，重点与工程实现进而简化过程以打通整个流程；另一方面，本系统中本身采用的就是积分的奖励形式，并不涉及金钱等实际货币交易，因为并无强烈的利益纠纷关系，因此这里直接按照计算结果四舍五入较为精准地分配积分，并不纠结于完全的结果精确。在计算完 S_{w_i} 后，平台将对每位工人 w_i 的账户积分进行奖励发放，从而实现众测激励。至此，一次完整的基于区块链的可信众测激励流程正式完成。

3.4 本章小结

本章结合现有研究技术，设计了可信众测激励方法。首先，本章介绍当前众测激励的现状，指出需要当前的有效性与可信性不足的两大问题。其次，结合反向拍卖模型设计了可信评分在线众测激励机制来实现有效性，采用区块链存储的方式来实现可信性。最后，完整详细地阐述可信众测激励的方法流程。

第四章 系统需求分析与总体设计

在设计完可信众测激励流程后，为实现本系统，本章将进行总体的设计。首先，本章从整体上描述系统情况。其次，本章进行需求分析，涉及功能与非功能方面，并通过用例描述其特征。再次，本章给出了架构图与四个视图，描述了系统的整体结构。最后，本章描述了系统联盟链数据模型。

4.1 系统整体概述

为解决众测激励中存在的可信性和有效性不足的问题，本文设计了可信众测激励方法，并根据流程与需求设计了面向可信众测激励的区块链服务系统。本系统的涉众主要有需求方、工人方和平台方三方，分别接入对应的区块链节点，经由三方进行共识后将众测数据实时导入区块链作为激励证据，任务完成后依据区块链中可信数据自动化进行奖励分配，并记录结果在区块链中，查询结果时各方节点直接读取区块链账本数据即可。本系统在实现方面采用 BaaS 架构分为三层，区块链基础平台封装了区块链服务并提供对外接口，可信众测激励服务端提供众测相关服务，包括任务需求服务、测试报告服务、最终报告服务和积分奖励服务，而客户端包括系统三方展示界面以及 Farbic Explorer 区块链浏览器。

4.2 系统需求分析

4.2.1 系统涉众分析

本系统的涉众主要包括需求方、工人方和平台方，下面介绍其主要特点：

需求方：本身并无专业的测试技能、借助众测平台来解决应用软件测试需求的个体或组织。需求方为寻求专业的测试结果，求助于平台发布任务，提交需求与软件，支付相应的报酬，从而获取专业的软件测试报告来提高测试效率，辅助其验收过程。

工人方：参与平台任务并能够按要求完成测试任务、拥有相关软件测试的经验与技能的个体用户。工人方通过平台接收任务，在测试完成后上传测试报告至平台，等待审核结果。在任务完成后，获得该任务相应的报酬。同时，工人希望能明确测试报告资产归属，保证经济利益不被侵害。

平台方：负责整个测试流程管理的平台。平台方是需求方和工人方之间的桥梁，接受需求方的委托对软件进行测试与相应报酬，并将任务发布给工人方，然后通过工人提供的测试报告进行聚合，在向需求方交付完整和高质量的测试报告后，对工人进行任务的报酬奖励。

4.2.2 功能性需求分析

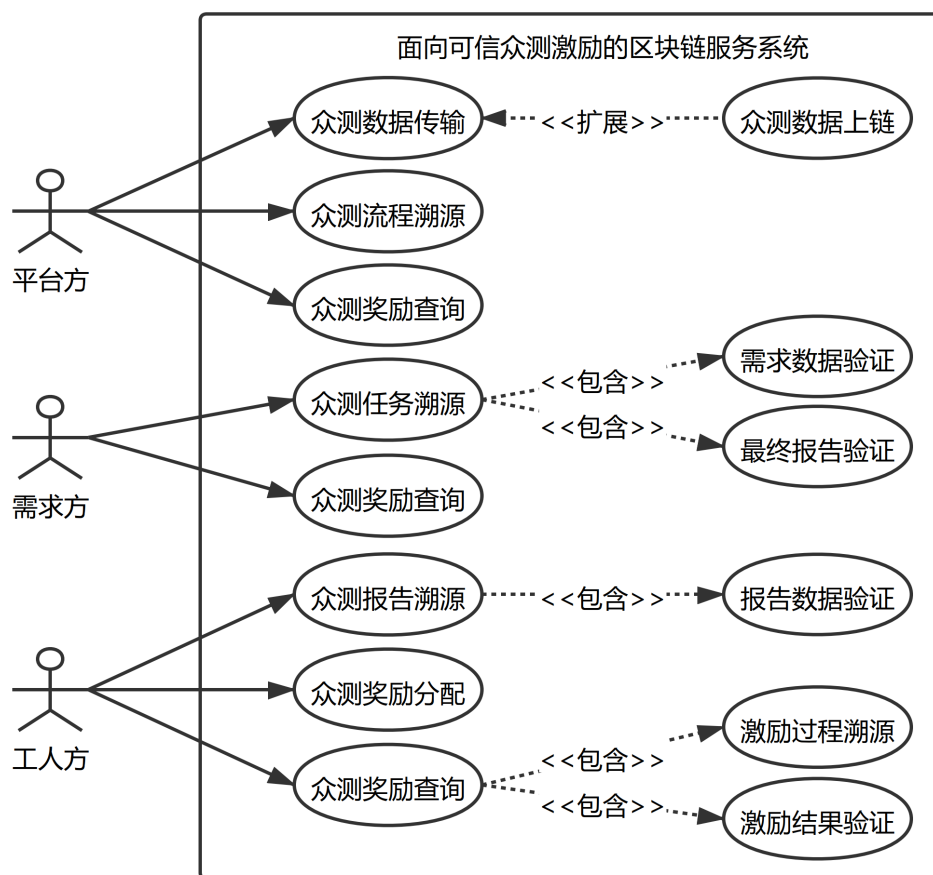


图 4-1: 系统用例图

图 4-1给出了系统各个涉众用户的相关用例情况。需求方可以通过系统对其众测任务的流程溯源，包括查看其提交的任务需求信息，任务需求资产确权，查看最终测试报告信息以及最终测试报告资产确权；也可以查询其奖励的

分配情况，包括激励过程溯源和激励结果验证。工人方能够借助系统查看其提交的报告信息并对报告流程溯源，其中包含查询平台方的对缺陷报告的审核信息以及其被聚合入最终报告与否，同时对报告进行资产确权；也可以查询其奖励分配情况，验证在该任务中自己获得奖励是否公平属实。平台方需要首先进行众测数据的接入，通过系统查看当前全部任务的具体流程进度，比如需求提交、需求审核、报告提交、报告审核、报告融合以及奖励分配阶段的详细信息；平台方同样能够查看任务激励情况。在数据验证功能中，可以通过区块链平台提供的数据验证平台方数据库数据是否与区块链中数据一致；在资产确权功能中，也可以通过区块链对资产确权生成确权报告以保障权益。下面将对系统需求进行详细的用例描述。

表 4-1: 众测数据传输用例描述

用例编号	UC1
用例功能	众测数据传输
参与者	平台方
目标描述	可信众测激励系统实时接入众测平台流程数据
前置条件	无
后置条件	各个节点共识成功后交易写入账本
优先级	高
正常流程	1. 需求方提交任务后众测激励系统接入需求与总奖励信息； 2. 平台方需求审核后众测激励系统接入需求审核信息； 3. 工人方提交测试报告后众测激励系统接入报告信息； 4. 平台方报告审核后众测激励系统接入报告评分信息； 5. 平台方将最终报告聚合后众测激励系统接入聚合信息； 6. 众测激励系统根据区块链记录奖励分配后将激励信息传送回平台。

表 4-1给出了众测数据传输用例。在众测流程中，由平台实时地将流程中数据发送给可信众测激励系统，需求方数据包括需求文档、测试软件及任务总奖励，工人方数据包括测试报告，平台方数据包括需求与测试报告的审核数据与最终报告的聚合数据。任务完成后，可信众测激励系统再按照记录完成奖励分配后将积分结果发送回平台方。

众测流程溯源用例描述如表 4-2所示。平台方能够追溯到众测任务当前的进度信息，内容包括需求信息如提交及审核数据、测试报告提交数据、缺陷报告审核数据、最终报告融合数据、奖励分配结果数据。

表 4-2: 众测流程溯源用例描述

用例编号	UC2
用例功能	众测流程溯源
参与者	平台方
目标描述	可信众测激励系统向平台展示所有众测任务当前进度
前置条件	平台方成功登录
后置条件	无
优先级	高
正常流程	1. 展开“需求提交”时间框显示任务提交数据； 2. 展开“需求审核”时间框显示任务审核数据； 3. 展开“报告提交”时间框显示报告提交数据； 4. 展开“报告审核”时间框显示报告得分数据； 5. 展开“报告聚合”时间框显示报告聚合数据； 6. 展开“奖励分配”时间框显示激励结果数据
可选流程	3.a 点击“全部”可查看该任务下所有测试报告数据

众测任务溯源用例描述如表 4-3所示，主要分为任务需求溯源和最终报告溯源。需求溯源主要包括区块链中的需求文档、待测软件、任务积分等信息，而最终报告溯源主要包括其包含的缺陷报告来源信息。对于任务需求与最终报告，均提供数据验证及区块链信息查看功能。

表 4-3: 众测任务溯源用例描述

用例编号	UC3
用例功能	众测任务溯源
参与者	需求方
目标描述	可信众测激励系统向需求方展示需求、总奖励及最终报告数据
前置条件	需求方成功登录
后置条件	无
优先级	高
正常流程	1. 需求方可查看自己提交的所有任务，包含其需求、总奖励等数据； 2. 需求方点击“报告聚合来源”按钮； 3. 可查看最终报告聚合的详细数据列表； 4. 需求方选择“数据验证”按钮； 5. 可查看链上哈希值与众测平台存储值的验证情况。

表 4-4: 众测报告溯源用例描述

用例编号	UC4
用例功能	众测报告溯源
参与者	工人方
目标描述	可信众测激励系统向工人方展示其测试报告与审核结果
前置条件	工人方成功登录
后置条件	无
优先级	高
正常流程	1. 系统展示工人方参与的所有众测任务，工人选择某个任务； 2. 可查看其提交的测试报告数据； 3. 工人方展开已完成报告审核的任务； 4. 可查看该测试报告中所有缺陷报告的得分数据； 5. 工人方选择“数据验证”按钮； 6. 可查看区块链数据与平台数据的比对结果。

表 4-4展示了众测报告溯源用例。可信众测激励系统向工人方展示了其上传的测试报告数据，若其包含的缺陷报告均已审核则会展示评分结果，包括评分、审核员等信息。系统也会展示报告的区块链信息，将区块链存储的哈希值与平台的进行比较以证明未篡改过数据。

表 4-5: 众测奖励分配用例描述

用例编号	UC5
用例功能	众测奖励分配
参与者	工人方
目标描述	工人方由系统分配获得奖励报酬
前置条件	工人方成功登录；众测任务已完成
后置条件	无
优先级	高
正常流程	1. 在报告聚合后，系统展示任务的各个缺陷报告评分； 2. 系统后台依据可信区块链数据，计算出各个工人在本次任务中应该获得的积分； 3. 系统将计算后分配的结果存入区块链平台并改动各用户相应积分。

众测奖励分配用例描述如表 4-5所示。众测奖励分配为系统在任务完成后对工人方依据可信区块链数据进行积分的分配，并将结果记录于区块链平台。

表 4-6: 众测奖励查询用例描述

用例编号	UC6
用例功能	众测奖励查询
参与者	需求方、工人方、平台方
目标描述	需求方、工人方和平台方可以在系统查看积分奖励结果
前置条件	需求方、工人方或平台方成功登录
后置条件	无
优先级	高
正常流程	1. 系统展示每项任务的基本信息，包括总积分等数据； 2. 用户点击“详情查看”； 3. 平台和需求方能看到该任务的所有积分分配情况，工人只能看到该任务自己的得分； 4. 用户点击“数据验证”； 5. 系统展示奖励分配的区块链数据验证结果。

众测奖励查询用例描述如表 4-6所示。奖励分配查询主要展示链上积分奖励分配信息，为需求方、工人方与平台方所服务。需求方和平台能够看到任务中积分的全部分配情况，而工人只能看到任务下自己的积分分配情况。用户也能够验证积分数据的真实性。

4.2.3 非功能性需求分析

鉴于本系统需要实时接入众测数据并进行区块链存储，在激励过程中也需要查询区块链以及存储结果，因此为了保证数据的可信性，性能、可扩展性与易用性是本系统着重考虑的非功能性需求。

性能：系统中用户的高质量交互体验展线在前端上，因此要降低系统响应时间，控制在 1 秒以内。服务端的响应时间也需要控制在 1 秒以内以适应前端。服务端支持的并发量应在 200 个用户以上。

可扩展性：由于系统的功能需求可能会随着业务场景的更新进而迭代变化，系统需要保证后续再次开发时便于新增模块。系统在设计上需注意高内聚低耦合的特性，保证其可扩展性和可维护性。

易用性：系统设计应从用户的角度出发，界面简洁美观、操作简单，注重人机交互体验，满足绝大多数用户的需求。

4.3 系统总体设计

本系统的各类用户及其期望的功能在前一节的需求中已列出，因此下文需根据功能完成总体的设计。首先，本节通过架构图的分层展示，清晰地反映系统的结构与各个模块的划分。然后，本节再借助“4+1 视图”即逻辑、开发、进程和物理视图，从四个不同的视角详细介绍本系统的结构。

4.3.1 系统架构

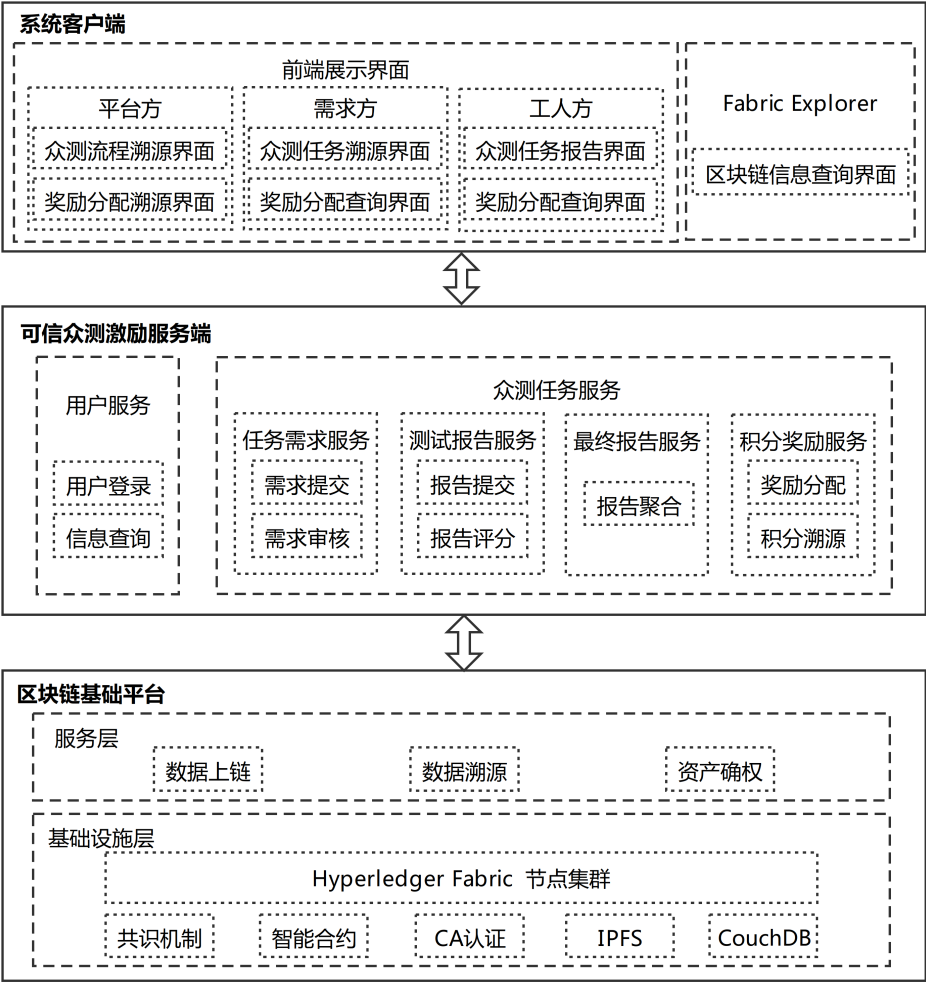


图 4-2: 系统架构图

系统采用 BaaS（Blockchain as a Service，区块链即服务）架构可抽象分为三层，如图 4-2所示，从上至下分别为系统客户端、可信众测激励服务端和区块链基础平台。

系统客户端由前端展示界面与 Fabric Explorer 共同组成。前端基于 Vue 框架进行开发, 主要提供系统与用户间的交互界面。根据用户类型分为三类, 平台方界面主要提供众测流程溯源与奖励分配溯源功能, 需求方界面主要提供众测需求溯源与奖励分配查询功能, 工人方界面主要提供众测报告溯源与奖励分配查询功能。Fabric 官方给出了区块链自带的浏览器 Explorer, 为可信众测激励系统补充了区块链详细信息的查询平台。而 Nginx 作为高性能的反向代理服务器, 实现了客户端与服务端之间的负载均衡。

可信众测激励服务端基于 SpringBoot 框架实现, 主要分为用户服务与众测任务服务两大模块。其中, 用户服务主要提供用户的登录、身份识别与个人信息的查询。众测任务服务主要包含任务需求服务、测试报告服务、最终报告服务、积分奖励服务模块。众测任务服务面向三方, 参与方通过该服务对指定任务的流程进度进行追溯与查询; 需求方通过任务需求服务完成需求数据的提交与查询, 平台方则通过其完成需求审核数据的提交; 工人方通过测试报告服务完成报告数据的提交与查询, 平台方则通过其完成报告评分数据的提交; 平台方通过最终报告服务完成报告聚合数据的提交, 需求方与工人方则通过其完成最终报告的查询; 需求方通过积分奖励服务完成任务总积分的提交, 自动分配后三方通过其追溯激励结果数据。可信众测激励服务端以区块链基础平台为底层平台, 在平台基础上通过调用其提供的对外开放 API 接口来实现各类服务, 既通过区块链数据的真实性保障了数据记录的可信性, 又通过智能合约的自动化流程提高了整个众包测试激励过程的效率, 从而在维护工人方的权益的同时提高了其工作积极性, 做到真正可信且有效的众测激励。

区块链基础平台作为底层平台, 是建立可靠共性服务、满足众包测试流程中各项需求的基础。当前本系统中仅有工人方可信众测激励服务端接入, 因此本文为面向可信众测激励的区块链服务系统, 采用 BaaS 架构的目的为便于后续新功能的接入与拓展, 如需求方激励或是众测资产确权等。本架构将区块链框架纳入云平台, 结合云服务的优势, 为上层开发者提供了性能与效率均提升的区块链配套服务。平台分为对外服务层与基础设施层两大部分。服务层主要封装了一些区块链的服务功能并对外提供 API 调用, 包括数据上链、数据溯源、资产确权等。基础设施层中采用 IBM 提供的 Hyperledger Fabric 超级账本开源区块链框架, 部署 Fabric 多节点的集群, 包含共识机制、智能合约、CA 认证、节点集群等多种服务功能, 采用 IPFS 进行链上链下相结合的文件存储方式, 配置映射数据库 CouchDB 作为缓存以优化区块链平台的数据读取性能。

4.3.2 4+1 视图

根据软件工程中 Kruchten 给出的“4+1”视图方法，本文分别从逻辑、开发、进程和物理视图四个不同视角出发，描述了软件体系结构，反映了动静态与软硬件部分间的联系，对系统进行整体的结构设计。

(1) 逻辑视图

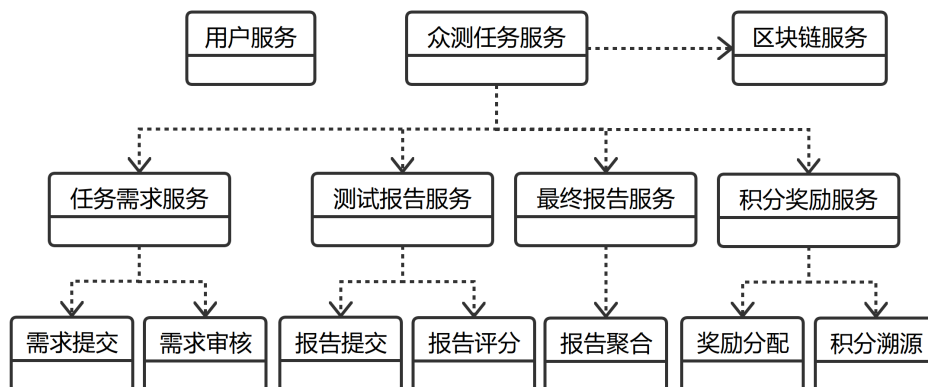


图 4-3: 系统逻辑视图

图 4-3描述了系统逻辑结构，从对象角度介绍了系统各模型。本文中数据实体可按众测步骤抽象为需求提交、需求审核、报告提交、报告评分、报告聚合、奖励分配、积分溯源七种模型，基础服务可抽象为用户服务、任务服务、需求服务、测试报告服务、最终报告服务、积分奖励服务、区块链服务七种模型。用户服务包含用户登录验证等功能；通过众测任务服务三方可进行任务的进度溯源；通过需求服务需求方可提交与查询需求，平台方可审核需求；通过报告服务工人方可提交与查询报告，平台方可审核报告；通过最终报告服务需求方可查询来源，平台方可提交报告；通过积分奖励服务三方可查询总奖励与积分分配结果。区块链服务提供区块链基础功能，包括数据上链、溯源等。

(2) 开发视图

上述逻辑设计按照用户视角推进，图 4-4给出了以开发者视角设计的系统开发结构。系统分离前后端，做到高内聚低耦合，根据分层设计的原则，划分为展示层、业务逻辑层和数据存储层。展示层采用 Vue 构建，按照源码文件夹下的目录分为五个部分，静态资源如 css 文件与图片等置于 assets，通用的模块组件置于 components，状态关联文件置于 stores，路由设置文件置于 routes，页面展示文件置于 views。业务逻辑层采用 SpringBoot 构建，展示层通过 VO 模型与业务层交互，数据存储层通过 Model 模型与业务层交互。其中，VO 模型包

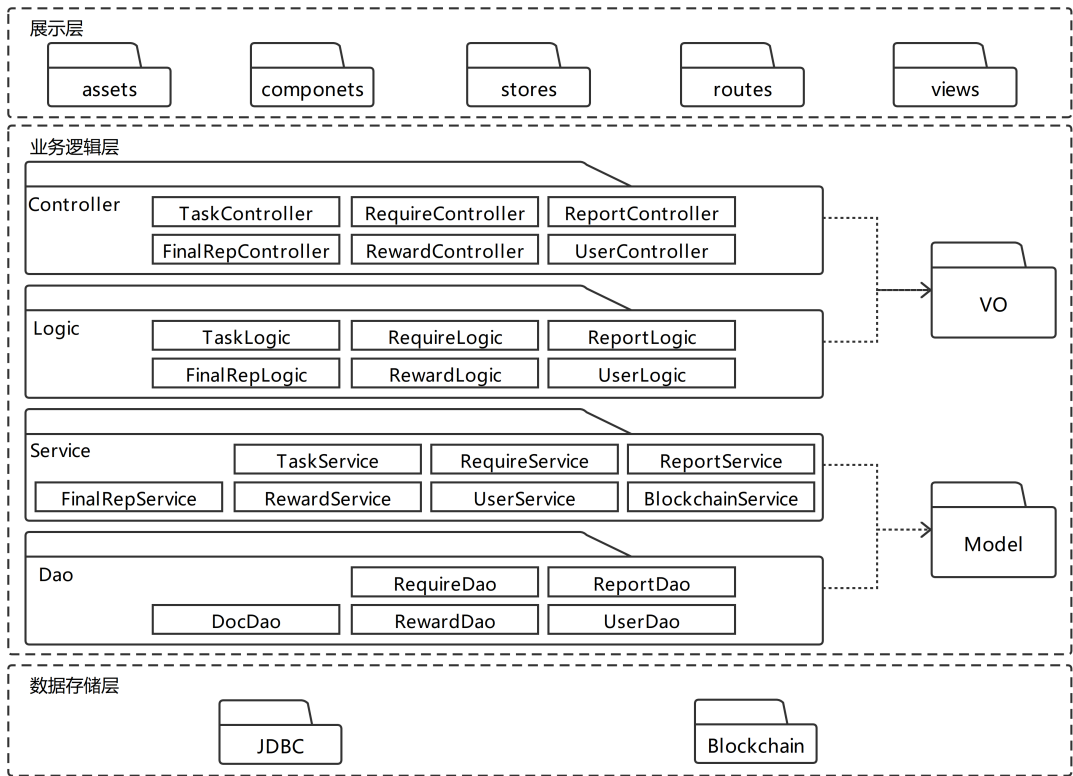


图 4-4: 系统开发视图

含接收前端请求交由下层处理的 Controller 层与根据业务加工数据的 Logic 层，Model 模型包含封装数据存取的 Service 层与直接操作数据的 Dao 层。数据存储层对外提供数据操作接口，如业务数据的 JDBC 与可信数据的区块链。

(3) 进程视图

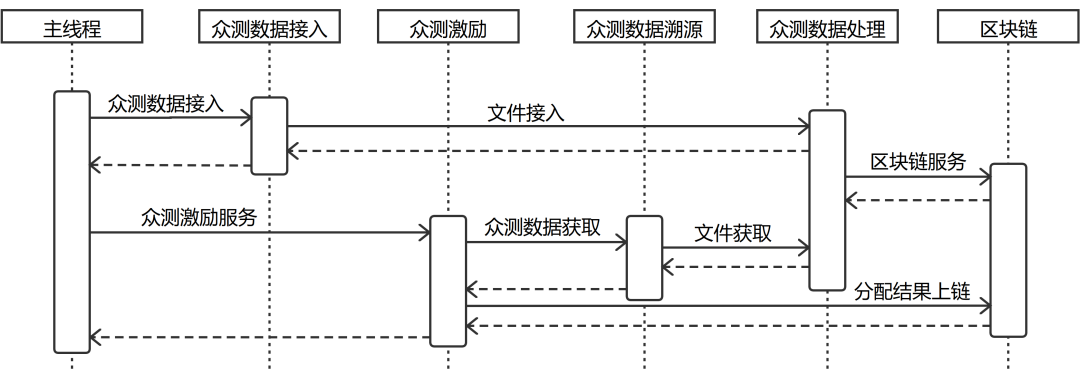


图 4-5: 系统进程视图

可信众测激励系统的进程视图如图 4-5所示，通过系统各服务进程间交互过程来描述其关联。系统主线程从两部分分别开始。任务开始后，首先主线程

接入众测数据，即流程中需求提交、需求审核、测试报告提交、缺陷报告审核、最终报告聚合阶段的众测数据实时接入可信众测激励系统，其中由于激励系统暂时只接入基础数据，此处存入文件哈希值，代替文件实体。数据经过处理后发送给区块链平台，调用服务通过智能合约发起交易，共识通过后自动上链存入账本。任务完成后，主线程再次由众测激励服务进行调用。首先调用众测数据溯源服务，对任务中所有报告的评分信息获取，随后依据获取的评分数据进行任务奖励的分配，将结果实时存入区块链，并反馈给主线程，主线程也可以通过服务进行奖励结果的查询。其中，众测溯源服务调用区块链服务直接查询账本数据即可，无需进行节点共识。

(4) 物理视图

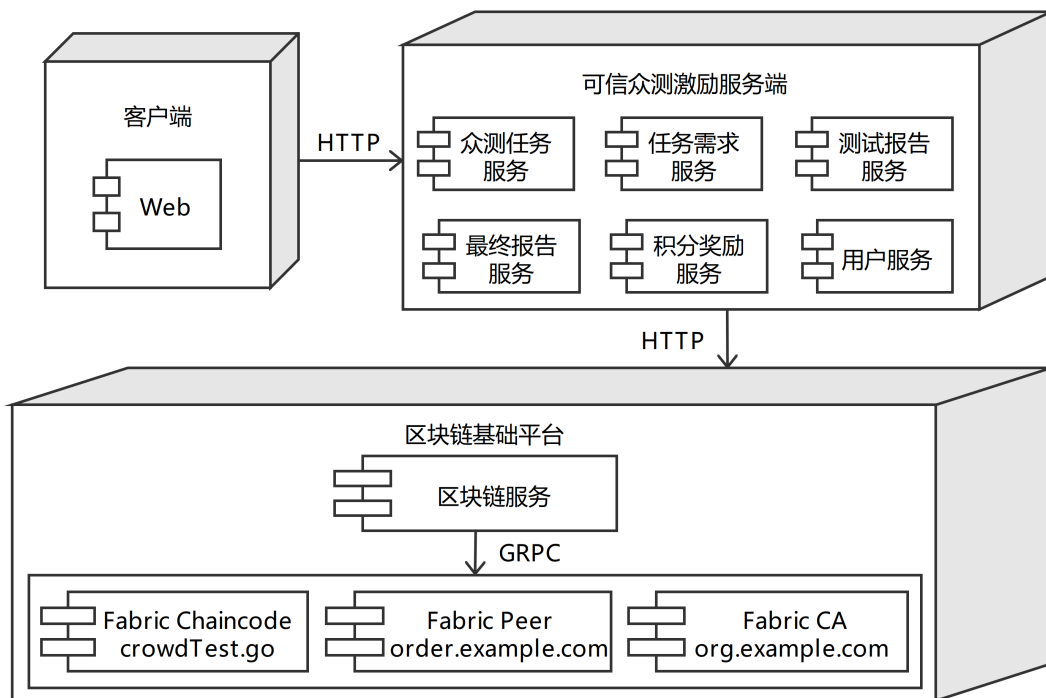


图 4-6: 系统物理视图

本系统的物理视图如图 4-6所示，通过展示在物理上系统完整各模块部署的方式，描述软件与硬件间映射关系。系统分为客户端、可信众测激励服务端与区块链基础平台。Docker 容器中分别部署了上述三个部分。客户端为 web 页面，通过网络中 HTTP 协议与服务端交互并使用其功能接口。服务端同样通过 HTTP 访问区块链平台的对外服务，而后者则通过 GRPC 协议访问 Hyperledger Fabric 节点集群，在平台中基础设施层部署了相关的各个组件，如 Fabric CA 身份验证组件、Fabric Peer 节点网络、Fabric Chaincode 合约链码等。

4.4 持久化模型设计

Fabric 支持启用 CouchDB 作为状态数据库，来映射世界账本中的数据。账本数据要求以键值对形式存储，本文采用 Java 中 UUID 类生成的唯一标识作为键 key，众测数据对象作为值 value。在众测数据对象方面，本文根据众测流程设计了 7 种联盟链存储模型，下面将分别介绍各个数据模型。

表 4-7: 众测任务数据模型

定义	字段	类型
任务标识	taskId	string
任务名称	taskName	string
任务状态	taskState	Integer
更新时间	updateTime	long

众测任务 TaskState 属性数据的联盟链存储模型如表 4-7所示。任务共有六种状态，分别对应前文的需求提交与审核、测试报告与审核、最终报告聚合以及奖励分配，当各个阶段提交数据后，智能合约都会更新任务当前状态。

表 4-8: 需求提交数据模型

定义	字段	类型
任务标识	taskId	string
任务名称	taskName	string
需求标识	requestId	string
需求方标识	requesterId	string
需求方姓名	requesterName	string
需求文档名	requestDocName	string
待测软件名	testSoftwareName	string
需求文档哈希值	requestDocHash	string
更新时间	updateTime	long

需求方的需求提交 RequestCommit 联盟链存储数据模型如表 4-8所示，反映了任务与需求之间的映射。taskId 为任务标识，taskName 为该任务名称，requesterName 为需求方姓名。需求方通过 requestId 溯源查询其需求提交数据，平台方通过 taskId 溯源查询该任务的需求提交数据。需求文档为 requestDocName，测试软件为 testSoftwareName。requestDocHash 为区块链存储

的文档唯一哈希值，以保证工人下载的需求文档与需求方提交的一致，从而证明数据的未被篡改、真实可靠。

表 4-9: 需求审核数据模型

定义	字段	类型
任务标识	taskId	string
任务名称	taskName	string
需求标识	requestId	string
需求审核结果	requestReviewResult	string
需求审核者	requestReviewer	string
更新时间	updateTime	long

需求审核 RequestReview 联盟链存储数据模型如表 4-9所示，描述了任务数据与需求审核数据之间的映射。reviewResult 记录需求审核结果，根据 taskId 查看需求审核数据。requestReviewer 为该需求的平台审核者信息。

表 4-10: 报告提交数据模型

定义	字段	类型
任务标识	taskId	string
任务名称	taskName	string
工人标识	workerId	long
工人姓名	workerName	long
测试报告标识	testReportId	string
测试报告名称	testReportName	string
测试报告哈希值	testReportHash	string
缺陷报告列表	bugReportList	list
更新时间	updateTime	long

工人方的测试报告提交 TestReport 联盟链存储数据模型如表 4-10所示，描述了任务与测试报告之间的映射。workerId 为工人方唯一标识，workerName 为工人名称。测试报告标识为 testReportId，报告名称为 testReportName，testReportHash 为该报告的哈希值，这里哈希值唯一，从而可以验证报告是否被篡改过。工人与测试报告提交模型的对应关系为一对一即仅能提交一份，但是该报告由多个缺陷报告组成。而包含的缺陷报告们分别与其具体信息如 Id、名称等封装组合成列表 bugReportList。

表 4-11: 报告审核数据模型

定义	字段	类型
任务标识	taskId	string
任务名称	taskName	string
测试报告标识	testReportId	string
缺陷报告标识	bugReportId	string
缺陷报告审核者	bugReportReviewer	string
缺陷报告评分	bugReportScore	long
更新时间	updateTime	long

缺陷报告的审核 ReportReview 联盟链存储数据模型如表 4-11所示，描述了测试报告与缺陷报告得分间的关系。工人方通过 taskId 溯源该任务下提交的所有缺陷报告的审核评分结果。平台方通过 taskId 溯源缺陷报告审核数据，查询该任务下所有缺陷报告的评分结果。缺陷报告标识为 bugReportId，缺陷报告审核者为 bugReportReviewer，缺陷报告得分为 bugReportScore。在审核报告过程中，平台方以缺陷报告为粒度并对其进行评分，最后该测试报告的总分数由包含的所有缺陷报告得分求平均值所得。

表 4-12: 报告聚合数据模型

定义	字段	类型
任务标识	taskId	string
任务名称	taskName	string
最终报告融合者	finalReportMixer	string
最终报告哈希值	finalReportHash	string
融合缺陷报告列表	mixedBugReportList	list
更新时间	updateTime	long

平台方缺陷报告聚合为最终报告的 ReportMix 联盟链存储数据模型如表 4-12所示，反映了任务与报告聚合之间的关系。由平台方聚合者该任务下在所有缺陷报告中选取一部分进而进行聚合组成最终报告并最后交付给需求方，因此在一次众测任务中，只有一份最终报告会交付给需求方。mixedBugReportList 为被聚合的缺陷报告列表，finalReportHash 为最终报告哈希值，这里的哈希值同样是唯一的，在验证中数据无法篡改，以保证报告的真实性。finalReportMixer 为最终报告融合者。

表 4-13: 奖励分配数据模型

定义	字段	类型
任务标识	taskId	string
任务名称	taskName	string
任务奖励总分	taskReward	double
工人奖励列表	workerRewardList	list
更新时间	updateTime	long

平台方记录奖励分配 RewardAssign 的联盟链存储数据模型如表 4-13所示，表示众测任务与奖励分配之间的映射。不论是哪类用户，在获取奖励信息时，均是通过任务标识 taskId 来进行查询。在任务完成激励后，所有参与工人的激励结果组成列表后表示该任务最终奖励结果，一名工人对应一份测试报告，也对应一份奖励结果。taskReward 为任务奖励总分，workerRewardList 为工人奖励列表，列表中包含每个工人的 Id、姓名与获得的积分等信息。

4.5 本章小结

本章从全局角度出发，对系统进行需求用例分析，并进行了总体架构上的设计。首先，本章简要描述了本系统情况，然后，对不同涉众参与方群体进行需求分析，根据分析结果绘制用例图，针对各个用例详细阐述其流程，除此之外总结本文需求，涉及功能性与非功能性方面。再次，本章从全局出发给出了完整的架构图，按照分层结构展示，并遵循“4+1 视图”的原则给出了逻辑、开发、进程、物理四大视图，分别从用户、开发者、进程、物理部署的视角对系统整体结构进行了解剖与分析。最后，由于系统数据需以键值对形式存于区块链中，本章设计了七种持久化数据模型。

第五章 系统详细设计与实现

基于需求分析与总体设计，本章将以类图、顺序图与代码等形式详细介绍服务端各模块的设计与实现及区块链平台的部署，最后展示系统界面。

5.1 众测激励系统服务设计与实现

本章将分为六个模块进行介绍：众测任务服务，负责平台方、需求方与工人方的任务进度溯源；任务需求服务，负责需求方的需求提交与溯源；测试报告服务，负责工人方的报告提交与溯源；最终报告服务，负责需求方的最终报告聚合与溯源；积分奖励服务，负责需求方与工人方的奖励分配与溯源；区块链基础平台服务，负责区块链与平台的数据交互。

5.1.1 众测任务服务

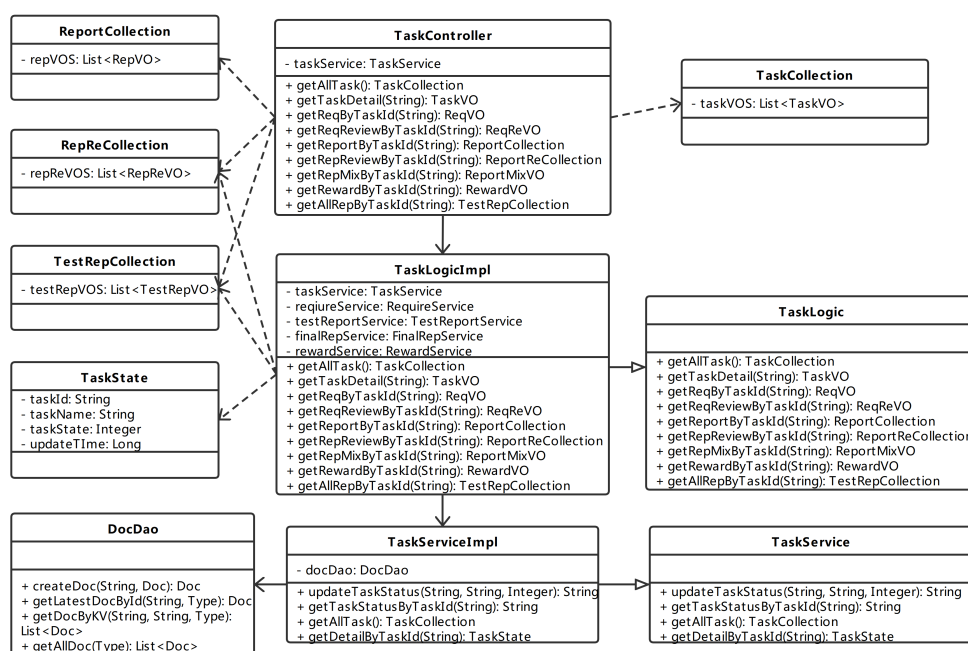


图 5-1: 众测任务服务核心类图

众测任务服务主要包括需求方、工人方以及平台方三方的众测任务进度溯源。这里根据此前的定义，分成 6 个阶段并分别对每个阶段设定一个状态值代表当前任务所处的阶段。比如，需求提交阶段表示为 0，需求审核阶段表示为 1，报告提交阶段表示为 2，报告审核阶段表示为 3，报告聚合阶段表示为 4，奖励分配阶段表示为 5，而在联盟链中为便于记录众测任务本身采用属性值 6 来表示。对于需求方来说，由于是任务发起者，只关心需求进度以及最终结果，因此能够看到该任务的需求提交、需求审核以及报告融合 3 个阶段。对于工人方来说，由于是任务完成者，只关心报告进度以及激励结果，因此能够看到该任务的报告提交、报告审核以及奖励分配 3 个阶段。对于平台方来说，作为第三方平台，有权限查看任务完整的进度，因此能够看到全部 6 个阶段。

众测任务服务核心类图如图 5-1 所示。TaskController 类为控制器，内含各类接口，返回不同阶段的数据。Controller 层与前端交互的数据模型为 TaskCollection、ReportCollection、RepReCollection、TestRepCollection。众测任务的业务逻辑实现为 TaskLogic 类，依赖其余众测服务类来实现。获取某个任务的详细数据需通过 getTaskDetail 方法，根据 taskId 获取需求数据需通过 getReqBy-TaskId 方法，根据 taskId 获取报告审核数据需通过 getRepReviewByTaskId 方法，根据 taskId 获取报告聚合数据需通过 getRepMixByTaskId 方法，获取所有众测任务信息需通过 getAllTask 方法。TaskService 类包含众测任务信息查询、状态更新等服务。TaskState 类是众测任务数据联盟链存储模型。

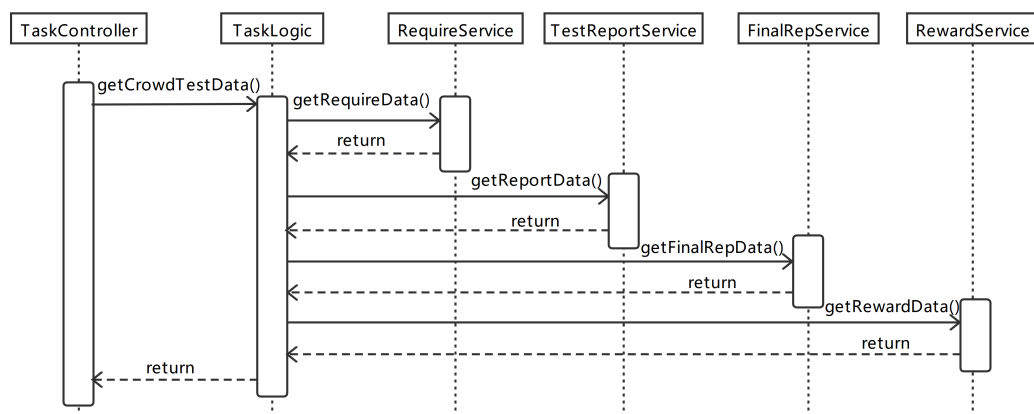


图 5-2: 众测任务溯源顺序图

图 5-2 展示了众测任务溯源调用顺序。首先由 TaskController 向 TaskLogic 发送请求，随后后者调用数据溯源接口得到众测数据，可以获取众测流程中各个阶段及其服务的数据，比如通过任务需求服务得到需求提交与审核的数据，

通过测试报告服务得到报告提交与审核的数据，通过最终报告服务得到报告聚合的数据，通过积分奖励服务得到激励的数据。最后将数据封装为数据模型返回给 TaskController。

```
//众测任务溯源
public TestRepCollection getAllrepByTaskId(String taskId) {
    TestRepCollection result = new TestRepCollection();
    List<TestReportVO> testReportVOS = new ArrayList<>();
    List<TestReport> testReports = this.testReportService.getTestRepByTaskId(taskId);
    testReports.forEach(v->{
        TestReportVO testReportVO = new TestReportVO();
        testReportVO.setTaskId(v.getTaskId());
        //省略参数赋值代码
        List<String> bugList = v.getBugReportList();
        List<BugRepScoreVO> bugRepScoreVOS = new ArrayList<>();
        bugList.forEach(w->{
            BugRepScoreVO bugRepScoreVO = testReportService.getBugVOBybugId(w);
            if(bugRepScoreVO.getBugID()!=null){
                bugRepScoreVO.setBugID(w);
            }
            bugRepScoreVOS.add(bugRepScoreVO);
        });
        testReportVO.setBugReport(bugRepScoreVOS);
        testReportVOS.add(testReportVO);
    });
    result.setTestReportVOS(testReportVOS);
    return result;
}
```

图 5-3: 众测任务溯源关键代码

众测任务溯源关键代码如图 5-3所示，这里展示获取所有测试报告的方法代码。首先根据所选任务的 taskId 通过测试报告服务查询到该任务下所有的测试报告组成的列表，随后将列表中每一份报告转成 TestReportVO 格式，其中包含缺陷报告列表。对每个缺陷报告，同样通过报告服务获取到其信息，若已审核则添加 bugRepScoreVO 格式的审核实体。最后将数据封装为 testReportVOS 模型并返回。

5.1.2 任务需求服务

任务需求服务包括需求方的需求提交与溯源、平台方的需求审核。需求提交内容包括待测软件、需求文档等内容，需求审核内容为平台方人员的审核结果，需求溯源展示需求方参与的所有任务的具体进度。图 5-4为任务需求服务的核心类图。其中，ReqController 类是提供外部服务接口的控制器，

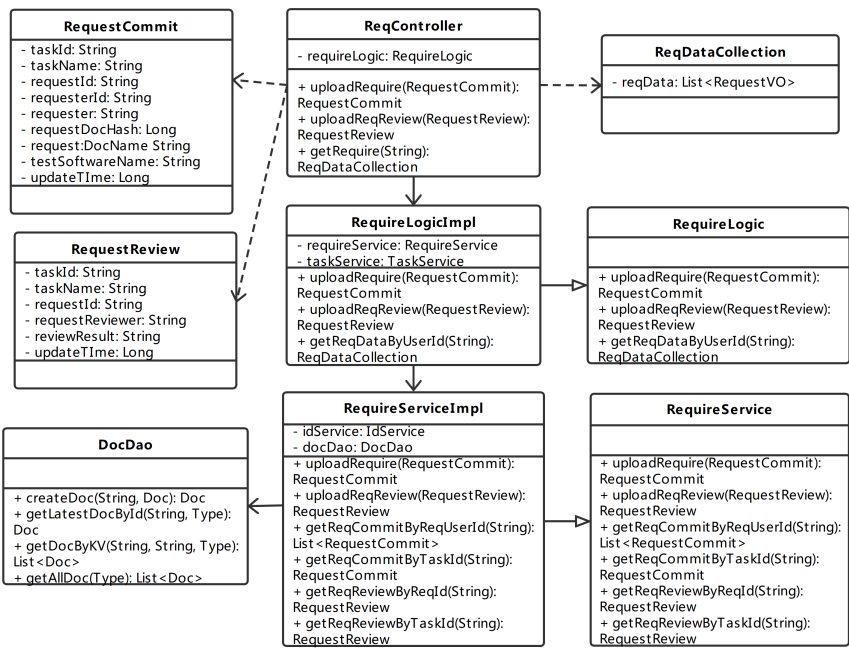


图 5-4: 任务需求服务核心类图

RequireLogic 类是处理具体的复杂业务逻辑，RequireService 类通过 Dao 将获取的数据上链以保证可信性。RequestCommit、RequestReview 分别为需求提交与审核的数据模型。

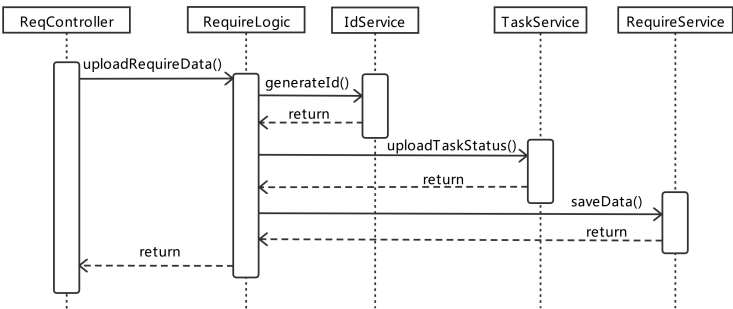


图 5-5: 任务需求提交顺序图

图 5-5展示了任务需求的提交顺序。首先由 ReqController 在接收需求数据，随后调用 RequireLogic 中的需求数据上传接口。其中，后者先通过 IdService 生成联盟链中需求的唯一标识，再通过任务服务更新当前任务状态，最后通过需求服务将数据封装为模型格式后上链。

图 5-6展示了任务需求溯源顺序。首先由 ReqController 收到溯源请求，随后通过任务服务获取当前状态，通过需求服务获取需求及其审核数据，若可以则继续通过最终报告服务获取最终报告数据。

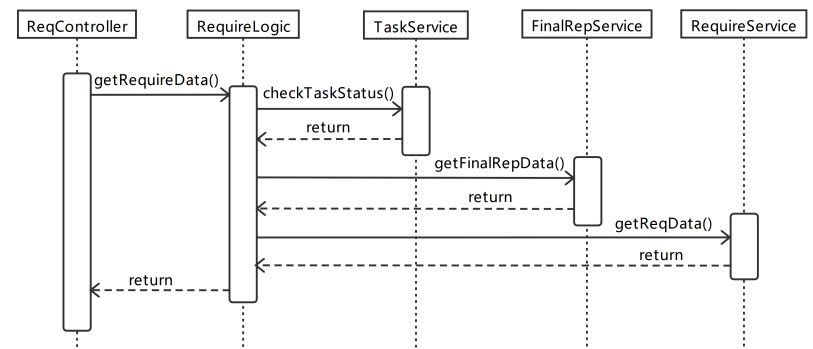


图 5-6: 任务需求溯源顺序图

图 5-7展示了任务需求提交的关键代码。在需求提交过程中，首先先调用任务类方法更新任务状态，需求提交阶段状态值为 0。随后将封装后的 requestCommit 数据模型返回给需求服务。需求审核与需求提交的逻辑一致，但是任务状态值为 1。服务层将业务数据发送给 Dao 层处理。docService 封装对区块链服务的调用，createDoc 为数据上链对外接口。

```
//需求提交
public RequestCommit uploadRequire(RequestCommit requestCommit) {
    taskService.updateTaskStatus(
        requestCommit.getTaskId(),requestCommit.getTaskName(),0);
    return this.requireService.uploadRequire(requestCommit);
}

public RequestCommit uploadRequire (RequestCommit requestCommit) {
    String docId = idService.genId();
    this.docDao.createDoc(docId,requestCommit);
    return requestCommit;
}
```

图 5-7: 任务需求提交关键代码

```
//需求溯源
public ReqDataCollection getReqDataByUserId(String usrId) {
    ReqDataCollection reqDataCollection = new ReqDataCollection();
    List<RequestVO> requestVOList= new ArrayList<>();
    List<RequestCommit>requestCommitList=
        this.requireService.getReqCommitByUserId(usrId);
    requestCommitList.forEach(v->{
        String taskStatus = this.taskService.getTaskStatusBytaskID(v.getTaskId());
        //省略赋值代码
        requestVOList.add(requestVO);});
    reqDataCollection.setReqData(requestVOList);
    return reqDataCollection;
}
```

图 5-8: 任务需求溯源关键代码

图 5-8展示了任务需求溯源的关键代码。在需求溯源过程中，首先根据需求方用户 Id，查询到其参与的所有任务并返回任务列表。随后对于每项任务，根据其任务 Id 可查询到任务具体信息。最后返回给前端时，转化为 requestVO 格式。

5.1.3 测试报告服务

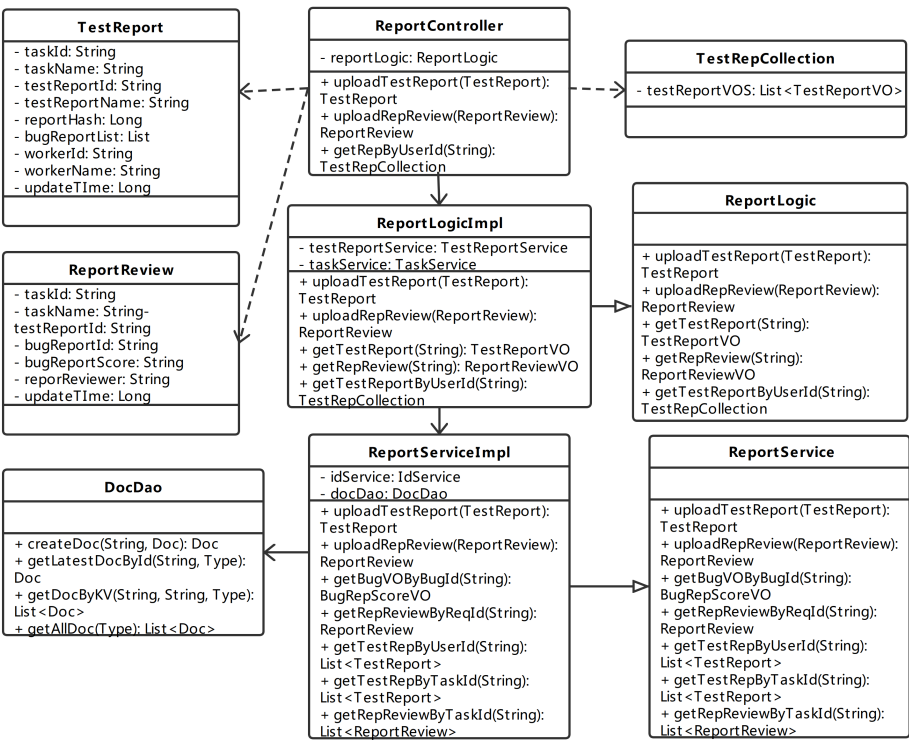


图 5-9: 测试报告服务核心类图

测试报告服务主要包括工人方的报告提交与溯源、平台方的审核。其中，溯源服务是对报告及其审核数据的溯源。图 5-9为测试报告服务核心类图。**ReportController** 类是提供对外接口的控制器。**ReportLogic** 类负责处理复杂业务逻辑，比如在需求上传阶段将数据处理为联盟链数据模型，或是在数据溯源阶段，鉴于平台审核报告时粒度划分到缺陷报告，需将测试报告与其内含所有缺陷报告组成的审核结果列表打包一同返回。**RequireService** 类封装了详细的数据读取操作。此外，该阶段的报告数据在传输中会被封装为多种持久化模型，比如测试报告数据模型 **TestReport** 类，报告审核数据模型 **ReportReview** 类，报告溯源数据模型 **TestRepCollection** 类。

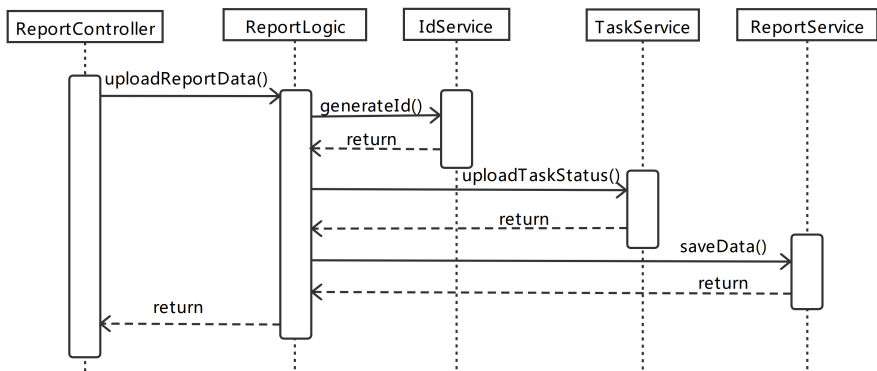


图 5-10: 测试报告提交顺序图

图5-10展示了测试报告的提交顺序。在测试报告提交流程中，首先由 ReportController 调用 ReportLogic 中的测试报告提交接口，接着后者生成区块链中报告的唯一标识，随后通过任务服务更新当前任务的状态，最后通过测试报告服务进行数据上链存储。

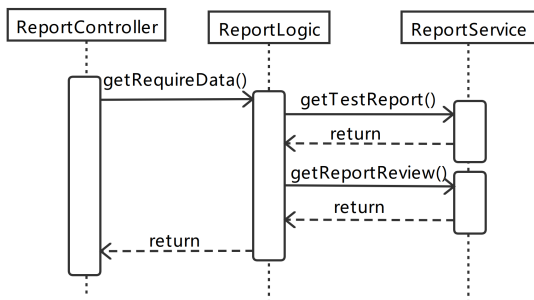


图 5-11: 测试报告溯源顺序图

图5-11展示了测试报告的溯源顺序。在测试报告溯源流程中，首先由 ReportController 调用 ReportLogic 中的测试报告溯源方法，随后后者通过测试报告服务分别获取该任务的测试报告及缺陷报告审核数据并返回结果。

测试报告提交关键代码如图 5-12所示。在测试报告提交方法中，首先接收 TestReport 模型数据，根据其 taskId 调用任务服务获取当前任务状态。由于在一次众测任务有多位工人将提交报告，可能存在某位工人提交的报告已经被平台方审核从而导致状态由报告提交阶段进入到报告审核阶段的情况，若当前状态已发生改变，则更改为报告提交阶段。最后调用测试报告服务将测试报告数据返回给服务层。缺陷报告审核的提交业务逻辑与测试报告一致，但是任务状态值为 3。其中，getTaskStatusBytaskId 方法实现了根据任务 Id 来获取任务状态，若没有任务状态则返回-1 代表需要更新。

```
//测试报告提交
public TestReport uploadTestReport(TestReport testReport) {
    String taskId = testReport.getTaskId();
    String status = this.taskService.getTaskStatusBytaskId(taskId);
    if(status.equals("-1")||status.equals("2")){
        this.taskService.updateTaskStatus(taskId,testReport.getTaskName(),2);
    }
    return this.testReportService.uploadTestReport(testReport);
}

//测试报告评分
public ReportReview uploadRepReview(ReportReview reportReview) {
    String taskId = reportReview.getTaskId();
    String status = this.taskService.getTaskStatusBytaskId(taskId);
    if(status.equals("-1")||status.equals("3")){
        this.taskService.updateTaskStatus(taskId,reportReview.getTaskName(),3);
    }
    return this.testReportService.uploadRepReview(reportReview);
}
```

图 5-12: 测试报告提交关键代码

测试报告溯源关键代码如图 5-13所示。在测试报告溯源方法中，首先根据工人 Id 获取其提交的测试报告列表。然后对于列表中每一份测试报告，获取该报告包含的缺陷报告数据列表。接着对列表中每一份缺陷报告，获取其审核数据。最后将结果封装为 TestRepCollection 测试报告集合后返回给前端。

```
//测试报告溯源
public TestRepCollection getTestReportByUsrId(String usrId) {
    TestRepCollection result = new TestRepCollection();
    List<TestReportVO> testReportVOS = new ArrayList<>();
    List<TestReport> testReports = this.testReportService.getTestRepByUserId(usrId);
    testReports.forEach(v->{
        TestReportVO testReportVO = new TestReportVO();
        testReportVO.setTaskId(v.getTaskId());
        //省略参数设置代码
        List<String> bugList = v.getBugReportList();
        List<BugRepScoreVO> bugRepScoreVOS = new ArrayList<>();
        bugList.forEach(w->{
            BugRepScoreVO bugRepScoreVO =testReportService.getBugVOBybugId(w);
            if(bugRepScoreVO.getBugID()==null){
                bugRepScoreVO.setBugID(w);
            }
            bugRepScoreVOS.add(bugRepScoreVO);
        });
        testReportVO.setBugReport(bugRepScoreVOS);
        testReportVOS.add(testReportVO);
    });
    result.setTestReportVOS(testReportVOS);
    return result;
}
```

图 5-13: 测试报告溯源关键代码

5.1.4 最终报告服务

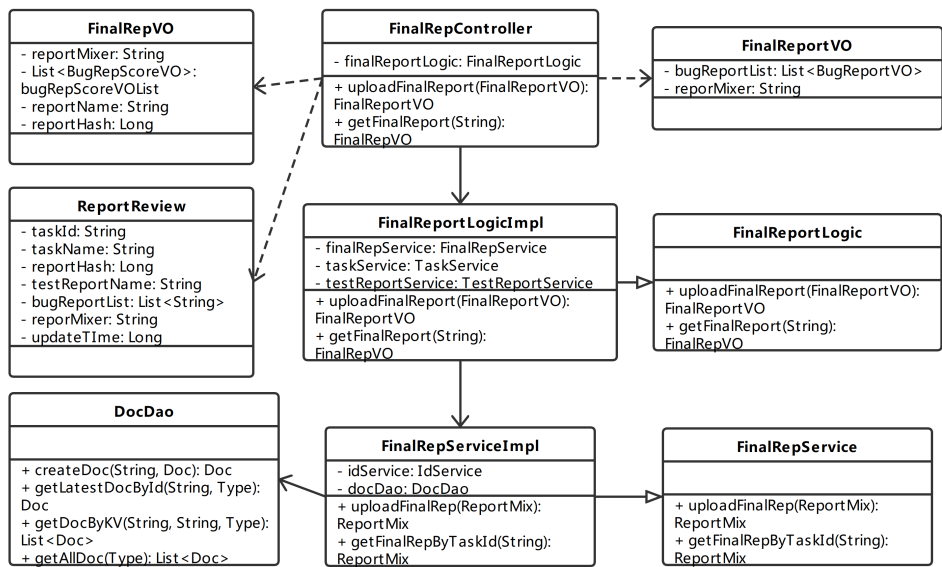


图 5-14: 最终报告服务核心类图

最终报告服务包括平台方的最终报告提交与需求方与平台方溯源服务。最终报告提交的内容包括基本信息及其聚合的缺陷报告列表等。最终报告溯源中，需求方可以查看平台交付的最终报告信息，平台方可以追溯聚合来源。最终报告服务核心类图如图 5-14所示。FinalRepController 类是对外接口的控制器，FinalReportLogic 类处理业务逻辑，FinalRepService 类封装数据存储接口。联盟链中持久化数据模型为 ReportMix，与前端交互的数据模型为 FinalRepVO，与平台交互的数据模型为 FinalReportVO。

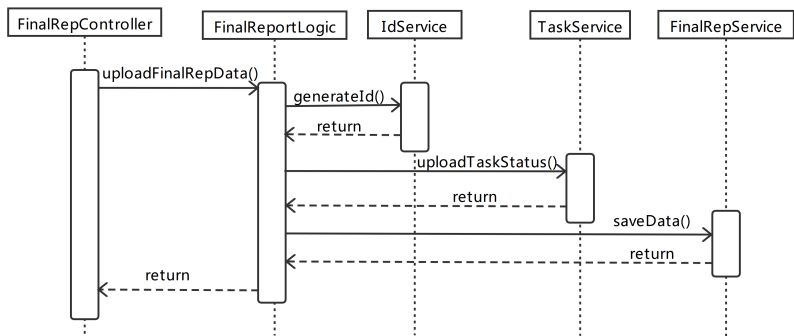


图 5-15: 最终报告提交顺序图

图 5-15展示了最终报告提交顺序。在提交过程中，首先由 FinalRepController 调用 FinalReportLogic 进行上传请求，随后后者接收数据后转换为持久化

模型，接着生成唯一标识，再调用任务服务更新当前任务状态，最后由最终报告服务将封装好的数据上链。

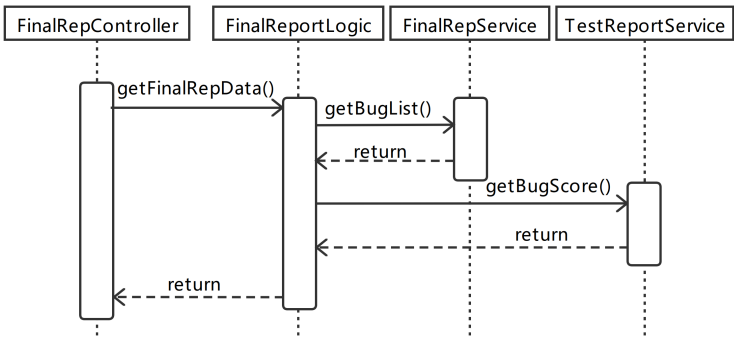


图 5-16: 最终报告数据溯源顺序图

图 5-16展示了最终报告溯源顺序。在溯源过程中，首先由 FinalRepController 调用 FinalReportLogic 请求查询任务最终报告数据，随后后者通过最终报告服务来获取缺陷报告列表，接着根据列表中每个缺陷报告的 Id 调用测试报告服务查询其得分，最后将数据封装后返回。

```
//最终报告提交
public FinalReportVO uploadFinalReport(FinalReportVO finalReportVO) {
    this.taskService.updateTaskStatus(
        finalReportVO.getTaskId(),finalReportVO.getTaskName(),4);
    ReportMix reportMix = new ReportMix();
    reportMix.setTaskId(finalReportVO.getTaskId());
    reportMix.setTaskName(finalReportVO.getTaskName());
    reportMix.setReportMixer(finalReportVO.getReportMixer());
    reportMix.setUpdateTime(finalReportVO.getUpdateTime());
    List<BugReportVO> bugReportVOList = finalReportVO.getBugReportList();
    List<String> bugRep = new ArrayList<>();
    bugReportVOList.forEach(v->{
        bugRep.add(v.getBugReportID());
    });
    reportMix.setBugReportList(bugRep);
    this.finalRepService.uploadFinalRep(reportMix);
    return finalReportVO;
}
```

图 5-17: 最终报告提交关键代码

最终报告提交关键代码如图 5-17所示。最终报告提交时，首先调用任务服务将任务状态更新为报告融合阶段，该阶段不需要校正当前任务状态。接着与区块链进行交互，将此前获取的最终报告 VO 模型转换为联盟链数据模型，随后遍历缺陷报告列表，将缺陷报告信息存入 bugReportVO 类列表。最后调用最终报告服务将数据封装为 ReportMix 返回。

```
//最终报告溯源
public FinalRepVO getFinalReport(String taskId) {
    FinalRepVO result = new FinalRepVO();
    List<BugRepScoreVO> bugRepScoreVOS = new ArrayList<>();
    ReportMix reportMix = this.finalRepService.getFinalRepByTaskID(taskId);
    List<String> bugList = reportMix.getBugReportList();
    bugList.forEach(v->{
        BugRepScoreVO bugRepScoreVO = testReportService.getBugVOBybugId(v);
        bugRepScoreVOS.add(bugRepScoreVO);
    });
    result.setReportMixer(reportMix.getReportMixer());
    result.setBugRepScoreVOList(bugRepScoreVOS);
    result.setReportHash(reportMix.getReportHash());
    return result;
}
```

图 5-18: 最终报告溯源关键代码

最终报告溯源关键代码如图 5-18所示。最终报告溯源时，首先根据任务 Id 调用最终报告服务获取报告聚合数据 **reportMix**，对其中的缺陷报告列表 **bugList** 中每个缺陷报告，通过测试报告服务获取其审核数据，并封装为 **bugRepScoreVO**。最后将列表组装为 **FinalRepVO** 并返回。

5.1.5 积分奖励服务

积分奖励服务是针对众测流程中的激励环节设计的服务，包括积分奖励提交与分配及积分奖励溯源服务。积分奖励提交主要由需求方在需求提交后，提交该任务的总积分奖励，积分奖励数据包括任务总积分及各个工人的积分分配列表。积分奖励分配则当任务完成后，由平台自动按照每位工人的测试报告得分进行积分分配，即提交一个奖励分配格式的列表，与任务的测试报告列表相对应。溯源服务是在任务完成时，需求方和平台方可以查看该任务下总积分对于每一位参与工人的分配结果。

积分奖励服务核心类图如图 5-19所示。**RewardController** 类是对外提供的控制器，定义了奖励数据提交接口 **uploadReward**、奖励数据分配接口 **assignReward** 以及奖励数据溯源接口 **getReward**。**RewardLogic** 类处理具体业务逻辑，在奖励提交与分配时向 **Service** 层服务传入转换好的联盟链数据模型，在奖励结果溯源时获取数据需转化为与前端交互的数据模型再交由 **Controller** 层处理。**RewardService** 类为上层封装数据存储接口。**RewardSubmit** 类是积分奖励提交的数据模型，**RewardIntegral** 类是积分奖励分配的数据模型，**RewardCollection** 类是积分奖励溯源数据模型。

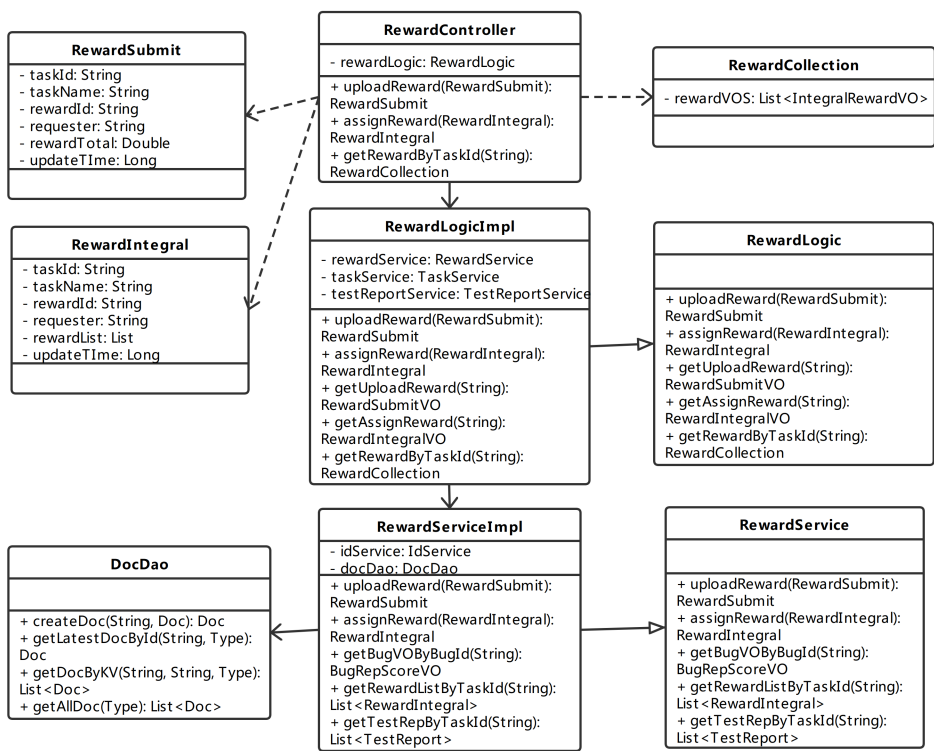


图 5-19: 积分奖励服务核心类图

积分奖励提交与分配调用顺序如图 5-20所示。首先由 RewardController 接收到平台发送的最终报告数据后，调用 RewardLogic 转换为联盟链数据模型，随后生成的唯一主键，再调用任务服务更新任务状态。在提交奖励时，直接调用 RewardService 存入总积分；在分配奖励时，在上述步骤后调用测试报告服务以获取分配依据进行分配，结果组装为 RewardIntegral 模型，最后 RewardService 将积分奖励数据发送给联盟链。

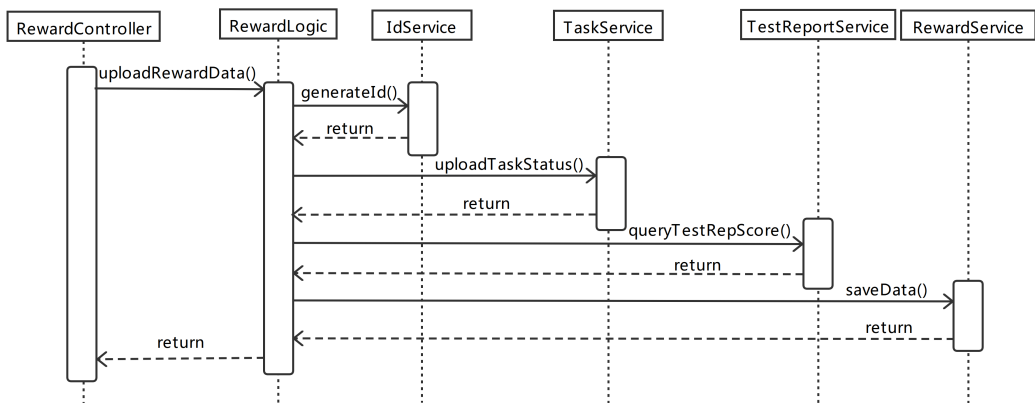


图 5-20: 积分奖励提交与分配顺序图

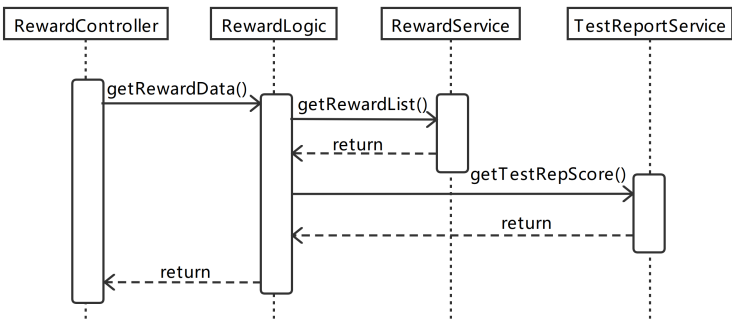


图 5-21: 积分奖励溯源顺序图

积分奖励溯源调用顺序如图 5-21所示。在溯源过程中，首先积分奖励溯源请求发送至 **RewardController**，调动逻辑层查询包含所有工人的激励分配列表的积分奖励数据。随后后者根据测试报告 **Id**，调用测试报告服务查询该任务下每个测试报告的评分数据，最后将数据封装为积分奖励数据溯源模型 **RewardCollection** 后，返回给 **RewardController**。

```
//积分奖励提交
public RewardSubmit uploadReward(RewardSubmit rewardSubmit) {
    String taskId = rewardSubmit.getTaskId();
    String status = this.taskService.getTaskStatusBytaskId(taskId);
    if(status.equals("-1")||status.equals("1")){
        this.taskService.updateTaskStatus(taskId,rewardSubmit.getTaskName(),1);
    }
    return this.rewardService.uploadReward(rewardSubmit);
}

//积分奖励分配
public RewardIntegral assignReward (RewardIntegral rewardIntegral) {
    this.taskService.updateTaskStatus(
        rewardIntegral.getTaskId(),rewardIntegral.getTaskName(),5);
    Double totalReward = rewardIntegral.getTotalReward();
    Int totalScore = rewardIntegral.getTotalScore();
    List<TestReportVO> testReportVOList =
        testReportVO.getTestReportListByTaskID(rewardIntegral.getTaskId());
    List<SingleRewardIntegral> rewardIntegralList = new ArrayList<>();
    testReportVOList.forEach(v->{
        SingleRewardIntegral singleRewardIntegral = new SingleRewardIntegral();
        singleRewardIntegral.setWorker(v.getWorker());
        singleRewardIntegral.setReward(v.getTestReportScore() / totalScore * totalReward);
        rewardIntegralList.add(singleRewardIntegral);
    });
    rewardIntegral.setRewardAssignList(rewardIntegralList);
    return rewardIntegral;
}
```

图 5-22: 积分奖励提交与分配关键代码

图 5-22给出了积分奖励提交与分配关键代码。积分奖励数据提交时，由于在需求提交阶段同时完成，因此只需校验任务状态为 1 即可，随后将控制层传入 RewardSubmit 模型，最后调用积分奖励服务中的 uploadReward 返回数据。积分奖励进行分配时，首先调用任务服务中方法更新当前众测任务状态至奖励分配阶段即 5。然后将控制层传入 RewardIntegral 模型，对于列表中的每位工人，由其对应的测试报告及其激励结果组装成 SingleRewardIntegral 实体，并合成激励结果列表。这里按照第三章设计的可信评分在线众测激励机制来计算激励结果。最后调用奖励服务封装数据后返回。

```
//积分奖励溯源
public RewardCollection getRewardByTaskID(String taskId) {
    RewardCollection result = new RewardCollection();
    RewardIntegral rewardIntegral =
        this.rewardIntegralService.getRewardIntegralByTaskID(taskId);
    List<SingleRewardIntegral> rewardIntegralList = rewardIntegral.getRewardAssignList();
    List<BugRepScoreVO> bugRepScoreVOS = new ArrayList<>();
    rewardIntegralList.forEach(v->{
        TestRepScoreVO testRepScoreVO =
            testReportService.getTestVOByTestId(v.getTestRepId());
        v.setTestRepScore(testRepScoreVO.getTestRepScore());
    });
    result.setTotalReward(rewardIntegral.getTotalReward());
    result.setRewardAssignList(rewardIntegralList);
    return result;
}
```

图 5-23: 积分奖励溯源关键代码

图 5-23给出了积分奖励溯源关键代码。在众测任务中，若状态为已完成，用户可查看其奖励分配的结果信息。在积分奖励溯源中，首先通过任务 Id 调用 rewardService 中的 getRewardByTaskID 方法，获取任务奖励分配结果。rewardIntegralList 为所有工人的奖励分配列表，对于列表中每一个测试报告，系统调用 testReportService 获取报告审核数据。最后将整合的结果包含每位工人奖励分配以及报告得分的信息返回给前端。

5.2 区块链服务设计与实现

区块链基础平台作为 BaaS 架构的数据底层，分为对外服务层和基础设施层两部分，其中服务层主要包含对外提供的数据溯源和数据上链等服务，基础设施层为区块链节点集群等设施。

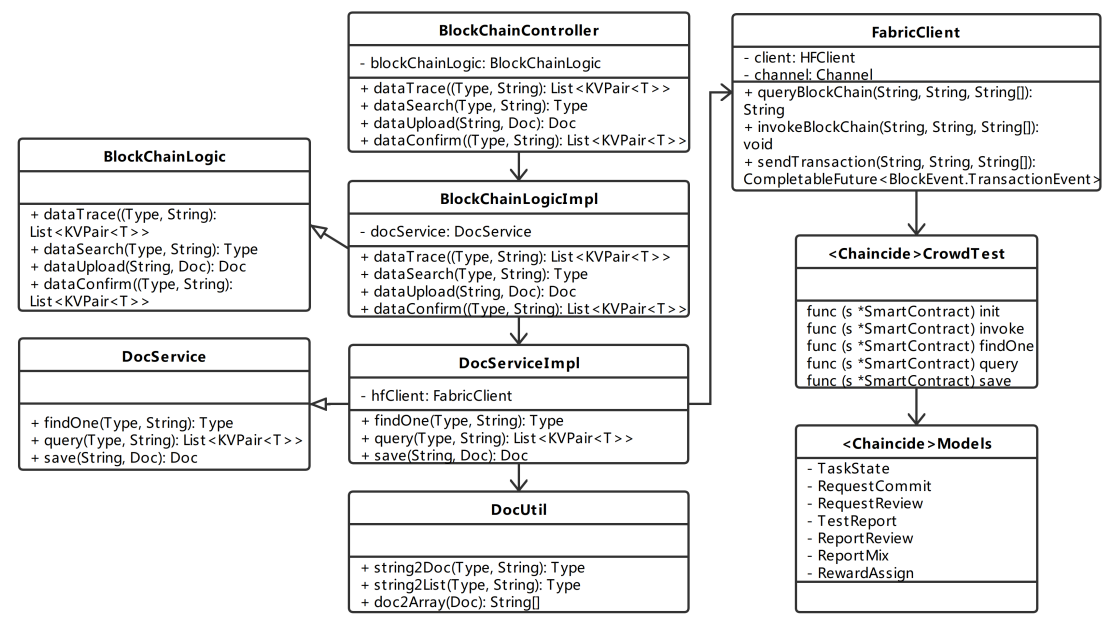


图 5-24: 区块链基础平台核心类图

图 5-24是区块链基础平台核心类图。BlockChainController 对外界提供了数据溯源、数据上链、资产确权等接口。BlockChainLogic 包含了区块链与外部交互数据的处理逻辑。DocServiceImpl 借助 FabricClient 调用智能合约进行区块链操作，具体实现了读取账本数据与存储账本数据的功能模块，其中账本数据的转换由 DocUtil 进行提供。本系统的智能合约是 go 语言写的 CrowdTest，在 Models 中定义了其所依赖的各阶段数据模型。

5.2.1 对外服务层

区块链基础平台基于 Fabric-Java-SDK 构建了对外服务层，该层提供区块链数据的溯源与上链等功能，通过接口形式为其他服务提供屏蔽底层复杂操作。

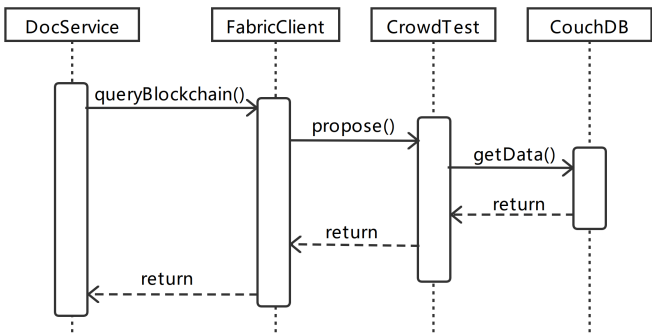


图 5-25: 区块链平台服务层数据溯源顺序图

图 5-25展示了区块链平台服务层数据溯源调用顺序。在链上数据溯源流程中，首先由 DocService 账本服务发起请求调用 FabricClient，然后后者触发本次查询交易，由于是账本读取操作并无写入进而无需共识验证。最后由智能合约根据传入参数直接查询账本数据的映射数据库 CouchDB 并返回数据。

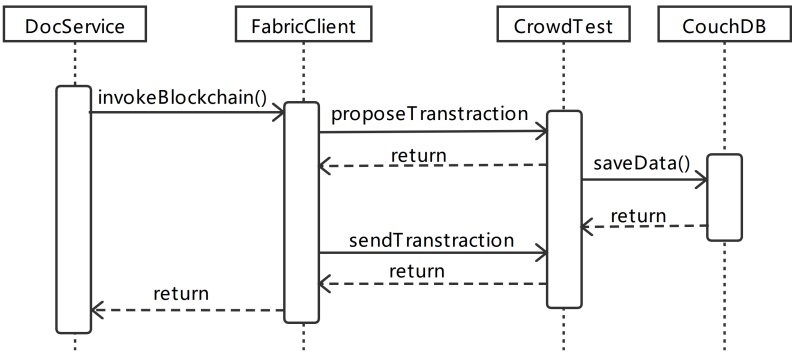


图 5-26: 区块链平台服务层数据上链顺序图

图 5-26展示了区块链平台服务层数据上链调用顺序。在数据上链流程中，首先由账本服务调用 FabricClient 中方法，然后由后者封装好区块后触发写入交易，同时向所有节点通知进行区块验证。若本次交易中节点共识成功，则将调用智能合约的 save 方法，将区块数据存入账本更新；若本次交易中节点共识失败，或是请求中存在非法参数，将停止交易直接退回。

```
public String queryBlockChain(String chaincode, String function, String[] args) throws
Proposal Exception, InvalidArgumentException {
    QueryByChaincodeRequest qpr = client.newQueryProposalRequest();
    ChaincodeID cid = ChaincodeID.newBuilder().setName(chaincode).build();
    //省略组装代码
    Collection<ProposalResponse> res = channel.queryByChaincode(qpr);
    for (ProposalResponse pres : res) {
        //省略异常处理代码
        String stringResponse = new String(pres.getChaincodeActionResponsePayload());
        return stringResponse;
    }
    return null;
}
```

图 5-27: 区块链平台服务层数据溯源关键代码

图 5-27给出了区块链平台服务层数据溯源关键代码。首先，由 client 作为 HFClient 实例，生成查询实体，并添加查询的参数。随后，查询实体交由通道发送给联盟链网络，返回查询结果。


```
public CompletableFuture<BlockEvent.TransactionEvent> sendTransaction(...)
    throws InvalidArgumentException, ProposalException {
    TransactionProposalRequest tpr = client.newTransactionProposalRequest();
    ChaincodeID cid = ChaincodeID.newBuilder().setName(chaincode).build();
    //省略 tpr 组装代码
    Collection<ProposalResponse> responses = channel.sendTransactionProposal(tpr);
    List<ProposalResponse> invalid =
        responses.stream().filter(ProposalResponse::isInvalid).collect(Collectors.toList());
    if (!invalid.isEmpty()) {
        invalid.forEach(response -> logger.error("tx"+response.getTransactionID()+"proposal
            invalid"+response.getMessage()));
        throw new RuntimeException("invalid response(s) found");
    }
    return channel.sendTransaction(responses);
}
```

图 5-28: 区块链平台服务层数据上链关键代码

图 5-28给出了区块链平台服务层数据上链关键代码。在数据上链时，数据写入账本的过程需要由节点发起交易请求，广播至其余节点后由各个节点进行共识验证。首先，由 `HFClient` 生成交易请求，并添加传入参数。随后，向通道发送组装好的交易请求，前者将各节点的响应内容组成列表后返回。遍历响应列表并检查每一个响应，若均有效则将其列表整体发送至通道并写入账本，若存在无效响应则抛出异常。

```
func (s *SmartContract) Init(APIStub shim.ChaincodeStubInterface) sc.Response {
    return shim.Success(nil)
}
func (s *SmartContract) Invoke(APIStub shim.ChaincodeStubInterface) sc.Response {
    function, args := APIStub.GetFunctionAndParameters()
    if function == "findOne" {
        return s.findOne(APIStub, args)
    } else if function == "save" {
        return s.save(APIStub, args)
    } else if function == "query" {
        return s.query(APIStub, args)
    }
    return shim.Error("Invalid Smart Contract function name.")
}
```

图 5-29: 智能合约关键代码

在本章的第一节中，可信众测激励服务端的各个服务均通过本小节介绍的区块链平台对外服务接口（即数据上链、数据溯源接口）来进行实现，而对外的区块链数据接口则过区块链服务层依赖底层 `Chaincode` 智能合约所实现的。

本系统的智能合约关键代码如图 5-29所示。在本文智能合约 CrowdTest 中，定义并实现了多种基本方法：init 方法为 Chaincode 的初始化函数，在首次部署时调用进行初始化；invoke 方法根据传递的具体参数通过调用其他函数来发起交易，实现对账本数据的读写等操作；query 方法实现账本数据关键字读取功能，根据关键字查询包含关键字的所有数据；findOne 方法实现账本数据定位读取功能，根据账本 ID 查询对应的某个键值对数据；save 方法实现账本数据存储功能，后面将进行详细介绍。

下面将详细介绍各个服务是从前端到智能合约层级的执行过程。在数据溯源阶段，众测流程中各个阶段的服务与任务完成后的众测激励服务均是同样的流程：在请求中传入参数后，由服务端进行数据处理，随后将参数发送至区块链平台的数据溯源接口，接着数据溯源接口通过区块链服务 BlockchainService，依赖 FabricClient 来调用智能合约 CrowdTest 中的 query 方法，读取账本数据获取结果。在数据上链阶段，需要将服务分为众测数据与众测激励两部分来分别介绍流程：在众测数据提交阶段，包括任务数据、需求数据、测试报告数据、最终报告数据等，在请求中传入参数后，由服务端进行数据处理，有部分服务这里需要多个查询账本的过程，这里在众测激励部分会介绍，随后返回联盟链模型数据至区块链平台的数据上链接口，接着数据上链接口通过区块链服务来调用智能合约 CrowdTest 中的 save 方法，将联盟链模型数据存入区块链；在积分奖励服务中，奖励提交步骤同上，而奖励分配（即众测激励）则根据流程先要进行一个查询账本数据的步骤（这里和众测数据提交中某些阶段相同），查询过程逻辑与数据溯源阶段一致，根据参数先由服务端调用区块链数据溯源接口，再由区块链服务调用智能合约 query 方法获取账本数据，在查询完成之后才进行数据处理最后封装后调用 save 方法上链。

```
func (s *SmartContract) save(APIstub shim.ChaincodeStubInterface, args []string) sc.  
Response{  
    doctype := args[1]  
    var docAsBytes []byte  
    if doctype == "0" {  
        doc := RequestCommit{Type:doctype, TaskId: args[2] ...}  
        docAsBytes, _ = json.Marshal(doc)  
    } //省略相同存储逻辑  
    APIstub. PutState(args[0], docAsBytes)  
    return shim.Success(nil)  
}
```

图 5-30: 智能合约 save 方法关键代码

图 5-30给出了智能合约中 save 方法的关键代码。Fabric 及其映射数据库 CouchDB 均要求以键值对形式存储数据。因此，在第四章中本文依据众测流程设计了 7 种不同类型的数据存入 Models 中作为联盟链数据模型，分别是 TaskState 众测任务模型、RequestCommit 任务需求提交模型、RequestReview 任务需求审核模型、TestReport 测试报告提交模型、ReportReview 测试报告审核模型、ReportMix 最终报告聚合模型以及 RewardAssign 积分奖励分配模型。智能合约首先将传入参数字符转换为定义的联盟链数据模型，随后通过 json 方法变更为 Byte 实体，随后写入账本完成数据上链。

5.2.2 基础设施层

基础设施层包含了区块链基础平台的所有底层设施，包括 Hyperledger Fabric 联盟链网络、节点集群以及相关功能组件。本系统提供了联盟链网络的部署脚本 docker-compose，各个组件依赖 docker 容器进行运行。下文将详细介绍区块链平台基础设施层的配置与部署。

```
networks:
  custom:
services:
  couchdb:
    container_name: couchdb
    image: hyperledger/fabric-couchdb
    environment:
      - COUCHDB_USER=
      - COUCHDB_PASSWORD=
    ports:
      - "5984:5984"
    networks:
      - custom
  ca.org1.example.com:
    image: hyperledger/fabric-ca:1.4.4
    environment:
      - FABRIC_CA_HOME=/etc/hyperledger/fabric-ca-server
      - FABRIC_CA_SERVER_CA_NAME=ca-org1
      - FABRIC_CA_SERVER_CA_CERTFILE=/etc/hyperledger/fabric-ca-server/...
      - FABRIC_CA_SERVER_CA_KEYFILE=/etc/hyperledger/fabric-ca-server/...
    ports:
      - "7054:7054"
    command: sh -c 'fabric-ca-server start -b admin:adminpw -d'
    volumes:
      - ../network_resources/crypto-config/peerOrganizations/org1.example.com/ca/: ...
    container_name: ca_peerOrg1
    networks:
      - custom
```

图 5-31: CouchDB 与 Fabric CA 组件部署关键代码

Fabric-CA 组件与 CouchDB 均通过该脚本进行部署。如图 5-31所示，脚本定义 couchdb 服务采用其作为状态数据库，配置其镜像，设置其账号等信息，将容器映射到对外开放的服务器端口 5984，并设定其使用的网络为 custom。在 CA 身份认证组件方面，镜像文件与区块链同步，使用 1.4.4 版本，服务器默认端口 7054，通过命令启动，为各个节点提供身份验证功能。通过默认管理员账户 admin 进行新用户的注册与通道的创建。其服务器的私钥和证书文件等将在初始化服务器后生成，之后将由 volumes 将配置映射到本地，包含镜像中的配置及证书文件等。

```
orderer.example.com:
  container_name: orderer.example.com
  image: hyperledger/fabric-orderer:1.4.4
  environment:
    - ORDERER_GENERAL_LOGLEVEL=debug
    - ORDERER_GENERAL_LISTENADDRESS=0.0.0.0
    - ORDERER_GENERAL_GENESIMETHOD=file
    - ORDERER_GENERAL_GENESISFILE=/etc/hyperledger/configtx/genesis.block
    - ORDERER_GENERAL_LOCALMSPID=OrdererMSP
    - ORDERER_GENERAL_LOCALMSPDIR=/etc/hyperledger/msp/orderer/msp
  working_dir: /opt/gopath/src/github.com/hyperledger/fabric/orderer
  command: orderer
  networks:
    - custom
  ports:
    - 7050:7050
  volumes:
    - ../network_resources/config:/etc/hyperledger/configtx
    - ../network_resources/crypto-config/ordererOrganizations/example.com/...
    - ../network_resources/crypto-config/peerOrganizations/... : ...
```

图 5-32: Orderer 节点部署关键代码

Orderer 节点部署关键代码如图 5-32。镜像文件同样采用了 1.4.4 版本，在环境设置中，脚本定义了诸如 orderer 的日志级别、监听地址、创世区块文件的类型与路径、本地 MSP 的 ID 与文件夹等属性。脚本还定义了所在的默认工作目录，并设置了容器启动后的第一条运行命令，即启动 orderer 节点。

图 5-33给出了 Peer 节点部署关键代码。系统按照用户角色设定需求方、工人方与平台方三类对等节点。镜像版本依然是 1.4.4，在环境设置中，脚本定义了 peer 节点的 ID、访问地址与监听地址、智能合约日志级别、账本映射数据库、TLS 相关参数等属性，其中 GRPC 通信端口为 7051，事件监听端口为 7053。脚本还定义了默认工作目录，并设置了启动后第一条启动 Peer 节点的命令。最后设置了其依赖，包括 Orderer 节点与 couchdb 数据库。

```
peer0.org1.example.com:
  container_name: peer0.org1.example.com
  image: hyperledger/fabric-peer:1.4.4
  environment:
    - CORE_PEER_ID=peer0.org1.example.com
    - CORE_CHAINCODE_LOGGING_LEVEL=DEBUG
    - CORE_LEDGER_STATE_STATEDATABASE=CouchDB
    - ...
  working_dir: /opt/gopath/src/github.com/hyperledger/fabric
  command: peer node start
  ports:
    - 7051:7051
    - 7053:7053
  networks:
    - custom
  volumes:
    - /var/run:/host/var/run/ ...
  depends_on:
    - orderer.example.com
    - couchdb
```

图 5-33: Peer 节点部署关键代码

5.3 系统界面展示

本文实现了面向可信众测激励的区块链服务系统。系统服务于众测各方参与者，包括需求方、工人方与平台方。系统界面如下文所示。



图 5-34: 需求方主页界面

需求方主页界面如图 5-34所示。界面整体分为三部分，上面是区块链数据，中间是个人信息，下面是任务列表。区块链部分有节点、高度、交易等参

数信息，若想查询更详细数据可进入 Explorer 查看。个人信息部分还包含其参与的任务与需求的数据统计。任务列表部分包含其参与各任务，若已结束则可追溯最终报告来源。列表中可以验证链上哈希与平台是否相同，或是直接查看当前需求的区块链数据以确保数据的可信，也可以直接查看该任务总积分奖励具体分配的信息，还能够追溯当前任务的进展流程。

数据溯源 / 溯源详情 / 最终报告信息

任务名称: QQ音乐众测任务	最终报告: QQ音乐最终报告	融合时间: 2021-03-20 14:34:25	融合者: 李欣
数据验证:	区块链信息:		

缺陷报告	众测工人	报告得分	评审人	数据验证	区块链信息
页面闪退	张三	10	刘伟维		
无法重播歌曲	王飞	8	刘伟维		
无法添加评论	赵四	9	李东		

图 5-35: 需求方报告聚合溯源界面

需求方报告聚合溯源界面如图 5-35所示。需求方可查询最终报告包含的缺陷报告来源列表。对于每一份缺陷报告，系统展示了其提交的工人、得分、审核人等，便于后续出问题能够追责。

节点数 3

区块高度 125

交易笔数 125

链码数 1

个人信息

角色: 工人方
单位:
手机:
邮箱:

ZhangSan

测试报告审核分析

缺陷报告采纳分析

已审核报告 2

未审核报告 2

任务名称	测试报告	当前状态	提交时间	数据验证	区块链信息	奖励分配	数据溯源
QQ音乐众测任务	QQ音乐测试报告-张三	已审核	2021-02-22				
QQ视频众测任务	QQ视频测试报告-张三	已审核	2021-02-24				

图 5-36: 工人方主页界面

工人方主页界面如图 5-36所示。工人方界面中部与需求方不同，展示了测试报告审核与缺陷报告采纳的数据统计。参与的任务列表中，包括了任务及其对应测试报告、当前状态等字段，工人方同样能够和需求方一样验证数据哈希值是否一致，并且溯源测试报告，查询激励结果等。



图 5-37: 工人方测试报告溯源界面

工人方测试报告溯源界面如图 5-37所示。任务溯源以时间线形式展示，若已完成则可点入下个阶段。工人方追溯任务时，仅有需求提交、报告提交与审核阶段的数据对其可见。工人方能够查询到需求提交时的需求信息；追溯到该任务下所有测试报告的提交信息，该信息以链式结构展示；查询到该任务自己每个缺陷报告的得分信息。

数据溯源 / 奖励分配信息

任务名称: QQ音乐众测任务	任务需求方: 陈强	完成时间: 2021-03-20 14:34:25	总奖励积分: 100				
数据验证: 	区块链信息: 						
测试报告	 众测工人	 提交时间 	 审核得分	积分奖励	数据验证	区块链信息	
 QQ音乐测试报告-张三	张三	2021-02-22	8	36			
缺陷报告	报告得分		评审人		采纳结果		
页面闪退	10		刘伟维		 采纳		
无法点击收藏	6		刘伟维		 未采纳		
 QQ音乐测试报告-王飞	王飞	2021-02-23	7	32			
 QQ音乐测试报告-赵四	赵四	2021-02-23	7	32			

图 5-38: 工人方激励结果溯源界面

工人方奖励分配界面如图 5-38所示。当任务完成后，工人可点击奖励分配查看该任务所有参与者的奖励信息。上方展示该任务基本信息，包括任务总积分，对该任务积分情况可以进行数据验证。下方列表将展示每位参与工人的具体测试报告总得分，以及根据报告相应获得的奖励积分。对于每位工人，其测试报告可点击展开框查看内含所有缺陷报告的地方情况，从而便于信息公开与后续核对。

5.4 本章小结

为完成面向可信众测激励的区块链服务系统的开发，本章将系统分为服务端与区块链基础平台两部分，分别进行具体的介绍。服务端包含任务、需求、报告、奖励等模块，而区块链平台包含服务层与设施层。本章先给出各个模块的类图与顺序图，动静结合来阐述详细的设计思路，再给出代码部分，介绍具体实现的步骤，在区块链设施层还给出了部署脚本。最后，通过三方的主要界面截图梳理系统整体功能。

第六章 系统测试与案例分析

本章对系统进行测试及真实案例的分析。本章先描述测试环境，再根据第四章的功能与非功能需求设计测试用例，测试涉及功能、性能与安全性三个方面。完成基本测试后，系统结合真实案例，接入众测比赛数据并分析结果。

6.1 测试环境

系统各个模块的硬件环境与软件环境如表 6-1所示。本系统遵循分层设计原则划分为客户端、可信众测激励服务端以及区块链基础平台，下文将分别对三个部分进行测试。

表 6-1: 系统测试环境

模块	硬件环境	软件环境
Web 客户端	腾讯云服务器， Ubuntu16.04，1 台	Vue 3.0， Nginx 1.10.2
可信众测激励 服务端	腾讯云服务器， Ubuntu16.04，1 台	JDK 1.8.0， Maven 3.3.9， SpringBoot 2.0.2
区块链 基础平台	腾讯云服务器， Ubuntu16.04，2 台	Hyperledger Fabric 1.4.4， Docker 18.06.10

客户端与可信众测激励服务端部署在同一台云服务器上，型号为 ecs.c1.large，腾讯云服务器，操作系统为 Ubuntu16.04。客户端前端界面采用 Vue 框架实现，使用蚂蚁提供的开源 ant-design 组件协助开发，Nginx 作为高性能的反向代理服务器，实现了客户端与服务端之间的负载均衡。可信众测激励服务端基于版本 1.8.0 的 JDK 环境，采用 SpringBoot 框架进行开发，版本为 2.0.2，开发完成后的部署采用 Maven 打包服务。

区块链基础平台的排序与对等节点分别在两台云服务器上部署，通过 GRPC 实现节点间的通信，采用版本为 1.4.4 的 Hyperledger Fabric 开源联盟区块链框架，平台基础设施层组件运行与 Docker 容器中，版本为 18.06。

6.2 测试指标

表 6-2给出了系统测试评价指标，包括并发用户数、响应时间、吞吐量、事物成功率及资源使用率。

表 6-2: 系统测试评价指标说明

评价指标	相关说明
并发用户数	同时向系统发送请求的用户数量
响应时间	从客户端发出请求到接受到服务端响应的的时间
吞吐量	单位时间内系统处理的请求数量
事务成功率	单位时间内系统通过测试的成功比率
资源使用率	运行程序时 CPU 资源占用率、内存使用率、磁盘 I/O 等

测试指标中，并发用户数指在同一时段与服务器交互的在线用户数量；响应时间描述请求发出到返回的时长，可直观地反映系统的快慢；吞吐量指在一次网络传输过程中数据量的总和，反映了服务器能承受的压力 [56]；事务成功率指系统在某时间段内成功完成定义事务的个数，反映了系统的处理能力；资源使用率则与硬件资源的消耗直接相关。

6.3 测试设计

6.3.1 功能测试设计

功能测试的目的是保证系统功能满足需求，其过程为黑盒的，不关心代码的具体内容，关注于项目的主要功能。本节根据此前给出的用例图中所有用例，设计了对应功能的测试，分为众测数据传输、众测流程溯源、众测任务溯源、众测报告溯源、众测奖励分配、众测奖励查询六大模块。

众测数据传输测试用例如表 6-3所示。该用例用来模拟由众测平台实时进行数据传输，结果接入可信众测激励系统的流程。首先由众测平台调用众测数据传输接口并实时发送众测来源数据，随后由系统接收到数据并对其处理，这里实际上系统也对处理完的数据调用区块链基础平台的数据上链接口进行存储，接着在区块链平台内由该参与方节点触发交易并广播，若共识成功后账本数据写入，最后返回传输结果。本用例期望系统在实时传输中，系统能接入正确数据，区块链平台迅速处理并写入数据。

表 6-3: 众测数据传输测试用例

测试编号	TC1
测试功能	众测数据传输
测试目标	激励系统能实时接入众测平台数据并存入区块链
测试步骤	1. 调用接口发送数据至激励系统以模拟接入流程； 2. 可信众测激励系统采集数据并整理，随后提出上链的交易； 3. 节点共识通过后，在账本内更新数据。
期望结果	1. 能够正常调用众测数据传输接口； 2. 可信众测激励系统能采集到正确数据并及时响应共识结果； 3. 共识成功后数据能够正常上链。

需求方众测任务溯源测试用例如表 6-4所示。用例模拟需求方登录系统，查看其参与的任务并进行进度溯源、最终报告来源以及数据验证。在预期情况下，系统展示需求方参与的所有任务，并展示任务进度、最终报告来源以及数据验证信息。

表 6-4: 众测任务溯源测试用例

测试编号	TC2
测试功能	众测任务溯源
测试目标	需求方可查看任务进度与其需求提交、审核及最终报告数据
测试步骤	1. 需求方通过账号密码登录； 2. 需求方主页需显示其参与所有任务； 3. 进行任务溯源，点击后测试是否跳转至任务进度溯源界面； 4. 对已聚合任务来源追溯，点击后测试是否跳转至最终报告聚合列表界面； 5. 对需求文档进行验证，点击后测试验证结果是否由弹出窗展示。
期望结果	1. 账号密码正确成功登录； 2. 展示当前参与的任务列表； 3. 进入任务溯源界面并展示该任务进度溯源； 4. 展示最终报告来源详细信息； 5. 展示数据验证信息。

工人方众测报告溯源测试用例如表 6-5所示。用例模拟工人方登录系统，查看其参与的任务并进行众测报告溯源，查询测试报告提交数据、缺陷报告审核结果与数据验证结果。在预期情况下，系统展示需求方参与的所有任务，并

表 6-5: 众测报告溯源测试用例

测试编号	TC3
测试功能	众测报告溯源
测试目标	工人方查询任务进度与其测试报告提交、审核数据
测试步骤	1. 工人方通过账号密码登录； 2. 工人方主页需显示其参与所有任务； 3. 进行任务溯源，点击后测试是否跳转至任务进度溯源界面； 4. 对缺陷报告审核追溯，点击后测试是否跳转至缺陷报告审核列表界面； 5. 对测试报告进行验证，点击后测试验证结果是否由弹出窗展示。
期望结果	1. 账号密码正确成功登录； 2. 展示当前参与的任务列表； 3. 进入任务溯源界面并展示该任务进度溯源； 4. 展示缺陷报告审核详细信息； 5. 展示数据验证信息。

展示已提交测试报告列表，同时对已审核报告展示缺陷报告审核数据，工人也能获取验证结果。

表 6-6: 众测流程溯源测试用例

测试编号	TC4
测试功能	众测流程溯源
测试目标	平台方查询任务的当前进度与过程数据
测试步骤	1. 平台管理员通过账号密码登录； 2. 平台方主页需显示当前所有任务众测任务列表； 3. 进行任务溯源，点击后测试是否跳转至任务进度溯源界面； 4. 对测试报告追溯，点击后测试是否跳转至测试报告提交列表界面； 5. 对缺陷报告追溯，点击后测试是否跳转至缺陷报告审核列表界面； 6. 对报告聚合追溯，点击后测试是否跳转至最终报告来源列表界面。
期望结果	1. 账号密码正确成功登录； 2. 显示系统中所有众测任务的列表； 3. 进入任务溯源界面并展示该任务进度溯源； 4. 跳转到测试报告提交溯源页面； 5. 跳转到缺陷报告审核溯源页面； 6. 跳转到报告聚合溯源页面。

平台方众测任务溯源测试用例如表 6-6所示。用例模拟平台方登录系统后，追溯平台中所有任务的当前进度与过程数据。期望的结果是，由系统展示当前所有任务组成的列表，在选取某个任务后，进入该任务的溯源界面。当点击“全部测试报告”时，跳转到测试报告提交溯源列表页面，其中包含该任务下全部测试报告及其缺陷报告。当点击“全部缺陷报告”时，跳转到缺陷报告审核溯源列表页面，列表中包含全部缺陷报告的得分结果。当点击“最终报告”时，系统跳转到最终报告来源列表界面，其中将会展示最终报告全部来源信息，包括其聚合的所有缺陷报告列表。

表 6-7: 众测奖励分配测试用例

测试编号	TC5
测试功能	众测奖励分配
测试目标	工人方由系统分配获得奖励报酬
测试步骤	1. 工人方通过账号密码登录； 2. 在参与任务列表中选取某项已完成的任务，点击跳转至奖励查询界面； 3. 查看该任务中，各工人的报告得分是否属实，获得的奖励分配结果是否正确。
期望结果	1. 账号密码正确成功登录； 2. 成功跳转至该任务的奖励查询界面； 3. 显示任务中工人的报告得分和奖励分配结果信息，并数据属实且结果经过核算无误。

工人方众测奖励分配测试用例如表 6-7所示。用例模拟工人方用户登录系统，在选定某项已完成任务后，展示该任务中各工人的报告得分信息与获得的奖励分配结果信息。在预期情况下，工人方登录后能够查询到已完成任务中各工人的报告得分与奖励分配结果信息，报告得分与区块链中数据核对后属实，奖励分配结果经计算后无误。

需求方、工人方、平台方众测奖励查询测试用例如表 6-8所示。用例模拟三方用户登录系统，在选定某项任务后，展示该任务初始总积分信息，并展示任务对应具体积分分配列表。在预期情况下，三方登录后能够查询到每项任务的积分奖励信息，包括任务初始总积分情况，以及若该任务已完成激励后的具体每位工人的奖励分配情况。

表 6-8: 众测奖励查询测试用例

测试编号	TC6
测试功能	众测奖励查询
测试目标	需求方、工人方、平台方追溯某任务的奖励积分分配结果
测试步骤	1. 需求方、工人方或者平台方通过账号密码登录； 2. 在参与任务列表中选取某项任务，查看是否跳转至积分奖励界面； 3. 查看界面是否展示任务初始总积分信息； 4. 对于已完成激励的任务，点击“分配详情”按钮，查看是否弹出该任务具体对应每位工人的分配信息。
期望结果	1. 三方账号密码正确成功登录； 2. 显示该用户参与的众测任务列表； 3. 显示任务初始总积分信息； 4. 显示任务积分分配列表。

6.3.2 性能测试设计

本系统设计了对可信众测激励服务端接口与区块链基础平台的性能测试。服务端接口性能测试通过 JMeter^①工具来进行。区块链性能测试通过开源的 Hyperledger Caliper^②基准框架来进行。

1) 服务端接口性能测试设计

表 6-9: 系统服务端关键接口

接口名称	接口功能
查询众测任务	查询当前所有众测任务
查询需求提交	根据需求方标识查询其提交的需求
查询需求审核	根据需求标识查询其需求的审核数据
查询测试报告	根据工人方标识与任务标识查询其提交的测试报告
查询报告提交	根据任务标识查询其包含的测试报告列表
查询报告审核	根据任务标识查询其包含的缺陷报告审核列表
查询最终报告	根据任务标识查询其最终报告数据
查询报告聚合	根据任务标识查询其最终报告来源列表
查询任务奖励	根据任务标识查询任务初始总奖励积分
查询奖励分配	根据任务标识查询其所有工人及其奖励分配结果列表

^①JMeter. <https://jmeter.apache.org/>
^②Hyperledger Caliper. <https://hyperledger.github.io/caliper/>

本文待测试的所有服务端接口如表 6-9所示。本系统服务端接口较多，主要分为各类服务的提交与查询两种。考虑到查询接口由于返回数据量大导致相应速度慢，且提交接口请求参数过多测试中不便于填写，本节着重测试查询接口的性能，在第五节案例分析中通过真实数据再测试提交接口的性能。测试人员可通过 JMeter 创建线程以调用接口从而观察应答效率，这里的线程相当于不变的虚拟用户。本文创建了线程组，随后对其添加了 HTTP 请求，地址为系统服务端 API，用户数量设置为 40，在 2 秒内各线程同步请求响应，最后将重复上述过程，共进行 2 轮测试。

2) 区块链性能测试设计

表 6-10: 区块链平台智能合约关键方法

方法名称	方法功能
save	区块链账本中添加新数据
findOne	根据账本标识查询数据
query	模糊查找涵盖指定关键字的所有数据

本文对智能合约 crowdTest 的三个主要方法进行性能测试，如表 6-10所示。区块链性能测试采用 Caliper 工具进行，其支持 Fabric 等多种联盟链，测试人员在设置并发数后执行智能合约方法，通过工具可查看吞吐量、延迟等性能参数信息。考虑到实际场景中可能存在的大量并发数场景，本文设置了 5 轮测试查看性能情况，将并发请求数从 50tps 到 250tps，每次增加 50tps 来进行测试。

6.3.3 安全测试设计

在面向可信众测激励的区块链系统中，数据上链与溯源均基于智能合约完成。若合约存在漏洞，则用户会对数据的真实性产生质疑，进而影响众测激励的可信性。Chaincode Scanner^①是针对 Fabric 智能合约的安全性分析工具，本文通过此工具来检查智能合约漏洞。使用时需要将代码上传到公共存储库并输入其路径，在完成代码分析后，将给出安全性测试报告。

Chaincode Scanner 中面临检查的合约漏洞如表 6-11所示。可见，账本数据存取时需注意避免幻读与脏读等操作，谨慎使用全局变量；智能合约方法运行时需注意使用前先验证参数，避免并发操作等。

^①Chaincode Scanner. <https://chaincode.chainsecurity.com/>

表 6-11: 区块链平台待检测智能合约漏洞

漏洞名称	漏洞描述
账本数据幻读	多次读取账本时产生幻读
变量写后读操作	可能在写操作相同变量之后脏读到旧值
全局变量使用	函数方法中不能依赖于全局变量
参数使用前未验证	输入可能为空值，需输入后检查
代码中使用并发	可能由于并发导致异常

6.4 测试结果分析

6.4.1 功能测试结果分析

表 6-12: 功能测试用例结果

需求用例	测试用例	测试结果
UC1	TC1	通过
UC2	TC2	通过
UC3	TC3	通过
UC4	TC4	通过
UC5	TC5	通过
UC6	TC6	通过

测试人员严格按照测试用例步骤，按顺序对照需求用例并执行所有测试用例，其次与预期结果进行验证后记录下各用例的测试情况。根据此前测试用例中的描述，系统功能测试执行结果如表 6-12所示，本次所有测试用例结果全部通过。综上所述，本系统按照第四章用例分析实现了其全部功能需求。

6.4.2 性能测试结果分析

1) 服务端接口性能测试结果

表 6-13给出了服务端接口性能测试结果。在 200 次请求下，各查询接口均能正确接收与应答。响应时间方面，查询接口平均 493 毫秒，中位数 490 毫秒，异常率为零，吞吐量上平均每秒能处理 15 个交易。经检验，服务端各个查询接口均正常响应，高并发下性能测试结果良好。

表 6-13: 服务端接口性能测试结果

接口名称	请求总数	平均值	中位数	最小值	最大值	成功率	吞吐量
查询众测任务	200	542	534	496	588	100%	12.6/s
查询需求提交	200	499	492	425	534	100%	15.8/s
查询需求审核	200	493	496	458	521	100%	16.7/s
查询测试报告	200	504	509	476	529	100%	13.0/s
查询报告提交	200	548	532	504	595	100%	11.3/s
查询报告审核	200	516	508	492	543	100%	12.1/s
查询最终报告	200	418	422	396	442	100%	17.4/s
查询报告聚合	200	478	462	434	510	100%	14.6/s
查询任务奖励	200	411	416	388	432	100%	18.1/s
查询奖励分配	200	524	528	489	552	100%	13.2/s
总计	2000	493	490	388	595	100%	14.6/s

2) 区块链性能测试结果

表 6-14: 区块链平台智能合约方法性能测试结果

方法名称	平均延迟	最小延迟	最大延迟	成功数	失败数	吞吐量
save	0.18s	0.05s	0.41s	1000	0	50tps
findOne	0.10s	0.04s	0.25s	1000	0	50tps
query	0.12s	0.02s	0.28s	1000	0	50tps

表 6-14给出了区块链平台智能合约性能测试结果。并发请求数设定为 50tps，即每秒发起 50 笔交易，结果三种方法均能成功完成。在延迟时间方面，由于 save 方法发起上链交易须节点共识，因此时间较长；其余两种查询方法无需共识，因此时间较短。智能合约方法吞吐量至少达到 50tps，性能良好。

表 6-15: 区块链平台资源占用测试结果

节点名称	平均 CPU 率	最大 CPU 率	平均内存值	最大内存值
peer0.org1.example.com	29.46%	48.34%	138.3MB	140.8MB
peer0.org2.example.com	29.35%	44.32%	117.6MB	120.3MB
orderer.example.com	14.24%	21.80%	27.5MB	28.8MB
ca.org1.example.com	0.00%	0.00%	6.4MB	6.4MB
ca.org2.example.com	0.00%	0.00%	19.1MB	19.1MB

区块链资源占用结果如表 6-15所示。org1 和 org2 为对等节点，需验证与存储数据导致资源使用较多，约占用了 29% 的 CPU 与 120MB 的内存；orderer 节点为排序节点，只需交易排序导致使用较少，约占用了 14% 的 CPU 与 28MB 的内存；CA 节点为身份验证节点，资源消耗更小，几乎可忽略。

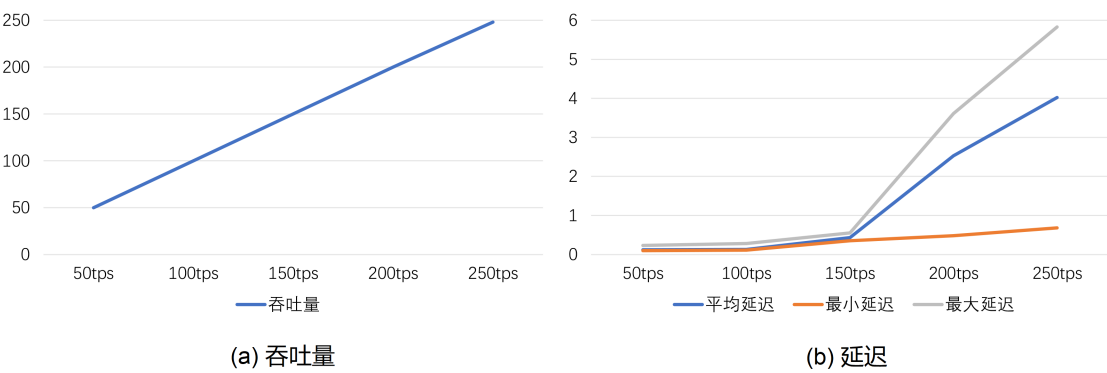


图 6-1: 区块链平台智能合约高并发性能测试结果

区块链智能合约吞吐量及延迟的高并发性能测试结果如图 6-1。表 6-14显示 50tps 的结果，将并发数从 50tps 依次提高至 250tps，智能合约吞吐量由 50tps 增加至 248tps，各方法平均延迟由 0.11 秒增加至 4.02 秒。综上所述，本系统的吞吐量基本达到高并发要求，延迟在 150tps 以下较低，在 200tps 以上略高。

6.4.3 安全测试结果分析

表 6-16: 区块链平台智能合约安全测试结果

漏洞名称	检测数量
账本数据幻读	0
变量写后读操作	0
全局变量使用	0
参数使用前未验证	0
代码中使用并发	0

表 6-16给出了智能合约安全测试结果。通过 Scanner 扫描后，测试结果表明账本数据幻读、变量写后读操作、全局变量使用、参数使用前未验证、代码中使用并发中均无安全漏洞，智能合约安全性良好。

6.5 案例分析

本文考虑到系统未来将投入于真实众测场景使用，为确保在生产环境下的功能与性能，本文将接入慕测平台月度赛的真实数据，模拟众测过程以进行系统案例测试。接入比赛为 2020 年 4 月慕测月度赛，时间 25 日下午 1 至 5 点。

表 6-17: 众测月度赛数据统计

类型	描述	来源	数量
待测软件	咕咚翻译 APP 软件	需求方	1
需求文档	咕咚翻译软件	需求方	1
测试报告	由缺陷报告组成	工人方	64
缺陷报告	包含文字与图片	工人方	408
最终报告	由缺陷报告聚合而成	平台方	1

接入的众测月度赛数据统计如表 6-17所示。本次比赛需求方为咕咚翻译，待测软件为咕咚翻译 APP 软件，需求文档也由需求方提供；由于有 64 位学生也就是测试工人参与，因此测试报告总共有 64 份；上述所有的测试报告包含的缺陷报告累计总数为 408 份，包含文字与图片；仅一份最终报告，由缺陷报告聚合而成，由平台方交付给需求方。



图 6-2: 众测月度赛数据示例

众测月度赛数据示例如图 6-2所示。图 (a) 为需求方提交的需求文档，描述了软件信息与测试的各项要求。图 (b) 为工人方提交的缺陷报告，含有缺陷标识、分类、优先级等字段进行描述。图 (c) 为平台方评价缺陷报告的结果，根据所有工人的得分进行后期的奖励分配。图 (d) 为平台方聚合的最终报告，选取了部分优质缺陷报告进行聚合，列表中展示被选取的报告部分字段的信息。在模拟了四小时按顺序接入上述众测月度赛数据后，任务接近完成，随后系统存储众测数据并自动进行了众测激励。下面将展示部分系统界面与结果分析。



图 6-3: 平台方数据统计界面

平台方数据统计界面如图 6-3所示，平台方主页点击“数据统计”即可跳转至本界面。众测月度赛中 64 位学生的测试报告与其缺陷报告均成功导入系统，交付的最终报告同样成功接入，选取了 18 份缺陷报告进行聚合。经检验，众测数据接入无误，其功能在真实案例环境下仍保持稳定。

数据溯源 / 奖励分配信息

任务名称: 4月份月度赛	任务需求方: 慕测科技	完成时间: 2020-04-25 17:05:00	总奖励积分: 1000			
数据验证:	区块链信息:					
测试报告	众测工人	提交时间	审核得分	积分奖励	数据验证	区块链信息
+ QQ音乐测试报告-王...	王...	2020-04-25 13:34:26	4	14		
+ QQ音乐测试报告-王...	王...	2020-04-25 13:48:54	4	14		
+ QQ音乐测试报告-王...	王...	2020-04-25 13:52:14	5	17		
+ QQ音乐测试报告-王...	王...	2020-04-25 14:04:34	4	14		
+ QQ音乐测试报告-王...	王...	2020-04-25 14:09:17	5	17		
+ QQ音乐测试报告-王...	王...	2020-04-25 14:48:10	6	21		
+ QQ音乐测试报告-王...	王...	2020-04-25 14:51:15	6	21		

图 6-4: 平台方奖励分配界面

众测数据接入功能正常，下面将检验系统中最主要的众测激励功能。平台方主页点击“奖励分配”后跳转至激励界面，如图 6-4所示。列表展示了本次任务中所有工人与其对应测试报告的得分以及获得的奖励分配积分情况，点击展开框可查看每个缺陷报告的具体得分。本次任务设定总积分为 1000 分，经检验与核对，列表中 64 位参与者均确实按照各自报告评分在总分中的占比获得了积分奖励，激励结果在二次计算后均正确无误，分配过程严格按照第三章设计的可信评分在线众测激励机制，鉴于该方法已有研究证明有效，进而本系统采用该方法实现了激励的有效性。对于每条激励结果，每位工人都能够追溯完整流程中各个阶段的数据并进行验证以确保真实性，如报告审核数据与分配数据，点击“数据验证”进行区块链数据查看及与平台的核对，从而确保激励流程公开透明，实现了激励的可信性。真实案例数据的测试证明了系统在生产环境下众测数据接入、可信激励等功能仍保持高可用。

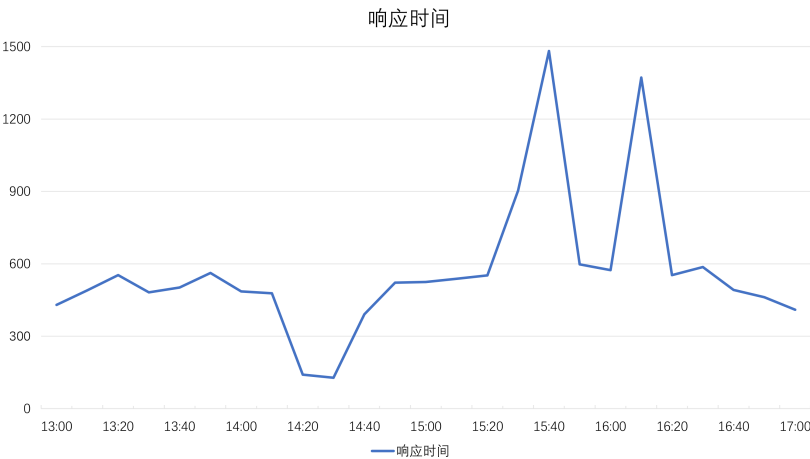


图 6-5: 众测月度赛响应时间性能统计

前一节性能测试聚焦于服务端的查询接口，这里通过真实案例数据测试整个系统所有服务的响应时间，包括此前不便于测试的提交接口等。如图 6-5所示，系统平均花了约 500 毫秒的时间进行响应。系统响应时间结果随着时间段一直变化，如 16 点左右由于大量工人提交报告导致负载上升，15:38 时段一度达到最高响应时间，接近 1500 毫秒,16:10 左右为次高响应时间。整体来看系统能够在高并发下 2 秒以内及时响应，真实环境下依然保持较高性能。

本次众测月度赛中系统 CPU 占用率性能统计如图 6-6所示。众测数据接入期间，系统平均占用 3% 的 CPU 资源。CPU 占用率结果也一直随着时间段在变化，在 16 点左右同样负载增加达到占用率顶峰，最大占用率一度达到约 23%，但此时仍在可用范围之内，随后逐渐下降并恢复至正常。

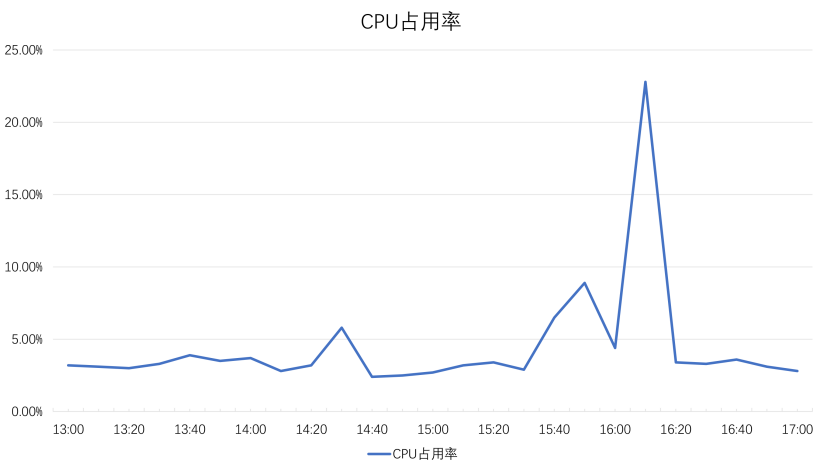


图 6-6: 众测月度赛 CPU 占用性能统计

综上所述，本系统在模拟众测月度赛中，历时四小时定时接入真实案例数据，并对结果进行了详细验证与分析。结果证明，本系统在真实高并发环境下，众测数据接入、激励等功能均保持可用，运行结果均正确无误，未有异常；响应时间与 CPU 占用率等参数虽然在高峰期偏高但仍可控，性能方面整体稳定，基本满足预期要求。

6.6 本章小结

本章对面向可信众测激励的区块链系统进行了多方面的测试与真实场景的案例分析。首先介绍本章测试的一些基本情况，比如测试环境，评估指标等；然后按照第四章需求分析设计了功能测试用例，根据系统技术栈设计了服务端与区块链的性能测试、智能合约的安全测试；针对上述测试内容，本文进行测试并列表给出了结果，数据表明系统在功能、性能、安全性上均未出现问题，能够满足用例需求；最后接入真实场景的案例数据模拟众测进行测试，通过实际众测比赛数据来再次检测高并发下系统的功能的完整性与性能的稳定性。

第七章 总结与展望

7.1 总结

在众包测试中，为了吸引更多工人报名参与任务，同时也为了鼓励工人提高测试质量从而更好地完成任务，平台常常会对其进行激励。然而当前众测平台中的激励存在一些问题。一方面是当前平台常见的激励机制往往效果不佳，劳动力评估方式不科学或是分配方式不公平导致无法保障工人权益。另一方面是平台中的报告与其得分情况往往作为激励的参考，而此类关键性参照数据通常由传统数据库进行中心化管理，容易造成单点攻击等安全性问题或者是数据欺诈等真实性问题。本文结合反向拍卖模型设计了可信评分在线众测激励机制来解决众测激励中的有效性问题，采用与区块链相结合的存储方式来解决众测激励中的可信性问题，并根据该解决方案设计了相应激励流程。本文利用区块链多节点的特性将众测流程中各个参与方视为节点，通过在线评分激励机制进行劳动力公正评估与分配，解决了分配方式不公平以致激励效果不佳的问题；通过区块链的去中心化特性与不可篡改性进行分布式可靠存储，解决了第三方数据真实性存疑的问题；通过区块链的溯源特性进行劳动力资产的全链路确权，解决了激励过程难以公开透明的问题。

本文将需求方、工人方、平台方三方视为节点，构建区块链基础平台并应用于可信众测激励。本文通过智能合约实时将众测流程中数据导入区块链平台作为激励参照数据，发起交易请求，由各个节点共识通过后才能追加写入账本。当执行奖励分配时，由智能合约先通过数据溯源从账本中追溯出激励所依据的众测数据信息如报告及其评分结果，再通过在线评分激励机制完成众测激励并将结果存入区块链平台以记录完成流程；当执行奖励查询时，直接通过区块链账本来获取可信激励结果数据。面向可信众测激励的区块链服务系统采用 BaaS 架构，共分为三层：区块链基础平台分为服务层和基础设施层，服务层封装对外提供的接口，包含数据上链、数据溯源和资产确权三大功能，基础设施层则是采用了开源联盟链 Hyperledger Fabric 框架实现智能合约、CA 认证等功能并部署成为多节点集群；可信众测激励服务端分为用户服务和众测任务服务

两大类，用户服务包括用户登录与信息，而众测任务服务包含任务需求服务、测试报告服务、最终报告服务以及积分奖励服务；系统客户端分为两部分，前端展示页面包括三方用户的系统界面，还提供了 Fabric Explorer 的区块链浏览器以查询相关区块链信息。

最后，本文对系统进行了功能、测试与安全测试。测试结果表明，系统实现了众测激励等功能，能够在有效时间内响应用户的请求，吞吐量达到 250tps，在高并发下仍然具有良好的可用性，且智能合约安全性较高，系统总体情况满足预期的要求。同时，本文接入众测月度赛的真实案例数据来模拟众测流程，案例分析结果同样可证明系统在大量负载下能保持良好性能，并且经检验，确实所有工人均根据报告评分分配的奖励，激励数据无误且公开透明，足以保障其权益并带动其积极性。

7.2 展望

本系统目前仍存在部分不足之处，后期希望能继续进行改进与拓展，主要集中在以下三个方面：

第一，对众测流程中的需求方也进行激励。当前众测激励系统的激励对象主要为工人方，所考虑的也是工人的参与积极性。然而，需求方作为众测任务的提供者，也应该是众测激励对象，需要鼓励其积极参与众测任务。后期希望能通过调研现有机制与方式，在众测平台中增加对需求方激励的形式，形成需求方激励体系，通过积分返还或者其余奖励的手段，鼓励需求方积极发布测试任务，也吸引更多新需求者的加入。

第二，为区块链基础平台提供更完善的图形化管理界面。当前的区块链信息主要通过 Fabric 开源的 Explorer 展示，数据较为固定，后期考虑实现定制化数据展示，并显示完整区块链路。对于完整激励过程，同样考虑可视化，在系统中以链路形式展示，增加用户信任度。

第三，对当前平台的激励机制继续进行改进。当前平台结合反向拍卖模型设计了可信评分在线众测激励机制来实现激励的有效性，然而整个流程是简化版的反向拍卖模型，一方面由需求方事先定价导致总奖励固定，激励力度不够诱人，可设置追加奖励提高总金额；另一方面在积分分配过程中难免由于四舍五入造成微小积分差异，对于分配的算法需再次改进以做到更精准地根据得分进行分配，或者在结果计算后要求需求方对总奖励的差额进行多退少补。

致 谢

时光飞逝，一眨眼两年的研究生生涯就要到达终点。在两年的研究生生涯中，我经历颇多，也收获良多。遥想两年前的我本科毕业，带着美好的憧憬踏入校园，在两年的研究生学习中，一方面继续我的专业知识学习，从学院的课程到实验室区块链小组的研究都让我不断丰富知识；另一方面通过面试和实习，让我在实际动手能力方面有了锻炼，也意识到自己的不足，这些经历都在正式踏入社会之前提醒我要继续努力，完善自我。

当然，在毕业季我也要对所有帮助过我的人一一致谢。首先，要感谢的是我的导师陈振宇老师与实验室区块链的小组所有其他老师们。感谢陈老师作为导师在整个研究生期间对于我们的指导，在学术方面或是在毕业论文方面，老师都抽空和我进行交流并给出帮助和建议。而实验室与小组内其余的指导老师，在此前的项目、毕设以及论文方面也都给予我极大的帮助，监督与指导我完成任务，在此向老师们表达敬意。

其次，我要感谢的是周围的小伙伴们。在实验室的项目期间，感谢刘子寒、乔力两位学长与常家鑫学弟的协助，正是大家集体的努力才得以完成这次的毕设项目。而在实验室之外，我要感谢我的室友和同学，感谢大家对我平时的学习上甚至生活上的帮助，以及两年期间大家的陪伴，希望同学们以后都能够前程似锦。

最后，我要感谢我的父母。感谢父母们一直以来对我的关爱，在我一次次面对困难的时候在背后默默支持我。如今我即将踏入社会，在未来的日子里定会好好报答父母的恩情！

参考文献

- [1] SEGEV E. Crowdsourcing contests[J/OL]. European Journal of Operational Research, 2020, 281(2): 241–255.
<https://doi.org/10.1016/j.ejor.2019.02.057>.
- [2] 冯剑红, 李国良, 冯建华. 众包技术研究综述 [J]. 计算机学报, 2015, 38(9): 1713–1726.
- [3] 沈鑫, 裴庆祺, 刘雪峰. 区块链技术综述 [J]. 网络与信息安全学报, 2016, 2(11): 11–20.
- [4] ZHANG T, GAO J Z, CHENG J. Crowdsourced Testing Services for Mobile Apps[C/OL] // 2017 IEEE Symposium on Service-Oriented System Engineering, SOSE 2017, San Francisco, CA, USA, April 6-9, 2017. [S.l.]: IEEE Computer Society, 2017: 75–80.
<https://doi.org/10.1109/SOSE.2017.28>.
- [5] MAO K, YANG Y, WANG Q, et al. Developer Recommendation for Crowdsourced Software Development Tasks[C/OL] // 2015 IEEE Symposium on Service-Oriented System Engineering, SOSE 2015, San Francisco Bay, CA, USA, March 30 - April 3, 2015. [S.l.]: IEEE Computer Society, 2015: 347–356.
<https://doi.org/10.1109/SOSE.2015.46>.
- [6] ZHANG Y, van der SCHAAAR M. Reputation-based incentive protocols in crowdsourcing applications[C/OL] // GREENBERG A G, SOHRABY K. 2012 IEEE INFOCOM, Orlando, FL, USA, March 25-30, 2012. [S.l.]: IEEE, 2012: 2140–2148.
<https://doi.org/10.1109/INFOCOM.2012.6195597>.
- [7] WU W, TSAI W, LI W. An evaluation framework for software crowdsourcing[J/OL]. Frontiers Comput. Sci., 2013, 7(5): 694–709.
<https://doi.org/10.1007/s11704-013-2320-2>.

- [8] BOUDREAU K J, LAKHANI K R. Using the Crowd as an Innovation Partner[J]. Harvard business review, 2013, 91(4): 60–69.
- [9] 赵江华, 穆舒婷, 王学志, et al. 科学数据众包处理研究 [J]. 计算机研究与发展, 2017, 54(2): 284–294.
- [10] HUBERMAN B A, ROMERO D M, WU F. Crowdsourcing, attention and productivity[J/OL]. J. Inf. Sci., 2009, 35(6): 758–765.
<https://doi.org/10.1177/0165551509346786>.
- [11] 夏恩君, 赵轩维, 李森. 国外众包研究现状和趋势 [J]. 技术经济, 2015, 34(001): 28–36.
- [12] YANG D, XUE G, FANG X, et al. Crowdsourcing to smartphones: incentive mechanism design for mobile phone sensing[C/OL] // The 18th Annual International Conference on Mobile Computing and Networking, Mobicom'12, Istanbul, Turkey, August 22-26, 2012. [S.l.]: ACM, 2012: 173–184.
<https://doi.org/10.1145/2348543.2348567>.
- [13] WANG Y, CAI Z, YIN G, et al. An incentive mechanism with privacy protection in mobile crowdsourcing systems[J/OL]. Comput. Networks, 2016, 102: 157–171.
<https://doi.org/10.1016/j.comnet.2016.03.016>.
- [14] ZHANG X, YANG Z, ZHOU Z, et al. Free Market of Crowdsourcing: Incentive Mechanism Design for Mobile Sensing[J/OL]. IEEE Trans. Parallel Distributed Syst., 2014, 25(12): 3190–3200.
<https://doi.org/10.1109/TPDS.2013.2297112>.
- [15] ZHU X, AN J, YANG M, et al. A Fair Incentive Mechanism for Crowdsourcing in Crowd Sensing[J/OL]. IEEE Internet Things J., 2016, 3(6): 1364–1372.
<https://doi.org/10.1109/JIOT.2016.2600634>.
- [16] LI W, ZHANG C, TANAKA Y. Privacy-Aware Sensing-Quality-Based Budget Feasible Incentive Mechanism for Crowdsourcing Fingerprint Collection[J/OL]. IEEE Access, 2020, 8: 49775–49784.
<https://doi.org/10.1109/ACCESS.2020.2974909>.

- [17] LEE J, HOH B. Sell your experiences: a market mechanism based incentive for participatory sensing[C/OL] // 8th Annual IEEE International Conference on Pervasive Computing and Communications, PerCom 2010, March 29 - April 2, 2010, Mannheim, Germany. [S.l.]: IEEE Computer Society, 2010: 60–68.
<https://doi.org/10.1109/PERCOM.2010.5466993>.
- [18] WU C, CHEN K, CHANG Y, et al. Crowdsourcing Multimedia QoE Evaluation: A Trusted Framework[J/OL]. IEEE Trans. Multim., 2013, 15(5): 1121–1137.
<https://doi.org/10.1109/TMM.2013.2241043>.
- [19] CHITTELAPPILLY A I, CHEN L, AMER-YAHIA S. A Survey of General-Purpose Crowdsourcing Techniques[J/OL]. IEEE Trans. Knowl. Data Eng., 2016, 28(9): 2246–2266.
<https://doi.org/10.1109/TKDE.2016.2555805>.
- [20] XIE H, LUI J C S, JIANG J W, et al. Incentive mechanism and protocol design for crowdsourcing systems[C/OL] // 52nd Annual Allerton Conference on Communication, Control, and Computing, Allerton 2014, Allerton Park & Retreat Center, Monticello, IL, USA, September 30 - October 3, 2014. [S.l.]: IEEE, 2014: 140–147.
<https://doi.org/10.1109/ALLERTON.2014.7028448>.
- [21] XIE H, LUI J C S, JIANG W. Mathematical Modeling of Crowdsourcing Systems: Incentive Mechanism and Rating System Design[C/OL] // IEEE 22nd International Symposium on Modelling, Analysis & Simulation of Computer and Telecommunication Systems, MASCOTS 2014, Paris, France, September 9-11, 2014. [S.l.]: IEEE Computer Society, 2014: 181–186.
<https://doi.org/10.1109/MASCOTS.2014.31>.
- [22] YU Y, LIU S, GUO L, et al. CrowdR-FBC: A Distributed Fog-Blockchains for Mobile Crowdsourcing Reputation Management[J/OL]. IEEE Internet Things J., 2020, 7(9): 8722–8735.
<https://doi.org/10.1109/JIOT.2020.2996229>.
- [23] LI M, WENG J, YANG A, et al. CrowdBC: A Blockchain-Based Decentralized Framework for Crowdsourcing[J/OL]. IEEE Trans. Parallel Distributed Syst.,

- 2019, 30(6): 1251–1266.
<https://doi.org/10.1109/TPDS.2018.2881735>.
- [24] AMETRANO F M. Bitcoin, Blockchain, and Distributed Ledger Technology[J]. Social Science Electronic Publishing, 2016.
- [25] SIDHU J. Syscoin: A Peer-to-Peer Electronic Cash System with Blockchain-Based Services for E-Business[C/OL] // 26th International Conference on Computer Communication and Networks, ICCCN 2017, Vancouver, BC, Canada, July 31 - Aug. 3, 2017. [S.l.]: IEEE, 2017: 1–6.
<https://doi.org/10.1109/ICCCN.2017.8038518>.
- [26] WOOD G. Ethereum: A secure decentralised generalised transaction ledger[J]. Ethereum project yellow paper, 2014, 151(2014): 1–32.
- [27] 邵奇峰, 金澈清, 张召, et al. 区块链技术: 架构及进展 [J]. 计算机学报, 2018.
- [28] DING W. Block chain based instrument data management system[J]. China Instrumentation, 2015, 10(1): 15–17.
- [29] ANDROULAKI E, BARGER A, BORTNIKOV V, et al. Hyperledger fabric: a distributed operating system for permissioned blockchains[C/OL] // OLIVEIRA R, FELBER P, HU Y C. 13th EuroSys Conference, EuroSys 2018, Porto, Portugal, April 23-26, 2018. [S.l.]: ACM, 2018: 30:1–30:15.
<https://doi.org/10.1145/3190508.3190538>.
- [30] 袁勇, 王飞跃. 区块链技术发展现状与展望 [J]. 自动化学报, 2016, 42(4): 481–494.
- [31] TSAI W, BLOWER R, ZHU Y, et al. A System View of Financial Blockchains[C/OL] // 2016 IEEE Symposium on Service-Oriented System Engineering, SOSE 2016, Oxford, United Kingdom, March 29 - April 2, 2016. [S.l.]: IEEE Computer Society, 2016: 450–457.
<https://doi.org/10.1109/SOSE.2016.66>.
- [32] PETERS G W, PANAYI E, CHAPELLE A. Trends in crypto-currencies and blockchain technologies: A monetary theory and regulation perspective[J]. Journal of Financial Perspectives, 2015, 3(3).

- [33] PUZIS R, BARSHAP G, ZILBERMAN P, et al. Controllable Privacy Preserving Blockchain - FiatChain: Distributed Privacy Preserving Cryptocurrency with Law Enforcement Capabilities[C/OL] // DOLEV S, HENDLER D, LODHA S, et al. Lecture Notes in Computer Science, Vol 11527: Cyber Security Cryptography and Machine Learning - Third International Symposium, CSCML 2019, Beer-Sheva, Israel, June 27-28, 2019. [S.l.]: Springer, 2019: 178–197.
https://doi.org/10.1007/978-3-030-20951-3_16.
- [34] ZHU S, CAI Z, HU H, et al. zkCrowd: A Hybrid Blockchain-Based Crowdsourcing Platform[J/OL]. IEEE Trans. Ind. Informatics, 2020, 16(6): 4196–4205.
<https://doi.org/10.1109/TII.2019.2941735>.
- [35] LIN C, HE D, ZEADALLY S, et al. SecBCS: a secure and privacy-preserving blockchain-based crowdsourcing system[J/OL]. Sci. China Inf. Sci., 2020, 63(3).
<https://doi.org/10.1007/s11432-019-9893-2>.
- [36] ZHANG J, CUI W, MA J, et al. Blockchain-based secure and fair crowdsourcing scheme[J/OL]. Int. J. Distributed Sens. Networks, 2019, 15(7).
<https://doi.org/10.1177/1550147719864890>.
- [37] KADADHA M, MIZOUNI R, SINGH S, et al. ABCrowd: An Auction Mechanism on Blockchain for Spatial Crowdsourcing[J/OL]. IEEE Access, 2020, 8: 12745–12757.
<https://doi.org/10.1109/ACCESS.2020.2965897>.
- [38] TAN L, XIAO H, SHANG X, et al. A Blockchain-based Trusted Service Mechanism for Crowdsourcing System[C/OL] // 91st IEEE Vehicular Technology Conference, VTC Spring 2020, Antwerp, Belgium, May 25-28, 2020. [S.l.]: IEEE, 2020: 1–6.
<https://doi.org/10.1109/VTC2020-Spring48590.2020.9128425>.
- [39] HOWE J. The Rise of Crowdsourcing[J]. Wired Magazine, 2006, 14(6): 1–5.
- [40] ALYAHYA S, ALRUGEHBH D. Process Improvements for Crowdsourced Software Testing[J]. International Journal of Advanced Computer Science and Applications, 2017, 8(6).

- [41] ORNELAS T, CARABAN A K, GOUVEIA R, et al. CrowdWalk: leveraging the wisdom of the crowd to inspire walking activities[C/OL] // MASE K, LANGHEINRICH M, GATICA-PEREZ D, et al. 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and 2015 ACM International Symposium on Wearable Computers, UbiComp/ISWC Adjunct 2015, Osaka, Japan, September 7-11, 2015. [S.l.]: ACM, 2015: 213–216.
<https://doi.org/10.1145/2800835.2800923>.
- [42] KHERN-AM-NUAI W, KANNAN K N, GHASEMKHANI H. Extrinsic versus Intrinsic Rewards for Contributing Reviews in an Online Platform[J/OL]. Inf. Syst. Res., 2018, 29(4): 871–892.
<https://doi.org/10.1287/isre.2017.0750>.
- [43] 王莹洁, 蔡志鹏, 童向荣, et al. 基于声誉的移动众包系统的在线激励机制 [J]. 计算机应用, 2016, 36(8): 2121–2127.
- [44] FENG Z, ZHU Y, ZHANG Q, et al. TRAC: Truthful auction for location-aware collaborative sensing in mobile crowdsourcing[C/OL] // 2014 IEEE Conference on Computer Communications, INFOCOM 2014, Toronto, Canada, April 27 - May 2, 2014. [S.l.]: IEEE, 2014: 1231–1239.
<https://doi.org/10.1109/INFOCOM.2014.6848055>.
- [45] WANG X, TUSHAR W, YUEN C, et al. Promoting Users' Participation in Mobile Crowdsourcing: A Distributed Truthful Incentive Mechanism (DTIM) Approach[J/OL]. IEEE Trans. Veh. Technol., 2020, 69(5): 5570–5582.
<https://doi.org/10.1109/TVT.2020.2982243>.
- [46] ZHENG Z, XIE S, DAI H, et al. An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends[C/OL] // KARYPIS G, ZHANG J. 2017 IEEE International Congress on Big Data, BigData Congress 2017, Honolulu, HI, USA, June 25-30, 2017. [S.l.]: IEEE Computer Society, 2017: 557–564.
<https://doi.org/10.1109/BigDataCongress.2017.85>.
- [47] ZHENG Z, XIE S, DAI H, et al. Blockchain challenges and opportunities: a survey[J/OL]. International Journal of Web and Grid Services, 2018, 14(4): 352–

375.
<https://doi.org/10.1504/IJWGS.2018.10016848>.
- [48] CHEN C L P, WEN G, LIU Y, et al. Adaptive Consensus Control for a Class of Nonlinear Multiagent Time-Delay Systems Using Neural Networks[J/OL]. IEEE Transactions on Neural Networks and Learning Systems, 2014, 25(6): 1217–1226.
<https://doi.org/10.1109/TNNLS.2014.2302477>.
- [49] NAKAMOTO S. Bitcoin: A peer-to-peer electronic cash system[R]. [S.l.]: White Paper, 2008.
- [50] NGUYEN C T, HOANG D T, OTHERS. Proof-of-Stake Consensus Mechanisms for Future Blockchain Networks: Fundamentals, Applications and Opportunities[J/OL]. IEEE Access, 2019, 7: 85727–85745.
<https://doi.org/10.1109/ACCESS.2019.2925010>.
- [51] CACHIN C, VUKOLIC M. Blockchain Consensus Protocols in the Wild (Keynote Talk)[C/OL] // 31st International Symposium on Distributed Computing, DISC 2017, October 16-20, 2017, Vienna, Austria. 2017: 1:1–1:16.
<https://doi.org/10.4230/LIPIcs.DISC.2017.1>.
- [52] STIFTER N, JUDMAYER A, WEIPPL E R. Revisiting Practical Byzantine Fault Tolerance Through Blockchain Technologies[G/OL] // BIFFL S, ECKHART M, LÜDER A, et al. Security and Quality in Cyber-Physical Systems Engineering, With Forewords by Robert M. Lee and Tom Gilb. [S.l.]: Springer, 2019: 471–495.
https://doi.org/10.1007/978-3-030-25312-7_17.
- [53] EYAL I, SIRER E G. Majority Is Not Enough: Bitcoin Mining Is Vulnerable[C/OL] // CHRISTIN N, SAFAVI-NAINI R. Lecture Notes in Computer Science, Vol 8437: Financial Cryptography and Data Security - 18th International Conference, FC 2014, Christ Church, Barbados, March 3-7, 2014, Revised Selected Papers. [S.l.]: Springer, 2014: 436–454.
https://doi.org/10.1007/978-3-662-45472-5_28.

-
- [54] 欧阳丽炜, 王帅, 袁勇, et al. 智能合约: 架构及进展 [J]. 自动化学报, 2019, 45(3): 445–457.
- [55] CACHIN C. Architecture of the hyperledger blockchain fabric[C] // Workshop on distributed cryptocurrencies and consensus ledgers : Vol 310. 2016: 4–5.
- [56] MOLYNEAUX I. The art of application performance testing: from strategy to tools[M]. 2014.

简历与科研成果

基本信息

张皓明，男，汉族，1996 年 10 月出生，江苏省无锡市人。

教育背景

2019 年 9 月 — 2021 年 6 月 南京大学软件工程系

硕士

2015 年 9 月 — 2019 年 6 月 湖南大学软件工程系

本科

《学位论文出版授权书》

本人完全同意《中国优秀博硕士学位论文全文数据库出版章程》（以下简称“章程”），愿意将本人的学位论文提交“中国学术期刊（光盘版）电子杂志社”在《中国博士学位论文全文数据库》、《中国优秀硕士学位论文全文数据库》中全文发表。《中国博士学位论文全文数据库》、《中国优秀硕士学位论文全文数据库》可以以电子、网络及其他数字媒体形式公开出版，并同意编入《中国知识资源总库》，在《中国博硕士学位论文评价数据库》中使用和在互联网上传播，同意按“章程”规定享受相关权益。

作者签名：_____

_____年____月____日

论文题名	面向可信众测激励的区块链服务系统设计与实现				
研究生学号	MF1932238	所在院系	软件学院	学位年度	2021
论文级别	<div><input type="checkbox"/> 硕士 <input type="checkbox"/> 硕士专业学位</div> <div><input type="checkbox"/> 博士 <input type="checkbox"/> 博士专业学位</div> <div>(请在方框内画勾)</div>				
作者 Email	MF1932238@smail.nju.edu.cn				
导师姓名	陈振宇 教授				

论文涉密情况：

☐ 不保密

☐ 保密，保密期(_____年____月____日至_____年____月____日)

注：请将该授权书填写后装订在学位论文最后一页（南大封面）。

