

# 衍界

测试  
专刊

6月

2025

月刊 总第159期

主办：证券期货业信息技术测试中心（大连）

## 压力测试下基于纵向分层与横向关联分析的Web系统架构分类研究 P.09

“大模型 + MCP” 在软件测试中的实践 P.25

金融大模型安全治理体系研究：  
挑战、测评与防御技术 P.50



2025年6月 总第159期【测试专刊】

主管:  大连商品交易所  
DALIAN COMMODITY EXCHANGE

主办: 证券期货业信息技术测试中心  
(大连)

地址: 大连市沙河口区会展路129号  
期货大厦

电话: 0411-8480 6897

传真: 0411-8480 6566



扫码获取本刊电子版



大连商品交易所  
官方公众号



证券期货业信息技术测试中心  
官方公众号

### 版权声明

证券期货业《测试技术与质量管理》季刊自2015年12月创刊以来,受到证券、基金、期货公司和研究机构等广大读者的欢迎。为更好的传递先进的行业测试技术、分享宝贵的质量管理经验,证券期货业信息技术测试中心(大连)决定自本期开始与大连商品交易所《衍界》合刊,并于每季度(3月、6月、9月、12月)定期刊发,特此声明。

2024年12月31日

内部资料 妥善保存



专家视角

## 压力测试下基于纵向分层与横向关联分析的Web系统架构分类研究

本文提出一种结合“纵向分层”与“横向关联”分析的Web系统架构分类方法,突破传统分类对压力测试指导不足的瓶颈。该方法能精准识别系统各层级及模块间依赖关系,理清压力传导路径,有效提升测试效率与准确性。研究已在实际场景应用,助力复杂系统在高并发、大流量环境下保持稳定运行,具有显著的实践与推广价值。

热点文章



P25

“大模型 + MCP”  
在软件测试中的实践



P50

金融大模型安全治理体系研究:  
挑战、测评与防御技术

# CONTENTS 目录

---

## ■ 资讯聚焦 Information Focus

- 01 行业动态汇编
- 05 质量技术前沿
- 07 国际热点追踪

## ■ 专家视角 Expert Perspective

- 09 压力测试下基于纵向分层与横向关联分析的Web系统架构分类研究 / 中信建投证券股份有限公司 伊鹏、刘正新、刘锺

## ■ 探索应用 Explore Applications

- 17 全链路通关自动化测试解决方案设计与实施 / 大商所飞泰测试技术有限公司 邱月明、贾露翔、赵迎春、王长春
- 25 “大模型 + MCP” 在软件测试中的实践 / 广发证券股份有限公司 余俊彦、张世同、李娴、杜瑞罡、李立峰
- 30 基于设计稿驱动的智能测试 / 国泰海通证券股份有限公司 龚柳华、谷丽丽
- 40 基于DeepSeek大模型的网上交易客户端测试用例研究与实践 / 中信建投证券股份有限公司 陶剑峰、常征、刘浩博  
上海泽众软件科技有限公司 王宏亮、王春
- 50 金融大模型安全治理体系研究：挑战、测评与防御技术 / 上交所技术有限责任公司 陈洪炎
- 58 探索参数运营1+4治理体系建设 / 郑州易盛信息技术有限公司 路畏
- 64 云原生微服务架构下的性能测试实践 / 博时基金管理有限公司 沈红富
- 70 API安全防护现状及技术解决方案 / 湘财证券股份有限公司 胡利民、曹祥波、黄辉
- 81 基于微服务+DDD的期货公司应用系统构建方案 / 国联期货股份有限公司 王莎莎、张思运、王鑫、李涛

## ■ 我们 We

- 91 数字赋能仓储服务 飞泰护航系统升级
  - 91 飞泰公司率先构建测试大模型，推动智能化测试实现新突破
  - 91 全链路通关自动化测试解决方案
-

## 行业动态 汇编

### 中国证监会印发 《推动公募基金高质量发展行动方案》

中国证监会于近日公开印发《推动公募基金高质量发展行动方案》。行动方案坚持党对公募基金行业的全面领导,突出行业发展的政治性与人民性,坚持以投资者为本的发展理念,以强监管、防风险、促高质量发展为主线,探索建立适合中国国情、市情的公募基金发展新模式;坚持问题导向、目标导向,提出一系列回应市场和社会关切的改革措施,着力督促基金公司、基金销售机构等行业机构从“重规模”向“重回报”转变,形成行业高质量发展的“拐点”。行动方案共提出25条举措。

### 大商所推出三个化工品月均价期货合约



4月24日,大连商品交易所发布公告,就线型低密度聚乙烯(LLDPE,“聚乙烯”)、聚氯乙烯(PVC)和聚丙烯(PP)月均价期货合约及相关规则向市场公开征求意见。据了解,月均价期货是一种以一个月内标的资产平均价格为结算依据的期货合约。根据公告,本次征求意见的三个化工品月均价期货合约的合约代码分别为“L合约月份F”、“V合约月份F”和“PP合约月份F”。三个化工品月均价期货合约采用现金交割

方式,即交易所以交割结算价为基础,划付持仓双方的盈亏,了结到期未平仓合约。这将是大商所首次推出现金交割期货合约。

### 摸底证券行业大模型实践与需求,深交所展开调研

2025年4月,深交所正针对证券行业在大模型基础设施领域的应用现状在行业内展开专项调研。此次调研由深交所旗下深圳证券通信有限公司联合四川证监局等单位共同发起,作为“证券行业大模型基础设施研究及应用示范项目”课题的重要组成部分。调研工作旨在通过系统梳理行业大模型基础设施的实际应用情况、现存挑战及发展诉求,同时广泛征集证券机构对于课题研究的建设性意见。

### 上交所向上市公司开放文档智能核验系统 开启信息披露“智”时代

2025年4月,上交所向上市公司开放了文档智能核验系统。该系统基于大模型机器学习技术,为上市公司提供文稿语义错误的检测服务,形成“智能检测—实时反馈—风险预警”的信息披露新范式。随着注册制改革深化,上市公司信息披露的准确性与时效性面临更高要求。然而,人工校对的局限性日益凸显:数十万字的年报文档需耗时数天反复核验,错别字、数据勾稽偏差、表述矛盾等“低级错误”不仅损害企业专业形象,更可能引发市场误读与监管风险。而文档智能核验系统的上线,则可提升工作效率。

## 上交所修订发布《股票上市规则》及配套规则指南

近日，经中国证监会批准，上交所修订发布主板、科创板《股票上市规则》及配套规则指南。本次修订旨在深入贯彻新《公司法》，衔接中国证监会《上市公司信息披露管理办法》《上市公司章程指引》等上位规则，推动上市公司优化内部治理，切实保护中小投资者合法权益。

## 广州期货交易所科技有限公司揭牌成立

4月25日，广州期货交易所科技有限公司在广州南沙举办揭牌仪式及成立动员会。广期科技是广期所的全资科技子公司，定位于“服务交易所、服务市场主体、服务粤港澳大湾区”，承担信息系统开发、测试、运维以及信息基础设施建设与维护等核心技术职能。

## 广期所修改交易规则，首次发布结算规则

5月9日，广期所修改了《广州期货交易所交易规则》，还首次发布《广州期货交易所结算规则》。其中，将“交易所上市经中国证监会批准的品种”改为“交易所上市经中国证监会注册的品种”，备受市场关注。此外，广期所还连续三年执行套期保值手续费减收，降本超3000万元，提升产业客户参与套期保值交易的积极性。

## 郑商所与国家气象信息中心联合研发新一批气温指数：服务区域不断拓展，指数体系更加完善

5月28日，郑州商品交易所与国家气象信息中心联合研发的第二批气温指数正式发布。指数覆盖珠三角、华北、中原和长江中游城市群，属于区域气温指数。新一批指数发布，意味着实体经济识别天

气风险的工具箱再度“上新”，也标志着金融气象协同联动服务高质量发展迈出新的步伐。

## 上海期货交易所铸造铝合金期货和期权注册



中国证监会近日同意上海期货交易所铸造铝合金期货和期权注册，标志着我国期货市场首个再生金属品种即将上市。证监会要求上期所做好各项准备工作，保障铸造铝合金期货和期权平稳推出和稳健运行。

## 北交所于6月14日开展北证专精特新指数第一次全网测试

6月12日，北京证券交易所办公室、全国股转公司办公室发布通知，于6月14日开展北证专精特新指数第一次全网测试。参测机构包括：北交所，全国股转公司，中国结算，深证通，中证指数，各证券公司、信息商、基金公司。测试时间为2025年6月14日（周六）8:30-12:00，模拟一个交易日。

## 中国期货业协会：5月全国期货市场成交量同比下降4.51%

以单边计算，2025年5月全国期货市场成交量为6.79亿手，成交额为54.73万亿元，同比分别下降4.51%和1.55%。1-5月全国期货市场累计成交量为33.37亿手，累计成交额为286.93万亿元，同比分别增长15.61%和21.33%。

## 证券结算风险金修订征意，下调比例、优化程序，当前规模超30亿不增缴

5月9日，证监会会同财政部起草形成《证券结算风险基金管理办法（修订草案征求意见稿）》，现向社会公开征求意见。修订涉及六方面内容，包括适应性调整计收范围，下调计收比例，完善风险基金规模相关规定，优化投资管理及存放，优化使用程序，丰富管理措施及规定。三大要点尤为值得关注：其一，结算参与人权益类和固定收益类品种现券交易计收比例下调至原有标准的十分之三，征求意见稿显示“权益类品种，按成交金额的百万分之九；固定收益类品种现券交易，按成交金额的百万分之三”；其二，明确风险基金净资产总额不少于30亿元；其三，“事前报批”修改为“事后报告”，即将风险基金使用程序由证券登记结算机构使用前向证监会、财政部报批，修改为证券登记结算机构动用后向证监会、财政部及时报告。

## 金融App频“越界” 中国互联网金融协会发布典型违规案例

2025年4月，中国互联网金融协会公布《移动金融客户端应用软件典型违规案例》，经梳理，这已是中国互联网金融协会年内第三次公布相关违规案例。受访者普遍认为，移动金融App违规收集个人信息背后原因是机构在收集大量的用户信息以进行分析和挖掘时，在隐私保护方面存在欠缺，未遵循用户知情同意、目的限制原则和最小必要原则，导致违规问题频发。

## 央行最新发声： 依法推动证券、保险领域的金融信用信息归集共享

2025年4月，国新办举行新闻发布会介绍健全社

会信用体系有关情况。下一步，人民银行将继续践行“征信为民”理念，持续推动公共征信系统扩容提质增效，进一步夯实征信服务主渠道作用：一是建强系统。不断拓展金融信用信息基础数据库的覆盖范围，依法推动证券、保险领域的金融信用信息归集共享，创新研发多元化征信产品，满足更多场景的征信服务需求。二是做优服务。加强以用户为中心的服务渠道建设，进一步完善征信中心自助查询机、APP和商业银行网银等“线上+线下”一体化征信服务网络，为社会公众提供更加高效便捷的征信服务。三是保障安全。强化数据库安全保障能力建设，建立完善数据全生命周期的安全治理机制，利用前沿技术提升数据安全防护水平，打造安全、稳定、高效的金融基础设施。

## 国家网络与信息安全信息通报中心 通报65款违法违规收集使用个人信息的移动应用

据国家网络与信息安全信息通报中心通报，依据《网络安全法》《个人信息保护法》等法律法规，按照《中央网信办、工业和信息化部、公安部、市场监管总局关于开展2025年个人信息保护系列专项行动的公告》要求，经国家计算机病毒应急处理中心检测，65款移动应用存在违法违规收集使用个人信息情况。

## 《证券期货业与银行间业务数据交换协议 第1部分： 三方存管、银期转账和结售汇业务》 等3项国家标准正式发布

《证券期货业与银行间业务数据交换协议 第1部分：三方存管、银期转账和结售汇业务》标准的发布实施对推动跨行业数据共享与协作，夯实金融基础设施具有重要意义。《证券交易数据交换协议》标准的发布实施，有效降低系统对接成本，保障交易数据传输的安全性与可靠性，为我国资本市场业

务创新、监管升级和对外开放提供了底层支撑。《期货市场客户开户数据接口》标准的发布实施，有利于破除期货市场信息孤岛，提升客户开户效率与数据统计质量，切实维护投资者合法权益，为保障期货市场数字化转型和稳健运行提供基础性支撑。

#### 第四届证标委召开工作组2025年上半年工作会议

2025年4月18日，第四届证标委工作组2025年上半年工作会议在成都召开。会议全面解读了《加强证券期货业标准化工作三年行动计划（2024-2026年）》政策文件，听取了秘书处2024年行业标准化工作整体推进情况、2025年行业标准化工作要点的汇报，各工作组交流讨论了标准规划落实、标准课题推进、标准研制及应用推广、2025年工作计划等。

#### 郑商所推出新套利指令 聚酯产业链风险管理体系再升级

5月26日，郑商所发布通知，为进一步提升期货市场服务实体经济效能，将于2025年5月29日推出PTA-瓶片、PTA-对二甲苯跨品种套利指令。其中，组合保证金跨品种抵扣自当日结算时起生效，跨品种套利指令于当日夜盘交易时正式上线，相关保证金收取和限仓制度将严格遵循《郑州商品交易所套利交易管理办法》执行。此次新指令覆盖聚酯产业链上游原料对二甲苯、中游核心产品PTA及下游瓶片，构建起从原料端到终端产品的全链条联动机制，将显著提升市场参与者风险管理能力。

#### 期货业两大新规有望年内落地 行业生态或重塑

2025年5月，中国证监会制定了2025年度立法工作计划，对全年的立法工作做了总体部署。其中，“力争年内出台的重点项目”包括制定《衍生品交易监督

管理办法》、修订《期货公司监督管理办法》。该办法的制定，对于规范衍生品交易市场、提升期货公司监管效能和推动期货业高质量发展具有重要意义，有助于构建一个更加健康、稳定、透明的期货市场环境，为投资者提供更加安全、便捷的投资渠道。

#### 香港证监会举办研讨会 加强与证券业界沟通

香港证券及期货事务监察委员会9号举办题为“乘风破浪 稳中求进”的证券业发展研讨会，吸引约700名证券业界高层代表参与，就监管观察所得及最新的合规事宜进行深入交流，进一步加强证监会与业界的联系。据悉，此次研讨会的参与人数为去年10月首届活动的三倍以上。会上讨论的议题涵盖首次公开招股市场的最新发展，以及促进交易所买卖产品和代币化产品创新的措施。

#### 首批以科创债为抵押品的外币回购业务 在外汇交易中心和上海清算所落地



近日，工商银行、浦发银行和中信银行通过中国外汇交易中心外汇交易平台达成首批以上海清算所托管的科创债券为抵押品的外币回购交易，并通过上海清算所完成清算结算。本次科创债外币回购业务的落地，标志着银行间市场科技创新债券的使用场景和外币回购抵押品类型进一步丰富，金融服务实体经济效能持续提升，支持科技创新的金融生态体系持续完善，有利于推动金融、科技、产业融合发展，为加快建设科技强国和实现高水平科技自立自强提供有力金融支撑。

## 监管要求机构切实提升对个人信息保护工作

证监会在业内向机构通报了行业个人信息保护有关问题。通报称，部分机构移动应用程序（APP）、小程序、软件开发工具包（SDK）在个人信息收集、存储、传输、利用过程中，未能严格落实《个人信息保护法》，存在合规问题和风险隐患。监管要求，机构切实提升对个人信息保护工作的重视程度，及时排查APP、小程序和SDK，并完成问题整改。机构可以通过APP检测认证、外部评估等方式，提升个人信息保护专业能力，加强问题分析研判，及时防范化解潜在风险隐患。

## 持续完善指数体系，上证380指数优化、上证580指数即将“上新”

为进一步优化上证指数体系，就上海证券市场中、小市值上市公司整体表现提供更全面的视角和参考，上海证券交易所与中证指数公司于2025年5月30日公布优化上证380指数编制方案，并发布上证580指数。2025年6月16日，上证380指数优化将正式实施，上证580指数也将发布实时行情。



## 质量技术 前沿

### 国家网络安全通报中心通报： 大模型工具Ollama存在安全风险

国家网络安全通报中心发布关于大模型工具Ollama存在安全风险的情况通报，据清华大学网络空间测绘联合研究中心分析，开源跨平台大模型工具Ollama默认配置存在未授权访问与模型窃取等安全隐患。鉴于目前DeepSeek等大模型的研究部署和应用非常广泛，多数用户使用Ollama私有化部署且未修改默认配置，存在数据泄露、算力盗取、服务中断等安全风险，极易引发网络和数据安全事件。

### 6项网络安全国家标准获批发布

由全国网络安全标准化技术委员会归口的6项国家标准正式发布。本次发布的标准聚焦于数据安全、生成式人工智能安全等关键领域，进一步丰富了大网络安全工作格局下的网络安全标准体系建设，为国家数据安全和人工智能安全的管理工作及产业发展提供标准支撑。标准具体内容如下：《数据安全技术敏感个人信息处理安全要求》、《网络安全技术 网络安全保险应用指南》、《数据安全技术 数据安全风险评估方法》、《网络安全技术 生成式人工智能预训练

和优化训练数据安全规范》、《网络安全技术 生成式人工智能服务安全基本要求》、《网络安全技术 生成式人工智能数据标注安全规范》，以上标准将于2025年11月1日起正式实施。

#### 第四届证标委信息安全专业工作组 (WG41) 召开2025年第一次工作会议

近日，第四届证标委信息安全专业工作组 (WG41) 在福州召开2025年第一次工作会议。证监会科技监管司、证标委秘书处、组内专家，以及相关单位代表共30余人参加会议。WG41工作组首席专家、国泰海通证券首席信息官俞枫主持会议。会议强调，工作组要加强行业标准宣传推广，加大拟立项标准潜在应用效果评估力度，定期评估已发布标准应用实施情况。同时，要强化专家履职管理，对履职不到位的专家及时督促整改打造一支专业素养高、责任意识强的专家队伍。

#### GB/T 45399 - 2025 《信息技术 云计算 超融合系统通用技术要求》正式发布

国家标准《信息技术 云计算 超融合系统通用技术要求》由TC28 (全国信息技术标准化技术委员会) 归口，TC28SC38 (全国信息技术标准化技术委员会云计算和分布式平台分会) 执行，主管部门为国家标准委，计划于2025-10-01正式实施。

#### CNAS 专项监督案例警示与政策宣贯会 于6月19日召开

中国合格评定国家认可委员会(CNAS) 将于2025年6月19-20日召开专项监督案例警示与政策宣贯会。会议对CNAS专项监督工作、认可规则、相关认可政策及专项监督典型案例进行宣贯和讲解，以进一步提升认可实验室的风险意识和质量意识。

#### 第十四届双态IT用户大会圆满结束! 共论AI管理及数据管理提升之道

2025年5月23日，2025第十四届双态IT用户大会之“人工智能管理与数据管理提升之道主题会议”圆满结束。本次会议由中国电子工业标准化技术协会信息技术服务分会、信标委人工智能分技术委员会指导，ITSS数据中心运营管理组(DCMG)、DCMM金融行业社区技术委员会主办，双态IT论坛承办，共有来自金融、央国企等行业的400余人到场出席。

#### 京东外卖因订单量激增，导致服务器负载过高，商家接单系统、骑手调配算法和用户支付链路全线崩溃

4月22日，京东外卖因订单量激增(订单量达到了平时的3倍)导致服务器负载过高，商家接单系统、骑手调配算法和用户支付链路全线崩溃，京东外卖在系统崩溃后迅速进行了技术排查，并在20分钟内恢复了服务。该事件，一是反映出系统在高峰期的处理能力和弹性扩展方面存在不足；二是应定期进行系统压力测试，评估系统的承载能力，及时发现并解决潜在问题。三是应建立实时监控和预警机制，及时发现系统负载过高的情况，并采取相应措施减轻负载。

#### 纳斯达克与AWS合作简化交易所云过渡进程

纳斯达克正在与亚马逊网络服务(AWS)合作，帮助交易所更高效地过渡到云服务，而不会产生大量的开发成本。该合作伙伴旨在提供基础设施、软件和数据管理服务。使交易所能够在不影响日常经营的情况下经济高效地克服数字化障碍。最初的项目包括与约翰内斯堡证券交易所和墨西哥Grupo BMV的合作，并计划在全球范围内扩张。据了解，包括纳斯达克和芝加哥商业交易所集团在内的主要交易所已分别与最大的云服务提供商AWS和谷歌合作。



## 国际热点 追踪

### 上交所与巴西证券期货交易所ETF互通 首只产品成功上市

5月26日,上交所与巴西证券期货交易所ETF互通项下首只产品在巴西正式上市交易,由巴西Bradesco资产管理公司发行,跟踪汇添富在上交所上市的沪深300ETF产品,标志着ETF互联互通机制首次在南美市场落地,南美市场投资者能够通过更加便捷、高效的跨境投资渠道,分享中国经济发展成果。

### 第79届世界质量与改进大会在美国举行

近日,美国质量学会主办的第79届世界质量与改进大会(WCQI)在美国科罗拉多州丹佛市举行。本届大会以“将质量提升到新的高度”为主题,举行了两场主旨演讲、近100场平行会议及案例研究。会议议题囊括质量管理的全链条,包括推动人才迈向巅峰征程中的工具与技术、实现流程卓越表现、赢得管理和领导实践、团队建设与动态绩效、项目和变更管理、流程设计、关键绩效指标(KPI)、衡量流程改进的影响、数据分析仪表盘、消除浪费、减少可变性等。

### 道明证券加速衍生品交易数字化进程

5月8日,道明证券(TD Securities)宣布已完成国际互换与衍生工具协会(ISDA)ISDA Create 电子协议平台的全球系统整合,进一步推动其衍生品交易文件的数字化转型,该平台由技术企业CreateIQ与标普全球市场

财智提供技术支持,通过全流程数字化管理(从条款谈判到最终签署)和AI辅助谈判功能(如 CiO Extract),显著提升跨境交易效率;同时满足全球合规要求。目前,ISDA Create 已被全球 500 余家机构采用,包括30家财富500强企业和6家顶级银行,标志着衍生品行业正式迈入智能协作新时代。此次合作也为未来区块链等技术的应用奠定了基础。

### XBIT去中心化交易所技术创新

2025年4月,欧洲央行宣布提前降息50个基点,XBIT通过跨链流动性聚合引擎,整合了以太坊、Solana等12条主流公链的深度流动性池,实现了0.05%的行业最低价差,较同类DEX降低了70%,大大降低了交易成本。其独创的AI做市算法能够实时监测市场动态,动态调整资金费率,将套利空间压缩至近乎为零,提升了市场的稳定性。

### 管道魔法木马利用Windows零日漏洞部署勒索软件

微软披露,一个现已修复的影响Windows通用日志文件系统(CLFS)的安全漏洞曾被作为零日漏洞用于针对少数目标的勒索软件攻击中。微软表示:"受害者包括美国信息技术(IT)和房地产行业组织、委内瑞拉金融行业企业、一家西班牙软件公司以及沙特阿拉伯零售行业机构。"相关漏洞编号为CVE-2025-29824,是CLFS中的一个权限提升漏洞,攻击者可利用该漏洞获取SYSTEM权限。微软已在2025年4月的补丁星期二更

新中修复该漏洞。微软将CVE-2025-29824漏洞的利用活动追踪命名为Storm-2460,攻击者还使用名为PipeMagic的恶意软件来投递漏洞利用程序及勒索软件载荷。

### 2025年5月,美国最大加密货币交易所Coinbase发生数据泄露事件



网络犯罪分子贿赂海外客服人员窃取客户数据,实施社会工程攻击。Coinbase预计损失1.8亿至4亿美元,用于客户赔偿、系统修复和安全加固等。Coinbase承诺赔偿受影响用户损失,并与执法部门合作追查攻击者。

### Gate.io合约服务中断与升级

2025年4月23日,加密货币交易所Gate.io因流量过大导致合约服务短暂暂停,部分用户合约资产无预警归零。平台随后完成紧急升级并补偿用户损失,但引发市场对交易所技术稳定性的担忧

### 华沙交易所暂停交易事件

2025年4月8日,波兰华沙证券交易所因“安全原因”暂停所有市场交易一小时。尽管官方未披露具体细节,但市场猜测可能与网络安全威胁有关,反映出金融机构对潜在攻击的警惕性提升。

### CME在香港建立铝仓库

芝加哥商业交易所(CME)正与香港特别行政区政府商讨在香港设立铝仓库,将其纳入铝期货合约的实物交割网络,旨在通过香港仓库提升亚太地区铝合约交易便利性,强化全球大宗商品市场布局,同时与伦敦金属交易所(LME)形成竞争互补格局。

(以上资讯来自互联网)

# 压力测试下基于纵向分层与横向关联分析的Web系统架构分类研究

伊鹏、刘正新、刘镭 / 中信建投证券股份有限公司

随着Web技术的不断发展, Web类系统架构日益复杂, 压力测试已成为系统开发和优化中的重要环节。传统的Web系统架构分类方法难以适配压力测试的对系统的分析要求。本文提出了一种基于纵向分层和横向关联性分析的Web系统架构分类方法, 为压力测试提供更精确的分析框架, 该分类方法已在实际测试工作中投入使用, 能够显著提升压力测试的效率和准确性。从系统的纵向分层和横向系统间的关联关系两个维度进行分类, 本文能够更细致地划分Web系统的架构类型, 为压力测试人员进行有效且精确的测试提供指导。通过该分类方法, 能够明确系统的依赖链条和交互路径, 进一步拆解系统的压力测试链路, 分析压力在不同层次间的传导路径, 从而提升测试效率。本文的研究为实际应用中的压力测试提供了有力支持, 为高并发、大流量环境下Web系统的稳定性和可靠性保障提供理论依据和实践指导。

## 一、引言

随着互联网的迅速发展, Web系统已成为现代信息系统的核心类别之一, 尤其在金融领域, 扮演着不可或缺的角色。据《全球数字报告2023》数据显示, 全球互联网用户已超过50亿, 其中约有70%的用户通过Web平台进行日常在线交易和服务消费。特别是在金融领域, 研究表明, 全球超过80%的金融交易现在都通过Web系统完成, 包括在线支付、投资交易、以及银行业务等。

Web系统不仅提供各种在线服务, 还连接了用户、设备和数据, 形成庞大的信息生态系统。然而, 随着访问量和业务需求的增加, Web系统的性能和可靠性成为开发和运维的主要挑战。而压力测试的目标是确保系统在高并发、大流量场景下的稳定性和响应速度, 被广泛用于测试Web系统在高负载、突发流量及系统故障时, 其是否能够保持稳定高效, 已成为确保Web系统可靠运行的关键步骤。

### (一) 压力测试在Web系统开发维护中的关键作用

压力测试不同于常规功能测试, 主要关注系统在高负载、长时间运行或突发流量等极限场景下的表现。通过模拟大量用户并发访问、资源消耗和负载均衡, 压力测试能揭示系统潜在瓶颈、性能下降趋势及高负载下的问题。

Web系统的压力测试不仅验证性能是否达标, 还评估系统的可靠性和可扩展性。在设计压力测试时, 测试人员需考虑Web系统各组成部分及其交互方式, 如前端UI、后端服务、数据库、缓存系统等, 确保各环节在高负载下的稳定性。例如, 前端响应时间、后端并发处理能力、数据库查询效率和网络带宽等, 都需在压力测试中全面评估。

### (二) 研究目的和意义

尽管压力测试在Web系统开发和运维中已广泛应用, 但如何高效、系统地进行测试, 特别是在面对陌生Web系统时, 快速剖析并准确分析压力传导路径, 仍然

耗时。因此,探索新的架构分类方法以提升分析效率至关重要。

本文从纵向分层和上下游依赖两个角度对Web系统架构进行深入分析,并提出了一种新的分类方法:基于纵向分层与横向关联分析的Web系统架构分类。

## 二、Web类系统架构概述

### (一) 架构的定义

软件架构是指软件系统的顶层结构,涵盖系统的静态组织、动态行为和运行时特性。它定义了各子系统、模块和组件的协作关系,展示它们如何共同实现功能和性能目标。Web应用架构本质是合理拆分系统,拆分的各个组件间关系涉及前后端交互、数据处理和业务逻辑等多个方面。因此,架构设计直接影响系统的可扩展性、性能、安全性和维护性,需考虑高负载和高并发下的稳定性和响应能力。

### (二) 常见的Web类系统架构类型分类

Web系统的架构随着云计算、微服务、容器化等新兴技术的普及也变得更加复杂。当前,业界通用的Web系统架构分类如下。

#### 1. 单体架构

单体架构将所有业务功能集成在一个代码库中,适合小型应用和团队,开发和部署简单。但随着系统规模扩大,单体架构的扩展性差、维护困难等问题逐渐显现,难以应对高并发和大流量的场景。

#### 2. 分布式架构

分布式架构将模块部署到多个节点,提供更好的扩展性和容错能力。优点包括高可用性、负载均衡和故障隔离,但设计时需考虑高效通信、数据存储和一致性。然而,分布式架构也带来了数据一致性、服务间通信复杂度增加、以及跨节点事务管理等挑战。例如,在保证最终一致性和处理网络延迟的同时,如何有效地同步分布式数据仍是一个难题。此外,服务间

的通信通常需要额外的协议支持,如消息队列或API网关,以确保系统的可靠性和可扩展性。

#### 3. 微服务架构

微服务架构将应用拆分为独立的服务,具有更好的独立性、可扩展性和灵活性。每个服务可以独立开发、部署、扩展和更新,从而提高系统的可维护性和开发效率。但挑战包括服务间通信、分布式事务和数据一致性等问题。

虽然单体、分布式和微服务架构提供了不同规模应用的解决方案,但它们在压力测试中存在局限,这种分类方式难以清晰识别不同模块和层次之间的压力传递路径。为了更有效进行压力测试,本文提出从纵向分层和横向关联的两个维度对Web系统进行更细化的分类,有助于拆分系统的同时可以有效识别和分析压力传导路径。

## 三、基于纵向分层和横向关联的Web系统架构分类方法

在Web系统的压力测试中,理解系统架构及其依赖关系至关重要。传统架构分类多侧重于功能或服务拆分,忽视了架构层次和依赖关系的综合分析。为此,本文提出一种新的基于纵向分层和横向关联的架构分类方法。

1) 纵向分层:按请求处理顺序划分应用层次,如前端、应用服务、数据服务等,面向业务深度。纵向分层能够清晰划分系统功能,便于识别每一层的性能瓶颈和负载情况,例如:应用层可能受限于一些逻辑处理复杂度,而数据服务层则可能受数据库查询效率和存储性能的影响。通过这种分层,可以更有针对性地优化每一层的性能,提升系统整体响应能力。

2) 横向关联:主要是分析与系统相关联的其他服务和系统与被测系统之间的关联特性。特别是在微服务和分布式架构中,服务间的依赖直接决定压力的

传递方式。例如，某一服务的压力可通过API传递给下游服务，但下游服务可能受数据库性能和网络延迟的影响，进而影响到被测系统性能。分析横向关联有助于明确压力传导路径，识别跨模块和跨服务的瓶颈及故障点。

(一) Web系统架构的纵向分层

1. 分层架构的概念

纵向分层架构 (Vertical Layering Architecture) 通过按请求传递路径将Web系统划分为多个功能模块，以简化系统复杂度的管理。每个层次负责特定功能，职责独立。其目的是通过清晰的功能划分降低各层耦合度，便于压测分析。例如，在某券商电商平台的架构设计中，采用了纵向分层架构，成功地将前端展示层、业务处理层和数据存储层进行了明确分离。该架构显著提升了系统的可维护性和性能，尤其在进行了高并发压力测试时，能够快速定位到各层的性能瓶颈，如前端页面渲染延迟或数据库查询响应慢等问题，从而实现了有针对性的优化。

2. Web系统的不同层次

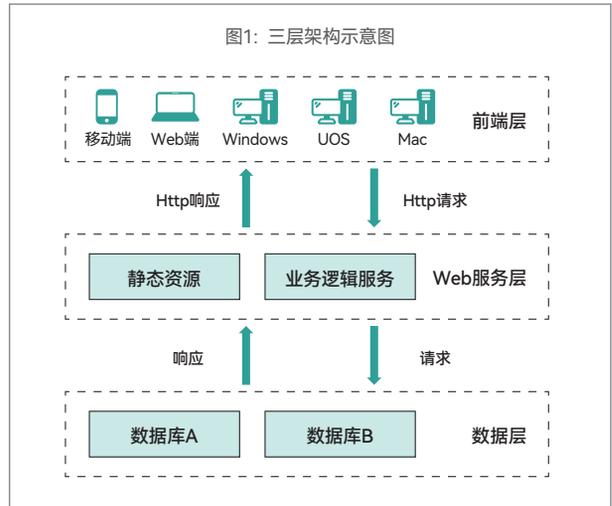
表1是常见的层次说明。

3. 典型分层案例

1) 三层架构

主要包括，前端层、Web服务层、数据层，其由于

系统架构较为简单，因此Web服务层兼应用服务层功能。三层架构如图1举例：



2) 四层架构

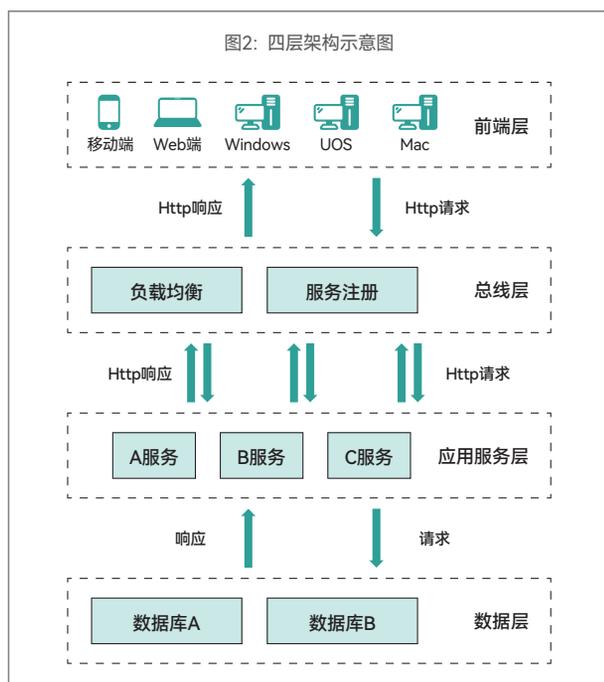
较为复杂的系统，一般在客户端与web服务层之间，增加一层总线层。因此四层架构为：前端层、总线层、应用服务层以及数据层。如图2举例，在三层架构中添加总线层，扩展为四层架构：

3) 四层以上架构

当系统业务复杂度增加，对并发和安全性要求提升时，需要扩展至四层以上架构。在三层架构基础上增加接入层，如下图案例，接入层主要为了增强安全策略。各层之间由于负载均衡和缓存配置也可能引入接入层。

表1: web系统不同层次说明

层次	主要功能	关键技术
前端层	用户交互界面，接收请求、展示数据、提供反馈	HTML、CSS、JavaScript，处理UI和交互
Web服务层	处理HTTP请求，提供静态资源、缓存、负载均衡	反向代理、Nginx、Apache，路由请求转发
总线层	服务间通信、负载均衡、日志监控、服务治理	消息队列、服务注册发现 (Consul、Eureka)
应用层	处理核心业务逻辑，执行业务计算并返回结果	微服务架构、Spring Boot、Node.js等
数据层	存储、查询、管理数据，提供增删改查功能	数据库 (MySQL、MongoDB)、缓存 (Redis)、文件管理等 (FCS)
接入层	优化请求分发，减轻后端压力，位于用户入口 (CDN、负载均衡) 或系统内部 (服务间)	CDN、负载均衡、边缘计算、安全管理



## (二) Web系统架构的上下游依赖

在复杂的Web系统中,上下游依赖关系对系统性能、稳定性和可扩展性至关重要。分析上下游依赖有助于明确系统边界并识别压力传导路径,为压力测试设计提供依据。本章将从上下游依赖的定义及分类两方面进行阐述。

### 1. 上下游依赖的定义

上下游依赖指通过接口或其他通信方式相互关联的系统模块。上游系统提供服务支持,通常为依赖源;下游系统调用上游服务,通常为依赖目标。在实际应用中,上下游依赖关系广泛存在。例如,在某金融系统中,支付模块依赖于用户认证模块,形成了上下游依赖关系。用户认证模块作为上游系统验证用户身份,并为支付模块提供必要的认证信息,确保支付操作的安全性。如果用户认证模块出现故障或延迟,支付模块的操作将受到影响,从而导致整个支付流程的中断。通过分析上下游依赖关系,可以更好地理解系统的工作流程和潜在故障点,确保系统稳定运行。



### 2. 两大类别

#### 1) 无上下游依赖系统

也被称作单体系统,这种系统为独立应用或模块,所有业务逻辑、数据存储和功能处理都在内部完成,不依赖外部服务或系统,架构简单、易开发部署。典型例子如博客系统、个人网站等,该类系统的所有功能都在单一应用中实现。

#### 2) 有上下游依赖系统

这类系统依赖外部服务进行数据或逻辑交互。

例如,某系统调用文件搜索服务、文件管理服务或流程引擎服务,并与其他系统共享数据,需要接口进行同步。

### (三) 基于纵向分层和横向依赖的分类框架

在纵向分层和横向依赖的分类方法下,压力测试不仅仅局限于评估单个层次的负载能力,而是能够形成一个完整的测试链路。在测试过程中,测试人员需要从各个层面入手,分析跨层次、跨模块的压力传导路径,从而通过测试分析确定相关层次的性能短板。通过这种细化的分类方法,Web系统的压力测试策略将变得更加全面、准确和高效。

#### 1. 分类框架概述

综合3.1和3.2小节内容,我们提出了一种新的Web系统架构分类框架,从纵向分层和横向依赖两个维度对系统进行拆分和分类:

1) 纵向分层: 将系统划分为多个层次,每个层次具备独立的功能和职责。

- 前端层(客户端或UI层)
- web服务层(前端静态资源处理层)
- 应用层(业务逻辑处理层)
- 数据层(数据库或数据存储层)
- 总线层(服务协调层,负责服务间的通信)
- 接入层(负责缓存、负载均衡、安全控制和流量控制等)

2) 横向依赖: 分析系统各层之间及其模块之间的依赖关系,分为有依赖和无依赖两类。

· 有上下游依赖: 模块间存在依赖关系,服务调用和数据传递影响系统性能。

· 无上下游依赖: 模块间独立运行,耦合度较低。

结合上述两点,常规的分类包括:

表2: web系统纵向分层与横向关联分类方法

	横向有关联系统/模块	横向无关联系统/模块
纵向三层	纵向三层横向无关联系统	纵向三层横向有关联系统
纵向四层	纵向四层横向无关联系统	纵向四层横向有关联系统
纵向四层以上	纵向四层以上横向无关联系统	纵向四层以上横向有关联系统

#### 2. 分类方法具体实施步骤

1) 层次划分: 根据系统功能的分工,按照3.1小节层次定义,将系统划分为多个层次,并明确每一层职责。

2) 依赖关系分析: 对于每一层及其模块,分析其与外部系统的依赖关系。识别出本系统与外部系统之间的服务调用、数据交换等依赖关系。相关分析方法可参考下表:

表3: 依赖关系具体分析方法

分析方法	具体形式
静态分析: 基于系统架构图和代码依赖关系进行分析	· 工具支持: 使用SonarQube等工具扫描代码,分析模块间的依赖关系。 · 文档查看: 生成依赖树或依赖图,展示系统间的静态依赖结构。
动态分析: 通过系统运行时的数据和服务调用监控依赖关系	· 日志分析: 通过调用链日志(如OpenTelemetry)记录服务间的调用关系和时间消耗。 · 链路跟踪: 使用分布式追踪工具(如Zipkin、Jaeger)监控系统间的动态依赖和传递路径。
模拟测试: 通过模拟上下游服务的行为,分析依赖关系的表现	· 服务降级模拟: 人为制造上游服务故障,观察对下游服务的影响。 · 流量压力测试: 对依赖系统的各模块施加不同负载,观察性能变化。

3) 压力测试链路识别: 基于系统的依赖关系和层次结构,识别压力测试链路。通过模拟不同层之间的负载传递和响应,识别可能的性能瓶颈。例如,前端层可能受到请求量激增的影响,应用层则可能在复杂的业务逻辑处理过程中成为瓶颈。

## 四、基于纵向分层与横向关联分类方法下的web系统压测案例分析

本章节基于特定案例系统，示范如何运用本文提出的分类方法对系统进行剖析，从而明确压测目标和发压点的位置。并且通过该分类方法分析系统架构，可以迅速梳理压力传导路径和压测数据监测点，从而提升测试人员的工作效率。

### （一）剖析被测系统确定具体类别

本案例系统是一个面向企业用户的文档管理平台，日均处理文档量超过50万。该系统主要功能是进行文档存储与预处理，包括文档上传、格式转换、内容提取、版本管理等。平台支持多种文档格式，如PDF、Word、Excel等，并能够对文档内容进行OCR识别与索引，便于后续搜索与检索。

对该系统的分析过程需要结合设计开发文档并与系统负责人沟通重点功能模块，了解系统架构组成。

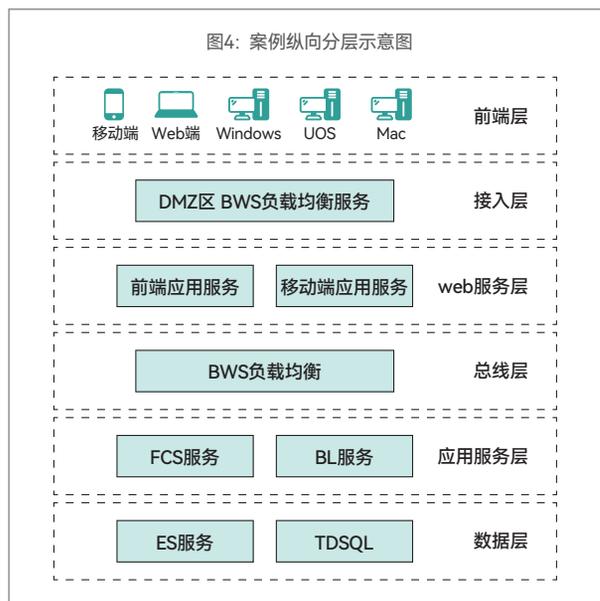
#### 1. 纵向分层分析

首先是纵向分层分析，需要根据已有信息梳理各个层次及其职责功能，当前案例系统分层层次如下，属于四层以上架构：

表4：当前案例系统纵向分层说明

架构层次	层次说明
前端层	该系统支持多种用户访问方式：移动端、web端、win端、mac端等。
接入层	该系统在接入层部署DMZ区，确保互联网非交易终端的网络安全，并提供非交易类业务系统的反向代理和接入服务。
Web服务层	该层部署系统的Web服务，负责提供静态资源和数据缓存服务。
总线层	该层负责将需要后台业务处理的请求从Web服务层转发至应用服务层，提供负载均衡和后端服务间的逻辑协调与通信。
应用服务层	提供具体的业务逻辑处理服务。
数据层	提供结构化和非结构化数据的存取服务。

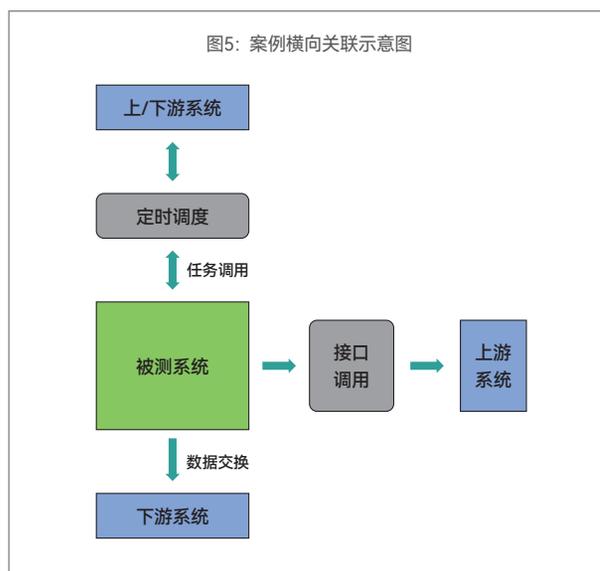
最终依据上述分层调研，绘制示分层意图如下：



通过纵向分层，确定该案例系统本身的架构设置，此处不涉及到上下游系统/模块的依赖。

#### 2. 横向依赖分析

本案例中对横向关联分析主要是使用静态分析方法，基于系统设计开发文档和代码依赖关系进行分析梳理，同时也需与主要系统建设负责人进行沟通，最终得到的横向关联图如下所示：



图中绿色部分为当前案例系统的抽象, 其与其他系统/模块的交互通过以下三种方式, 且交互的层次位于该案例系统的应用服务层:

1) 通过接口调用上游系统提供的文档信息辅助处理功能, 接口采用RESTful API进行通信, 数据格式为JSON。通过OAuth 2.0协议进行认证和授权, 确保数据安全;

2) 通过数据交换任务与其他系统的数据库直接信息特定数据的交换与处理, 该过程使用ETL(Extract, Transform, Load)工具进行数据提取、转换和加载, 系统通过数据库连接池实现与其他系统数据库的高效通信, 使用SQL语句或存储过程获取并更新数据;

3) 通过定时调度, 以固定频率拉去文档负责人等相关信息, 该过程使用定时任务调度框架(Quartz)来设定任务的执行频率和时机。定时任务在后台定期拉取其他系统的数据。

因此, 综合基于纵向分层和横向关联分析, 该案例系统架构属于纵向四层以上横向有关联类别。

## (二) 确定压测目的与发压点位置

纵向分层和横向关联分类方法助力确定压测目的与发压点位置。

### 1) 压测目的: 并发量测试

本案例系统压测旨在评估系统在特定并发用户数下的性能表现, 并验证接口的最大并发承载能力。在四层以上架构中, 压力从前端到数据层的传递路径至关重要, 因此, 本文提出的分类方法明确了各层职责, 有助于精准确定测试目标, 识别系统各层的负载情况与性能瓶颈。

### 2) 发压点位置: 前端层

基于本文的分类方法, 系统为四层以上架构, 因此发压点设定在前端层, 进行端到端的压力测试, 而非中间层介入。前端施压能够真实模拟用户行为,

全面评估请求在系统各层的传递延迟与瓶颈, 确保测试结果更具参考价值。

## (三) 确定压测链路

结合4.1小节被测系统剖析以及发压点位置, 得到本案例系统一个请求从压力工具发出到回到压力工具端的完整路径依次如下表所示:

表5: 案例压力路径

路径节点	压力链路传递
Jmeter工具发起请求	模拟用户在浏览器或客户端输入网址或点击链接。
接入层负载均衡转发请求	接入层接收HTTP请求, 根据负载均衡策略转发至Web服务层。
Web服务层接收请求	Web服务层接收请求, 提供静态资源, 若为动态请求则转发至应用服务层。
总线层分发中转请求	总线层作为应用服务间的请求和响应中转站。
应用服务层处理请求	应用服务器接收请求, 执行业务逻辑和数据验证, 并通过接口调用、定时调度、数据交换与上下游系统交互, 辅助相关业务逻辑的处理。
应用服务层调用依赖系统辅助处理业务逻辑	调用方式包括接口调用、数据交换、任务调度三种方式。
依赖系统返回相关信息给应用服务层	上游依赖系统返回信息。
应用服务层与数据库交互	应用服务器向数据库发送查询或更新请求, 数据库返回结果。
应用服务层生成响应	根据数据库结果, 应用服务器生成HTTP响应。
Web服务层转发响应	Web服务层将响应发送至BWS负载均衡器。
接入层转发响应	接入层将响应发送回客户端。
前端层接收响应	客户端接收响应并渲染页面或执行后续操作。

基于本文的分类方法, 我们不仅可以快速分析系统内部各个层次间的请求传递路径, 又可以分析系统各个层次与其他系统/模块之间的调用依赖关系, 熟知每个请求具体的流转路径。

## (四) 确定压测数据监测

在分析完压测链路后, 即可进入监测点确定环节, 监测点根据具体的测试需求从链路中各个请求传递节点进行筛选, 如下是本次并发量测试的监测点。

表6: 压力测试指标监测点

路径节点	部署的服务	可监测指标
应用服务层- 服务器1: xx.xx.xx.xxx	应用服务器 (测试环境) 后端服务1	CPU利用率 内存利用率 磁盘I/O 网络带宽
应用服务层- 服务器2: xx.xx.xx.xxx	应用服务器 (测试环境) 后端服务2	CPU利用率 内存利用率 磁盘I/O 网络带宽
数据库 服务器1	数据库服务器 (测试环境)	CPU利用率 内存利用率 磁盘I/O 网络带宽
前端层- 压测机	Jmeter	并发用户数 (最佳并发和最大并发) TPS (系统每秒处理事务数) 错误率 响应时间

重点监控应用服务层和数据层服务器的硬件指标信息,以及发压端的业务指标信息,需要注意测试工作成功可能根据具体结果调整监控点位置。

通过上述四个主要步骤,我们实践了本文提出的Web系统分类方法,实现了对系统架构的有效剖析,快速确定了合适的发压点和压测目标。在梳理系统的纵向分层与横向关联信息后,我们能够高效确定压测链路,明确请求传递路径,并为数据监测点的确定奠定基础。

实践表明,以该案例为参考,运用本文的Web系统分类方法,可以更加精细地划分系统架构类型,为压力测试提供明确的指导。该方法能够清晰展现系统的依赖链路和交互路径,深入拆解压测链路,分析压力在不同层次间的传导情况,从而提升测试效率。本文的研究不仅为实际应用中的压力测试提供支持,也为Web系统在高并发、大流量环境下的稳定性和可靠性保障提供了理论依据和实践指导。

## 五、总结与展望

随着Web技术的发展,系统复杂性不断增加,因此合理的架构设计和高效的压力测试尤为重要。本文提出了一种基于纵向分层和横向关联分析的架构分类方法,帮助理解和优化Web系统性能。纵向分层清晰划分功能,使每层独立处理其职责;横向依赖性揭示系统间/模块间的交互与协作。通过分析这些维度,压力测试人员可以快速剖析系统,识别压力传导路径,从而有效指导压测。

现代Web系统架构涉及更多维度,如跨平台、无服务器(Serverless)和事件驱动架构等。未来将结合公司系统类型和行业发展,进一步探索多维架构分类方法,结合架构特征和实践经验,提出更适应未来Web系统的架构模型,助力压力测试提质增效。

责任编辑:邓晓娜

## 参考文献

- [1] 张亮亮. 面向高并发的web系统架构设计与实现[D]. 中国科学院大学, 2018.
- [2] 黄芝龙, 徐莉莎, 瞿少成. 高并发Web电商系统的设计与优化[J]. 计算机与数字工程, 2019, 47(07): 1719-1724+1775.
- [3] Gao, X., Zhao, Y., & Liu, X. (2014). Scalability and Performance Evaluation of Web-based Applications. Computers in Industry, 65(2), 163-172.
- [4] 李真, 朱海滨. 基于Web服务总线的应用集成的研究与实现[J]. 科学技术与工程, 2009, 9(10): 2616-2620.
- [5] Chung, W., Lee, S., & Lee, W. (2012). The Architecture and Characteristics of Web Systems. Journal of Web Engineering, 11(4), 391-412.
- [6] 李颖, 王伟. Web前端技术与架构设计[J]. 软件学报, 2017, 28(5): 1023-1035.
- [7] 赵宁, 张鹏. Web系统应用层设计与性能优化[J]. 计算机应用研究, 2016, 33(6): 1421-1425.
- [8] 王涛, 刘端. Web系统中的数据库设计与优化策略[J]. 数据库技术与应用, 2018, 21(4): 67-71.

# 全链路通关自动化测试解决方案设计与实施

邱月明、贾露翔、赵迎春、王长春 / 大商所飞泰测试技术有限公司

随着证券行业的快速发展, 通关测试的重要性日益凸显。本文详细阐述了全链路通关自动化测试解决方案的设计与实施, 分析了当前通关测试面临的核心挑战, 并针对性地提出了解决方案。通过对方案的目标、总体设计、核心模块技术实现以及实施过程的详细介绍, 展示了该方案如何有效应对证券行业通关测试的难题。同时, 总结了方案的实施成效, 探讨了其在行业内的推广价值, 并对未来的发展方向进行了展望, 旨在为证券行业的数字化转型提供有力的支持。

## 一、引言

随着证券行业业务复杂度的提升和交易规模的扩大, 传统人工或自动化测试模式已难以应对高频次、高覆盖率、高精度的通关测试需求。全链路通关自动化测试解决方案是通过构建智能化的数据选择、用例生成、跨平台执行及结果校验体系, 实现移动端、PC端到柜台端的全面覆盖的自动化测试方案。该方案依托自动化测试平台ATF结合动态行情解析、AI视觉识别等核心技术, 解决了传统自动化测试中数据交互受限、用例维护成本高、跨平台兼容性差等痛点。本文将从当前通关测试核心挑战、方案设计原理、技术实现及实施路径方面, 为证券行业通关测试的数字化转型提供可复用的方法论与实践经验。

## 二、当前通关测试核心挑战

在证券行业中, 通关测试是确保交易系统稳定运行的重要环节, 涵盖了回归测试和新业务测试, 旨在验证系统的各项功能是否符合预期。然而, 当前的通关测试面临着诸多挑战: (一) 高频核心业务回归测试压力突出

柜台系统的每次升级都至关重要, 必须覆盖核心业务委托无异常、资金持仓正确性的校验。在证

券行业, 90%以上的券商将回归测试列为最高优先级。但由于通关测试的时间窗口有限, 而测试数据又极为复杂, 需要在短时间内完成大量的测试任务, 这无疑给回归测试带来了巨大的压力。

### (二) 新业务测试时效性要求提升

交易所新业务指引下发后, 需在3个工作日内完成测试用例设计与执行, 传统用例开发周期长达1周, 导致30%的新业务测试存在滞后风险。

### (三) 跨平台测试复杂度激增

证券交易涉及多个终端, 包括柜台、外围终端(如同花顺APP、自研APP、通达信PC端等)。不同终端的控件框架存在显著差异, 导致自动化脚本复用率低于40%, 用例建设维护成本极高。这使得跨平台测试成为一项艰巨的任务, 耗费大量的人力、物力和时间。

### (四) 传统自动化测试瓶颈

在当前的通关测试需求背景下, 传统自动化测试模式无法解决以下几个难点:

1. 数据交互限制: 出于生产数据安全考虑, 通关测试自动化无法通过直接访问生产数据库进行数据实时交互, 这增加了测试数据自动生成的难度。
2. 指令申报结果的不确定性: 交易系统的复杂

性使得指令申报后的结果具有不确定性，这对测试用例的构造和测试结果的校验提出了更高的要求。

### 三、方案设计

#### （一）方案目标：

为了应对上述通关测试核心挑战，全链路通关自动化测试解决方案已实现以下目标：

#### 1. 应对高频核心业务回归测试压力突出：

**智能选择回归用例和数据：**回归测试用例和测试数据应具备灵活性，能够根据测试覆盖需要进行选择。确保需求覆盖的全面性和准确性，提高回归测试的效率和质量。

#### 2. 应对新业务测试时效性要求提升：

**新业务用例快速生成：**能够根据通关指引自动生成测试用例设计，快速实现自动化用例。通过智能化手段，大幅缩减自动化用例建设耗时，及时响应新业务需求。

#### 3. 应对跨平台测试复杂度激增：

**实现跨平台自动化测试：**基于自动化测试平台结合AI视觉技术实现所见即可用的自动化跨平台测试，大幅减少自动化脚本的建设维护成本，提高跨平台测试的效率和准确性。

#### 4. 应对传统自动化测试瓶颈：

**a.保障数据安全，风险可控：**本地测试数据池与生产库零交互，确保数据的安全可控。通过数据隔离和加密技术，为测试数据提供全方位的保护。同时，测试过程具备可控性，能够实时监控测试进度和结果，确保测试的顺利进行。

**b.精准构造目标场景：**根据行情实时达成用例场景目标，确保100%覆盖所有业务交易场景。

#### （二）总体设计：

全链路通关自动化测试解决方案的总体设计包括数据层、用例层、执行层、校验层和报告层。

**1.数据层：**负责测试数据的生成和管理，支持数据的灵活选择和智能生成。通过数据隔离和加密技术，确保生产数据的安全性。数据层为整个测试过程提供基础数据支持，是保障测试数据质量和安全的关键环节。

**2.用例层：**负责自动化用例的生成和管理，支持用例的智能生成和筛选。通过智能化设计，在高效建设自动化用例的同时，实现精准覆盖指定业务和数据。用例层是测试过程的核心，直接影响测试的覆盖范围和准确性。



3.执行层：负责自动化用例的执行和监控，支持用例的快速执行和迭代，提高测试的效率和准确性。执行层是测试过程的实施阶段，确保测试用例能够按照预期的方式运行，并及时发现系统中存在的问题。

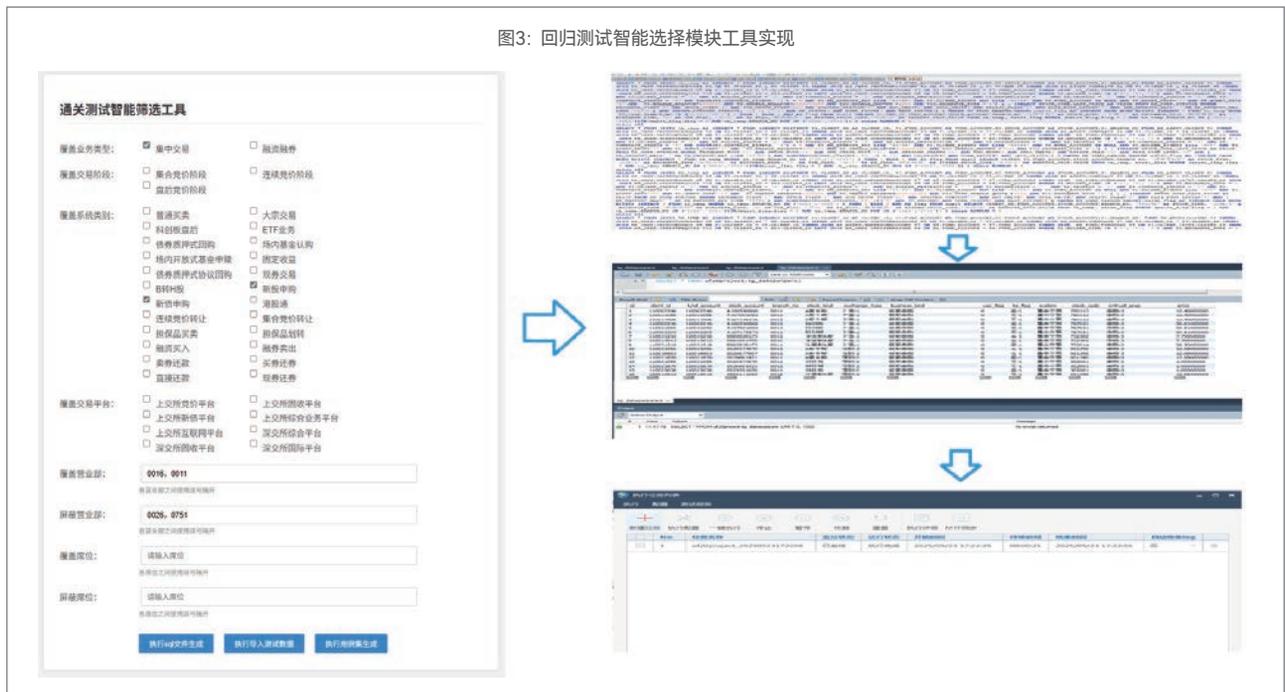
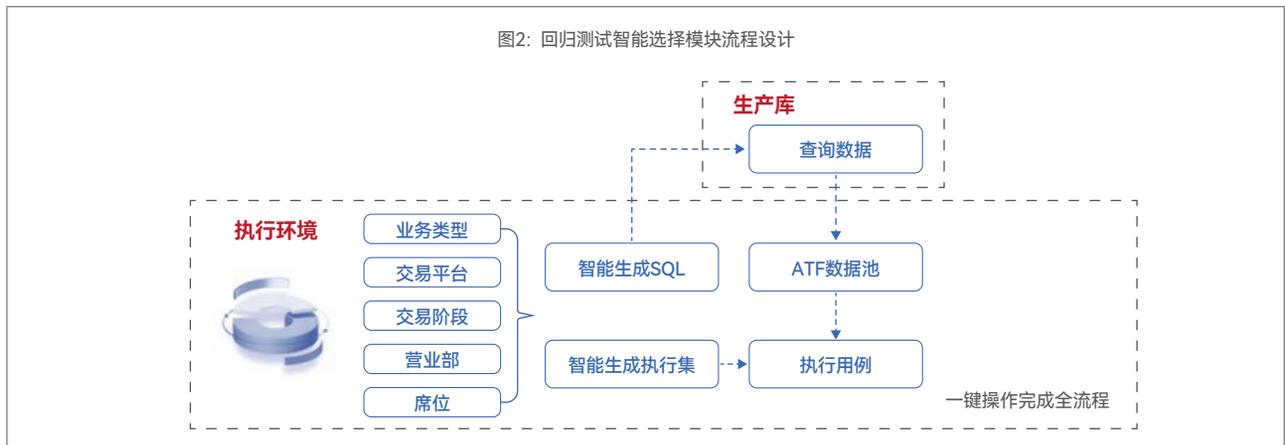
4.校验层：负责测试结果的分析校验，根据执行结果动态判断校验实际结果与期望结果的符合性。校验层是测试过程的质量保障环节，通过

对测试结果的准确分析，及时发现系统的缺陷和不足。

5.报告层：负责生成测试报告，提供业务覆盖和测试数据覆盖情况，支持用例的快速迭代执行。报告层是测试过程的总结环节，为测试人员提供直观的测试结果和分析，快速定位问题。

(三) 核心模块技术实现：

1. 回归测试智能选择模块：



a.数据隔离机制-保障数据安全，风险可控：设计执行环境数据池，保证用例执行阶段与生产库无交互，保障数据安全性。查询数据一键智能处理导入数据池，操作方便高效运行。通过数据隔离机制，有效避免了测试对生产数据的干扰，同时提高了数据处理的效率。

b.交互筛选机制-智能选择回归用例和数据：根据交互选择智能生成筛选数据sql，满足指定测试数据覆盖需求及黑名单数据抗干扰需求。同时智能生成用例执行集，更有针对性覆盖测试需求，提高测试效率。交互筛选机制使得测试数据和测试用例的选取更加精准，提高了回归测试的效率和准确性。

2. 新业务用例智能生成模块-新业务用例快速生成：

新业务测试对接AI大模型，根据分析测试指引修改统一测试业务兼容层的提示词生成获取测试数据的脚本，并进一步通过统一测试用例兼容层工具自动生成 ATF 平台自动化可执行用例，大幅缩减自动化用

例建设耗时。通过与 AI 大模型的结合，实现了新业务用例的快速生成，提高了测试的响应速度。

3.动态结果校验体系-精准构造目标场景：

a.行情解析器设计：设计行情解析器，实时获取行情信息生成证券代码、委托价格和委托数量等交易参数。

b.每个业务都设计对手方交易用例，满足成交状态全业务覆盖。同时动态获取委托状态，精准校验资金持仓变化。动态结果校验体系通过对行情信息的实时分析和对手方交易用例的设计，实现了对测试结果的精准校验，提高了测试的可靠性。

4.UI跨平台执行框架（针对外围测试PC端或APP端）-实现跨平台自动化测试，打造"数字员工"智能测试体系：

将传统的基于控件的对象识别和操作，转换为基于AI处理的图像、文字识别与操作，实现了跨平台的APP/PC自动化测试。

a.基于AI完成对象识别：摒弃传统的XPath定

图4：自动生成自动化用例举例

**背景介绍：**  
这是一个专门为证券交易系统设计的测试数据生成工具，其主要功能是根据业务规则自动生成测试数据集。  
**核心功能：**  
智能数据生成：通过精心设计的sql查询组合，自动生成符合实际业务场景的测试数据  
请根据以下要求生成可在mysql数据库执行的sql语句：  
1. 表格构造：  
a5\_manage\_tac khjyjd: 客户交易节点表  
字段: khh(客户号), stlx(系统类型)  
a5\_manage\_tac khxz: 客户信息表  
字段: khh(客户号), yzh(营业部), jgbc(机构标志), khzt(客户状态), wtf(委托方式), khxx(客户属性)  
a5\_fund\_tfc\_gjzh: 资金账户表  
字段: khh(客户号), gzh(资金账号), zzhz(账户状态), stlx(系统类型), bz(币种)  
a5\_sectrade\_tac gdh: 股票类表  
字段: khh(客户号), gdh(股票代码), jys(交易所), jydy(交易单元), gddzx(股票指定属性), gdat(股票状态)  
a5\_manage\_tac khdxsz: 客户适当性属性表  
字段: khh(客户号), fxcnsl(风险承受能力)  
a5\_sectrade\_tac zqmc: 证券代码表  
字段: zqdm(证券代码), zsp(持仓价), wsl(委托数量), wtxx(委托下限), gjvtxx(实际委托下限), zss(涨停价), zqbl(证券类别), jys(交易所), jyzt(交易状态), zqmc(证券名称), dslxxz(交易限制), zqb(最高价), zdbj(最低价), bksl(板块标识)  
2. 数据字典：  
以下每个字段的字段值都在中括号中显示，采用英文逗号进行分割  
字段: stock\_kind(证券代码类型):[A股主板,科创板]  
字段: bs\_flag(买卖方向):[买入-1,卖出-2]  
字段: entrust\_prop(交易类型):[限价-0,五档即成撤撤-1,五档即成转限价-2,本方最优-3,对方最优-4]  
逻辑要求：  
主键: 零连接cmc\_khjyjd, tac\_khxx, tfc\_gjzh, gsc\_gdh, cmc\_khdxsz表  
筛选条件：  
客户风险承受能力>1  
正常客户状态  
人民币有效资金账户  
上海交易所股东户  
非对股票  
价格有效性  
占位符替换：  
Client\_id\_placeholder需替换为实际排除的客户ID列表  
Jyzt\_00\_0001\_00\_placeholder需替换为上海交易单元号  
status\_placeholder需替换为实际委托状态码  
衍生规则：  
衍生规则基于两个字段的组合：  
根据数据字典中bs\_flag(买卖方向),entrust\_prop(交易类型)两个字段的值进行衍生  
某衍生规则如sql语句的衍生示例如下，则：  
每条原始sql个数 \* 买卖方向的字段个数 \* 交易类型的字段个数 = 每条原始记录生成衍生记录个数

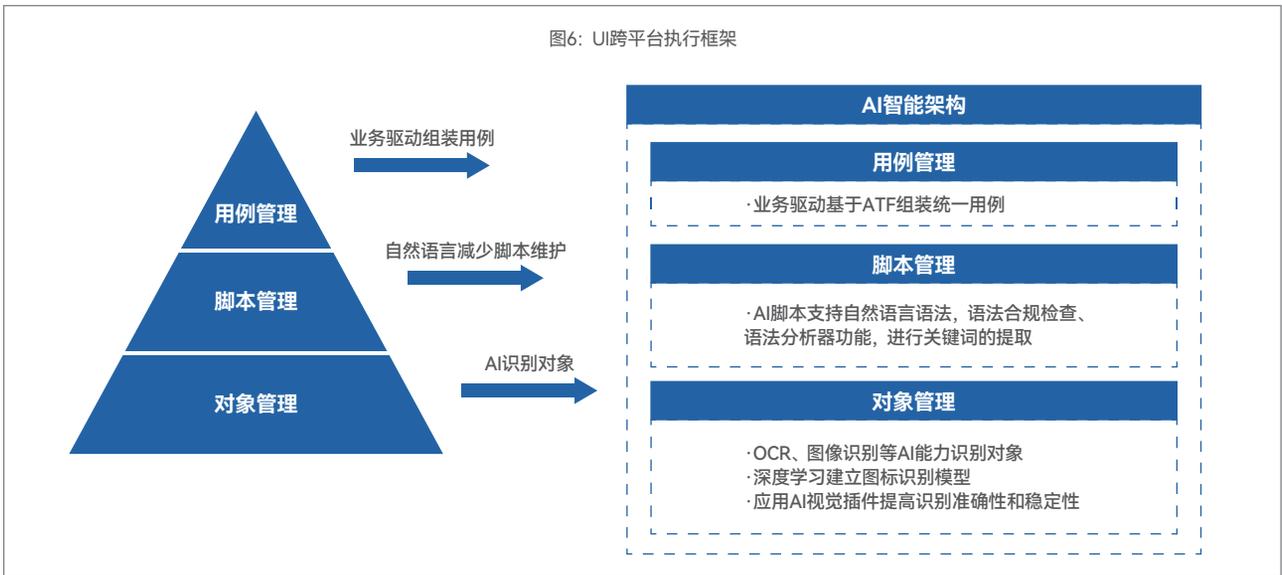
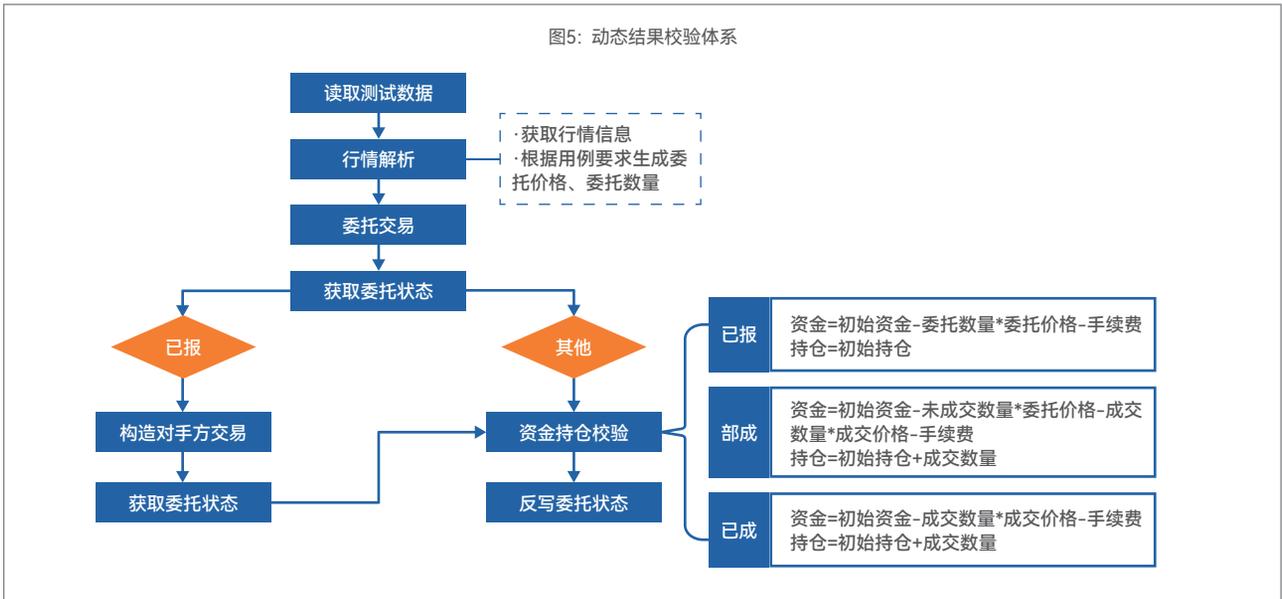
```

a5_manage_tac khjyjd: 客户交易节点表
字段: khh(客户号), stlx(系统类型)
a5_manage_tac khxz: 客户信息表
字段: khh(客户号), yzh(营业部), jgbc(机构标志), khzt(客户状态), wtf(委托方式), khxx(客户属性)
a5_fund_tfc_gjzh: 资金账户表
字段: khh(客户号), gzh(资金账号), zzhz(账户状态), stlx(系统类型), bz(币种)
a5_sectrade_tac gdh: 股票类表
字段: khh(客户号), gdh(股票代码), jys(交易所), jydy(交易单元), gddzx(股票指定属性), gdat(股票状态)
a5_manage_tac khdxsz: 客户适当性属性表
字段: khh(客户号), fxcnsl(风险承受能力)
a5_sectrade_tac zqmc: 证券代码表
字段: zqdm(证券代码), zsp(持仓价), wsl(委托数量), wtxx(委托下限), gjvtxx(实际委托下限), zss(涨停价), zqbl(证券类别), jys(交易所), jyzt(交易状态), zqmc(证券名称), dslxxz(交易限制), zqb(最高价), zdbj(最低价), bksl(板块标识)

```

Case Name	操作	交易所	委托方式	委托数量	委托价格	委托状态	成交数量	成交价格	成交状态
集中交易, 普通买卖, 上海, 买入, 限价, 已报	client Fund_stockbravo:stockshubuzhuanshu_bu_flagzqstocentn_gjocentn_trdgzstata	200A	1	10000	10.00	0	10000	10.00	0
集中交易, 普通买卖, 上海, 买入, 限价, 已报	##### A0091	200A	1	10000	0	0	0	0	0
集中交易, 普通买卖, 上海, 买入, 限价, 已报	##### A241	200A	1	10000	0	0	0	0	0
集中交易, 普通买卖, 上海, 买入, 限价, 已报	##### A0092	200A	1	10000	0	0	0	0	0
集中交易, 普通买卖, 上海, 买入, 限价, 已报	##### A0096	200A	1	10000	0	0	0	0	0
集中交易, 普通买卖, 上海, 买入, 限价, 已报	##### B476	200A	1	10000	0	0	0	0	0
集中交易, 普通买卖, 上海, 买入, 限价, 已报	##### A436	200A	1	10000	0	0	0	0	0
集中交易, 普通买卖, 上海, 买入, 限价, 已报	##### A010	200A	1	10000	0	0	0	0	0
集中交易, 普通买卖, 上海, 买入, 限价, 已报	##### A514	200A	1	10000	0	0	0	0	0

ATF 平台界面截图显示了用例列表，包括用例名称、操作、交易所、委托方式、委托数量、委托价格、委托状态、成交数量、成交价格、成交状态等列。



位，OCR技术来提高识别准确性和稳定性，可视AI解决安全加固等问题。

b.基于AI完成转换操作：通过约定的自然语言语法提取关键字使得AI脚本编写更加准确，维护成本更低，兼容性更高。

c.基于业务驱动，自动组装用例，快速创建UI自动化用例。

#### 四、方案实施

制定严格安全的方案实施策略，保证部署环境与被测环境隔离，仅能通过网络端口连接。网络仅在周末交易所升级测试期间开通，其他时间处于关闭状态，通过端口检测验证。

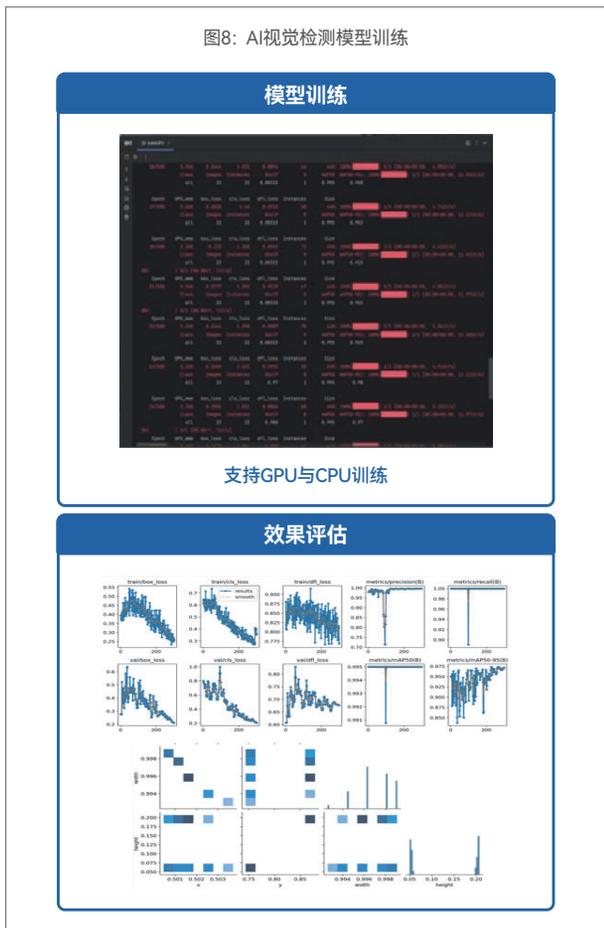
##### (一) 环境部署：

1. 通关自动化测试平台部署：通关自动化测试

图7: AI视觉识别对象

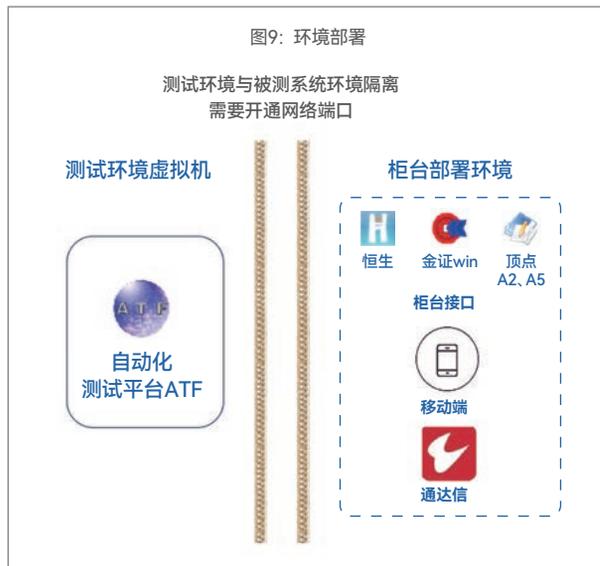


图8: AI视觉检测模型训练



平台部署在客户内网虚拟机，同时与生产环境进行网络隔离，只有在周末通关测试期间开通网络端口。这种部署方式有效保障了生产环境的安全，同时满足了通关测试的需求。

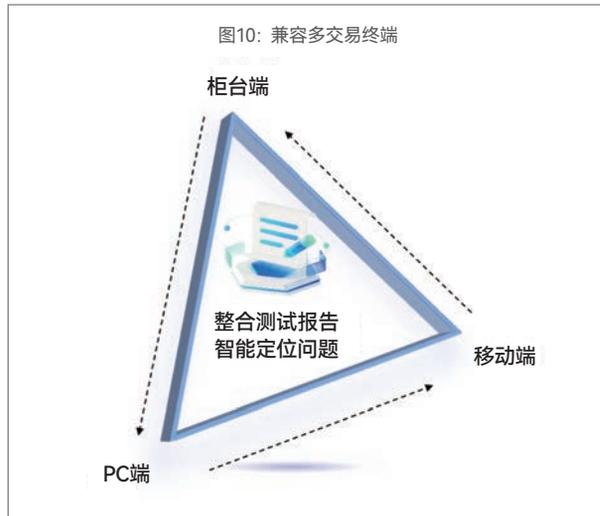
图9: 环境部署



2.部署"数字员工"智能测试体系:

a.通关自动化测试平台可兼容多个交易终端:  
可同时部署包括柜台、外围终端（如同花顺 APP、自研 APP、通达信 PC 端等）多套用例库，为通关测试提供全方位质量保障。

图10: 兼容多交易终端



b. 通关自动化测试平台可智能整合测试报告：可对多个交易终端用例库执行报告进行整合，通过对不同交易终端的测试报告进行分析比对，智能区分接口和UI失败原因，快速定位问题。

### （二）测试步骤：

全链路通关自动化测试系统提供智能交互界面，根据客户需求，智能选择测试数据和场景，操作简单，覆盖全面，高效运行

1. 步骤一，选择测试范围：测试前，测试人员确定本次通关测试的测试范围，包括业务范围和数据范围。通关测试时，在智能交互界面可从业务类型、交易平台、交易阶段、营业部、席位多维度确定覆盖范围。

2. 步骤二，执行测试：交互界面确定覆盖范围后，点击执行，测试系统会智能筛选数据和用例，完成本次通关测试执行工作。执行期间，测试系统提供监控界面，对用例执行进行实时监控。

3. 步骤三，生成测试报告：用例集执行完毕后，生成测试报告，对每条用例的所有步骤，测试数据和交易状态都有详细描述和分析，节省人工分析报告时间。

### （三）方案优势：

该方案目前已在23家券商实际落地，多家头部券商的通关测试实际应用中，展现出以下显著优势：

1. 全链路覆盖与精准测试：方案实现了从移动端(APP)、PC端到柜台端的无缝覆盖，并依托动态行情解析器与对手方交易用例设计，精准构造复杂交易场景（如特定行情下的委托成交状态），确保业务场景覆盖率提升至80%以上，资金持仓校验精度达100%，有效捕捉环境、数据、系统等多维度潜在问题。

#### 2. 智能驱动与高效响应：

a. 回归测试智能化：通过交互筛选机制，智能生成筛选SQL与用例执行集，精准覆盖指定测试数据（包括黑名单抗干扰）和业务需求，显著提升回归测试的针对性与效率。

b. 新业务敏捷化：借助AI大模型自动生成测试数据脚本与自动化用例，将新业务自动化用例建设周期从传统的一周大幅压缩，有力保障了3个工作日内完成测试的时效性要求。

#### 3. 安全可控与跨平台兼容：

a. 数据安全隔离：严格执行本地测试数据池机制，与生产数据库零交互，并通过网络端口仅在测试窗口期开通等策略，构建多重安全防护，确保生

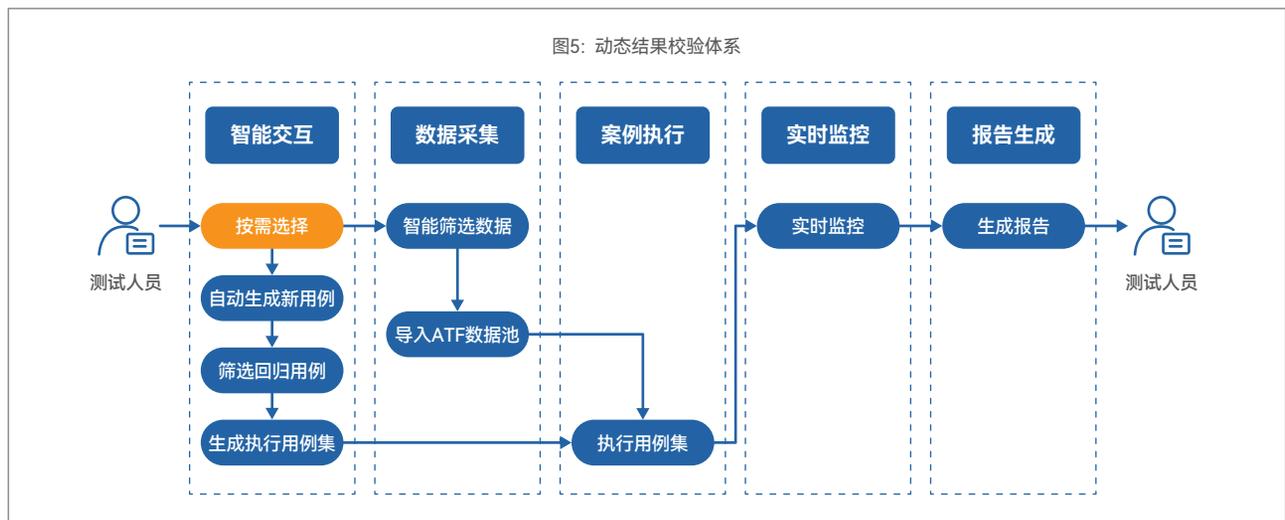


图5: 动态结果校验体系

通关自动化测试结果分析报告						
测试环境: 仿真环境 测试数据日期: 2025年04月01日 测试平台: 恒生uf20系统 测试执行情况: 1.总用例数: 1010条。 2.本次测试执行用例: 1010条用例, 总通过率为99.7%。 3.本次测试发现缺陷: 本次测试发现0个缺陷, 本轮测试引入0个缺陷。 4.本次测试效率汇总: 执行机1台4线程, 执行时间0.5小时;						
	用例总数	执行数	成功数	失败数	错误数	N/A数
总计	1010	1010	1007	1	2	0
结果分析汇总						
序号	用例状态	问题类型	问题详情	解决方案	影响用例数	
1	失败	环境问题	质押入库柜台抵押比率与交易所的抵押比率不一致		1	
2	错误	环境问题	B转H用例委托状态为1待报		2	

产数据绝对安全, 风险全程可控。

b. 跨平台统一执行: 基于AI视觉识别技术(OCR、图像识别)构建的UI跨平台执行框架, 摒弃了对特定控件框架的依赖(如XPath), 实现了“所见即可测”, 将脚本复用率提升至85%以上, 大幅降低不同终端(如柜台、同花顺APP、自研APP、通达信PC端)自动化脚本的建设与维护成本(降低60%)。

4. 高效执行与无人值守: 方案提供智能交互界面, 简化测试范围选择与执行流程。自动化执行引擎支持快速迭代, “数字员工”模式实现无人值守测试。智能报告层整合多终端测试结果, 自动分析失败原因, 快速定位问题, 将人工分析报告时间降至最低。

## 五、总结与展望

### (一) 方案实施成效总结:

本方案通过构建“数据-用例-执行-校验-报告”五层联动的全链路测试体系, 系统性解决了证券行业通关测试面临的高频回归压力、新业务时效滞后、跨平台兼容性差等核心痛点。

### (二) 行业推广价值:

作为证券业数字化转型的创新实践, 本方案首次实现接口-PC-APP三位一体的无人化测试闭环,

构建了可复用的“数字员工”测试范式。其分层架构设计支持快速适配不同券商的柜台系统, 动态数据策略可扩展至基金、期货等泛金融领域。

### (三) 未来发展方向:

面向资本市场全面注册制改革与跨境业务拓展趋势, AI人工智能的不断发展和优化, 后续将重点突破以下领域:

1. AI测试引擎升级: 集成AI大语言模型实现自然语言驱动测试编排, 通过自然语言自动生成复杂业务场景, 进一步提高测试覆盖度和减少人工成本;

2. AI测试报告智能分析: 基于AI大模型, 进一步精确“数字员工”测试报告的分析功能, 精准定位问题所在, 帮助人工快速识别缺陷;

3. 提升“数字员工”稳定性: 基于柜台接口-PC端-APP端三位一体的设计, 为更好实现“数字员工”全方位精准完成通关测试的工作, 进一步提升基于AI“数字员工”的稳定性和精准度, 实现无人通关测试更高的安全和效能要求。

本方案将持续迭代, 致力于成为证券业新型基础设施的智能测试基座, 为资本市场深化改革提供全天候、全场景的数字化质量保障能力。

责任编辑: 汪璇璇

# “大模型 + MCP” 在软件测试中的实践

余俊彦、张世同、李娴、杜瑞罡、李立峰 / 广发证券股份有限公司

## 一、背景

伴随数字化进程的高速推进，软件系统的规模与复杂度呈指数级增长态势。传统软件测试方法在应对当下复杂多变的软件环境时，逐渐暴露出效率欠佳、复杂场景覆盖不足以及对新兴技术适应性薄弱等问题。这些弊端不仅致使测试周期延长、成本攀升，还可能遗漏关键缺陷，对软件质量形成潜在威胁。与此同时，大模型技术的兴起以及MCP（Model Context Protocol，模型上下文协议）等新兴技术的出现，为破解长期困扰软件测试领域的难题带来了新的契机。本文将深入探究如何借助“大模型 + MCP”构建更为高效、智能的软件测试解决方案。

## 二、大模型与 MCP 技术的剖析

### （一）大模型：智能测试的新动力

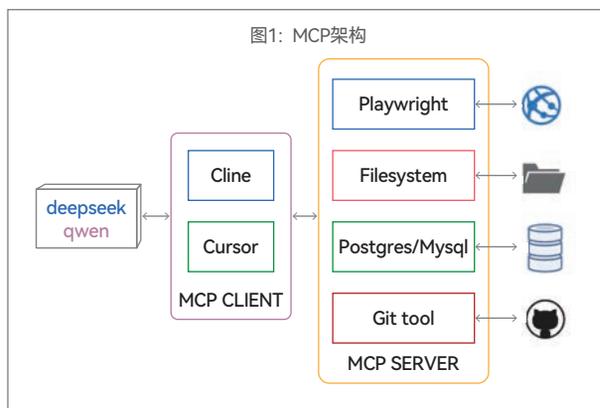
大模型凭借其卓越的自然语言处理能力以及对海量数据的学习优势，在软件测试领域展现出巨大的应用潜力。它能够迅速理解测试需求文档，自动生成测试用例，同时还可对代码及测试结果进行智能分析与总结。例如，在处理包含复杂业务逻辑的测试需求时，大模型能够通过历史项目数据和行业知识的学习，精准识别关键测试点，大幅提升测试用例的全面性与准确性。

### （二）MCP：大模型的强大“外脑”

MCP（Model Context Protocol，模型上下文协议）是Anthropic于2024年11月推出的开源协议。其设计理念旨在构建一种通用通信规则，助力AI助手实现与外部系统的实时连接。MCP的标准化设计，为开发者创建安全的双向接口提供了便利，使得AI能够顺利访问所需数据。该协议致力于消除AI助手

在数据孤岛中面临的局限性，拓展其在复杂任务处理中的能力。

Anthropic发布的官方标准文档<sup>[1]</sup>对MCP进行了详尽阐述，并提供了协议规范和SDK，以支持开发者快速构建MCP服务（MCP Server）或客户端（MCP Client）。标准文档着重强调了MCP的模块化与可扩展性，使其能够灵活适配Git、Slack、Google Drive和Postgres等各类外部工具。借助MCP，大模型得以突破自身数据局限，实时获取外部最新信息。在软件测试场景中，大模型可借此获取最新代码变更、业务数据、测试环境配置等关键信息，为智能测试提供更丰富的数据支撑。



## 三、“大模型 + MCP”辅助测试的实践

### （一）架构搭建

广发证券已私有化部署多款开源大语言模型，通过API为内部提供访问服务。在MCP client方面，可采用如Cline、Cursor等工具，构建起大模型与MCP Server交互的关键通道，为后续测试实践奠定了基础架构。

### （二）应用场景

1.访问wiki分析需求 (MCP: playwright) : 在开展新功能测试时, 测试人员可借助大模型调用Playwright MCP访问公司内部Wiki。在此过程中, 大模型能够自动提取相关技术文档与历史项目经验, 快速分析出需求中的关键信息及潜在风险点, 为后续测试用例设计提供有力支持, 助力测试人员精准把握测试重点。例如, 在测试新的金融交易功能时, 运用该方式, 测试人员可迅速从 Wiki 中获取以往类似交易功能的测试经验, 涵盖常见问题及解决方案, 进而更好地设计针对新功能的测试用例。

2.分析需求与代码实现是否匹配 (MCP: playwright+git) : 大模型通过Playwright MCP精准获取需求内容, 同时借助Git MCP执行git diff操作, 获取迭代相关的代码变更信息。随后, 大模型对需求内容与代码变更逻辑进行细致比对, 能够快速判断需求与实现是否一致, 有效减少因需求与代码不一致引发的缺陷, 提升软件质量。例如, 当需求明确规定某个交易规则的计算方式时, 大模型通过对比代码变更中的计算逻辑, 可及时发现其中存在的偏差。

3.代码变更分析 (MCP: git) : 通过Git MCP获取代码变更记录, 大模型对变更内容进行深入分析, 能够准确判断变更可能影响的功能模块和测试范围。这一分析结果帮助测试人员快速制定针对性的回归测试计划, 提高回归测试的效率和准确性, 确保软件变更后的稳定性。

4.变更代码与数据库联合分析 (MCP: git+postgres) : Git MCP获取代码变更信息, Postgres MCP从数据库中提取相关数据结构和数据记录。大模型将两者结合进行综合分析, 全面评估代码变更对数据库操作的影响, 确保数据库相关功能在代码变更后的正确性, 避免因代码变更引发数据库层面的问题。

5.自动发现代码缺陷 (MCP: git+filesystem) : 大模型借助Git MCP, 通过git diff分析代码变更差

异, 同时从Filesystem MCP获取上下文相关的代码文件。基于对这些信息的综合分析, 大模型能够自动定位代码中的潜在缺陷, 有效提升代码质量, 降低缺陷在后续开发和测试过程中出现的概率, 节省修复成本。

6.生成测试用例 (MCP: git+wiki+filesystem) : 大模型借助MCP, 从Wiki上提取需求文档, 并获取Git中的代码变更信息。基于这些丰富的数据, 大模型进行整合分析, 生成全面且高质量的测试用例。随后, Filesystem MCP将生成的测试用例写入本地, 极大地提升了测试用例生成的效率与质量, 为软件测试提供了充足且精准的测试依据。

7.反推代码逻辑, 定位功能 (MCP: git+filesystem) : 当测试人员需要了解某个功能的实现逻辑时, Git MCP从代码仓库拉取最新代码, Filesystem MCP提供代码相关文件。大模型通过对这些资料的深入分析, 能够反推代码逻辑, 并准确定位到实现该功能的具体代码模块, 帮助测试人员快速理解软件功能实现细节, 提高测试工作的针对性。

### (三) 实操演示

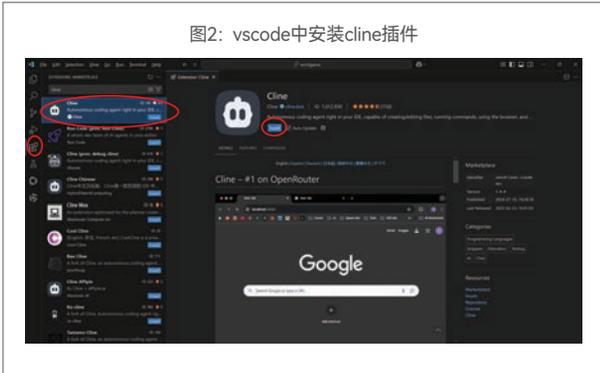
为了详细地描述我们如何利用大模型+MCP发现代码缺陷, 我们以Cline作为MCP客户端, 对上述“自动发现代码缺陷”为例来进行演示。

1.本地git环境准备: 需确保本地已完成 git 环境的安装与配置。git作为广泛应用的分布式版本控制系统, 是后续操作的基础。若尚未安装, 可访问 git 官方网站, 根据系统类型下载对应安装包进行安装。安装完成后, 通过命令行工具输入“git--version”验证安装是否成功。在安装过程中, 应严格按照系统提示完成环境变量配置等操作, 以保障git在系统中正常运行。

2.安装Cline插件: 打开VSCode编辑器, 在插件市场搜索“Cline”, 找到对应插件后点击安装。

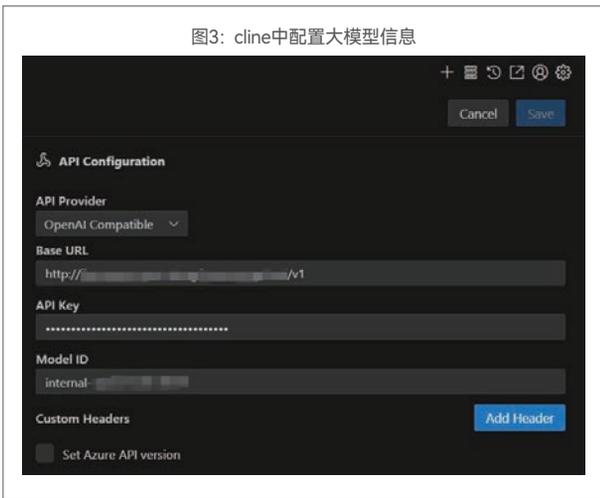
Cline插件作为MCP客户端，是实现后续功能的关键组件。安装过程中，VSCode将自动下载并安装相关依赖，安装完成后，可能需要重启VSCode以使插件生效。

图2: vscode中安装cline插件



3.配置Cline关联大模型：打开Cline插件的配置页面，找到与大模型连接相关的设置项。依据大模型提供方的说明，准确填写API密钥、服务器地址等必要信息，确保Cline与大模型建立稳定连接。

图3: cline中配置大模型信息



4.安装并配置Git MCP：进入Cline的MCP市场，搜索“Git Tools”进行安装。此安装方式依赖大模型自动分析所需安装的依赖，存在一定的安装失败几率。若自动安装未成功，可前往该MCP的GitHub页面，参照相关说明进行手动安装。

图4: cline中安装Git MCP



图5: Git MCP代码路径等配置



5.与大模型进行会话：完成上述所有配置后，通过Cline插件的操作界面，发起与大模型的会话请求。

图6: 大模型通过Git MCP获取代码变更内容



会话示例1：“应用service当前分支feature-\*\*\*与master比较，总结变更内容。”大模型将自动调用Git MCP，获取git diff内容进行变更分析，并输出变更内容及测试建议。大模型在分析变更内容时，会从代码逻辑、功能影响等多维度考量，针对关键变更点给出详细且具针对性的测试建议，如建议针对某些变更增加特定场景的测试用例。

图7: 大模型输出变更总结内容

```

Task Completed
已通过MCPIops_大模型工具对Lang项目中的feature-***分支与master分支的变更，主要修改内容如下：

1. AgreementTemplate.java
  - 新增字段: effectDate (生效日期) 及其对应的getter方法(name = "effect_date")注册
  - 新增Setter/Getter方法:
    public String geteffectDate() { ... }
    public void seteffectDate(String effectDate) { ... }
  
```

图8: 大模型输出测试建议

```

Task Completed
根据代码变更内容，以下接口路径需要重点测试：

1. 新增行测试
  - POST /api/v1/template: 验证新增 effectDate 字段的逻辑
  - PUT /api/v1/template/approve: 验证新增的审批生效日期字段的逻辑
  - POST /api/v1/template/clone: 验证新增 effectDate 字段的逻辑
  - PUT /api/v1/template (带 templateId 参数): 验证修改审批生效日期的逻辑
  
```

会话示例2：“分析应用service本次变更是否存在缺陷。”

从大模型的思维链输出可见，大模型利用Git MCP获取的Git diff信息，结合Filesystem MCP读取的与变更相关文件进行分析，输出缺陷具体信息：

缺陷一：某条件判断时误将逻辑表达式中的“与”(&&)运算符误写为“或”(||)。由于逻辑矛盾，导致该条件恒为true，进而使所有配置项被错

图9: 大模型输出代码缺陷内容

```

1. 逻辑判断错误 (静态缺陷)
文件: ..._java 和 ..._java
- 缺陷位置:
  // ..._java 中的 handle... 方法
  if (condition.getEventType() != 1 || condition.getEventType() != 2) { // 应为 && 运算符
    logger.info("配置项");
    ...
  }
- 缺陷描述:
  条件表达式: condition.getEventType() != 1 || condition.getEventType() != 2 恒为 true (因为一个值不可能同时等于1和2)，导致所有配置项被错误地处理。
  期望描述: 应为 && 运算符。
  if (condition.getEventType() != 1 && condition.getEventType() != 2) {
  }
  
```

误跳过，某功能完全失效。大模型发现此类缺陷时，会明确指出缺陷所在代码位置、错误原因及可能产生的影响，便于开发人员快速定位与解决问题。

缺陷二：某获取上一交易日的sql语句因条件方向错误，导致无法获取上一交易日数据，最终致使指定日期区间的订单无法获取。大模型还提供修复建议，如给出该sql语句正确的条件表达式写法。

图10: 大模型输出代码缺陷修复建议

```

2. SQL 查询语句缺陷 (中风险)
- 缺陷位置: ..._java 的 ... 方法
- 缺陷描述:
  select * from ...
  where ... <= ... order by ... desc
  ...
- 修复建议:
  请检查 SQL 查询语句中的条件表达式，确保使用正确的比较运算符。原代码中的 ... <= ... 应改为 ... >= ...。
  修复后的 SQL 语句如下:
  select * from ...
  where ... >= ... order by ... desc
  ...
  
```

通过大模型对代码的变更分析，能够快速识别逻辑矛盾问题，显著提升测试效率：

无需构造复杂场景：人工测试需模拟特定场景才能复现功能失效问题，而大模型通过逻辑推理直接分析代码，无需实际运行即可发现条件矛盾，节省场景搭建和调试时间。例如，人工测试可能需花费大量时间设置特定日期、订单状态等场景来验证功能，大模型则可直接通过代码分析发现问题。

精准定位与修复建议：大模型不仅能快速定位矛盾的条件语句，还会直接提供修正后的代码示例，帮助开发人员快速理解问题根源并完成修复，提升修复效率。

提前拦截风险：传统测试可能在功能上线后才暴露此类问题，而大模型在代码评审或静态分析阶段即可发现，有效避免缺陷进入生产环境，降低生产事故风险，保障软件系统稳定性。

#### 四、经验总结与展望

大模型与 MCP 的结合在测试领域展现出显著优势，实现了测试流程的智能化升级。实际应用数据表明，引入“大模型 + MCP”后，测试效率较传统

# Model Context Protocol

测试方式提升了约20%–30%，具体提升幅度因项目复杂度而异。在某项目迭代中，得益于大模型对需求以及代码变更的分析，原本需耗时两天完成的测试用例设计工作，借助该技术组合仅用一天便高质量完成。同时，缺陷发现率提高了30%–50%，许多在传统测试中易被忽视的逻辑错误、边界条件问题等均被提前发现。例如，在代码缺陷自动检测场景下，大模型通过分析代码变更及上下文文件，成功发现了因逻辑表达式错误导致的功能异常问题，这些问题在以往测试中通常要到系统集成测试阶段才会暴露。

大模型能够实时获取代码变更和需求文档等数据，避免人工信息收集的低效性；其对自然语言和代码逻辑的深度理解能力，与MCP的访问能力相结合，可覆盖更复杂的业务场景，精准识别需求与代码间的逻辑差异，减少人为疏漏；通过分析代码变更及上下文文件，能够定位缺陷根源并提出修复建议，降低排查成本；此外，该结合打破了产品、开发与测试团队间的协作壁垒，促进信息对齐与高效沟通。

未来，我们将持续深入研究“大模型 + MCP”技术组合在系统测试领域的应用，探索更多可能

性。我们设想了以下演进路径，以推动技术架构的发展并融入CI/CD和DevOps流程：

**与CI/CD流程集成：**开发人员提交代码后，流水线会自动触发AI调用多个MCP组合。这些组合将执行需求及代码变更分析，向测试人员输出变更内容总结、测试建议以及测试用例，同时对变更代码进行缺陷排查。

**与DevOps工具链融合：**当大模型通过MCP与代码、数据库、流水线、容器云、网络、存储等关键环节实现全面打通后，运维工作将迎来重大变革。以发版流程为例，借助大模型的能力，运维人员可在发版前开展全方位的升级预演，提前洞察潜在问题，精准制定应对策略，从而有效降低发版风险，为系统稳定运行提供有力保障。

通过持续的技术优化和流程整合，我们期望为软件测试行业带来更多创新与突破。这将有力推动软件测试向更加智能、高效的方向迈进，为软件开发全生命周期提供更坚实的质量保障。🔗

责任编辑：王晓光

参考文献

[1] modelcontextprotocol. <https://modelcontextprotocol.io/introduction>

# 基于设计稿驱动的智能测试

龚柳华、谷丽丽 / 国泰海通证券股份有限公司

在证券行业数字化转型加速的背景下，证券交易客户端的测试质效提升工作愈发急迫。为了解决测试资源与需求迭代速度之间的逐渐不匹配问题，本文提出一种基于设计稿驱动的智能测试框架GUIPilot，针对证券交易客户端测试场景构建了“理解-执行-验证”的全链路自动化解决方案。本文创新性地提出从设计稿源码中直接提取界面逻辑与界面说明，生成一系列操作指令，并调用大模型将其整合成测试用例规范格式；利用adb (Android Debug Bridge) 工具，在电脑上驱动客户端使用命令模式自动执行操作指令；最后，实现了对界面图片和设计稿图片的界面一致性对比。在理想条件下，本研究验证了将前端测试流程全部交给程序完成的可行性，为界面智能化测试研究方向提供了可复制的技术路径，对金融科技领域智能化测试体系建设具有重要参考价值。

## 一、引言

在证券行业数字化转型加速的背景下，证券交易客户端作为直接承载用户交易操作的核心系统，其软件质量已成为影响用户体验与业务稳健发展的关键要素。随着金融产品创新节奏加快和交易场景复杂化，交易客户端的业务逻辑复杂度和功能迭代速度不断提升，对测试团队的专业能力和执行效率带来极大挑战，主要问题包括：

1. 测试用例设计需要大量人力：对于新需求，测试人员依据设计稿和需求文档理解业务逻辑，并据此编写测试用例，除需要理解设计稿中体现的逻辑外，还要考虑到常见的交易规则，例如买卖申报数量、价格、种类等方面的控制。

2. 重复的测试操作导致效率低下：对于复杂的功能，传统的测试方法往往需要测试人员对于同一个功能点重复操作客户端进行功能验证，这种测试方式耗费了大量人力资源，严重影响整体效率。

3. 金融类客户端测试需要较强的行业知识储

备：由于证券类软件的金融属性，其对质量的要求近乎严苛，测试人员除了需要精通常规的测试技能，还需要掌握交易业务知识，在设计和执行测试用例时，需要考虑众多交易业务细节，对测试人员的知识储备有要求较高。

在测试领域，为了提升测试用例的生成效率与质量，业界做了很多尝试，例如通过接口文档自动生成接口测试用例，有效提升了接口测试效率。大模型技术出现之后，很多互联网公司以及证券公司开始尝试运用大模型的文本分析能力实现基于需求文档的测试用例自动生成。经过调研，目前尚未发现有通过设计稿图片生成测试用例的实践。在客户端自动化测试方面，行业内已有成熟的解决方案，但基本都需要测试人员提前录制或编写自动化脚本，当界面发生变化时需要及时变更自动化脚本，维护成本较高，基于设计稿自动生成用例并自动驱动测试设备执行的方式目前还未见到。

基于上述背景，我们提出了GUIPilot测试框架，

该框架能够智能解析设计稿中的界面元素与交互逻辑，自动将其转化为具体的操作指令（例如页面滑动、控件点击、表单输入等），实现用例生成及执行的自动化；基于机器视觉技术自动对比需求设计稿上的界面和客户端程序界面。本文主要介绍了该框架的设计思路和实际应用情况。

## 二、基于设计稿驱动的智能测试建设方案

### 2.1 总体设计

图1是本项目的总体设计思路，具体而言，首先从原始设计稿文件中提取出节点及跳转逻辑，将其转换为程序能理解的“图”，在转换过程中加入修正策略，自动规范并修正设计稿。在此基础上，一方面将提取出的“图”通过大模型技术处理成遵循一定格式的测试用例；另一方面，客户端提测后，通过执行机驱动程序遵循上一步生成的“图”驱动执行机自动执行测试用例，在每一个执行节点（即界面）处进行截图，通过目标检测技术检测界面上的重要控件，并提出最优控件匹配算法来进行界面匹配与一

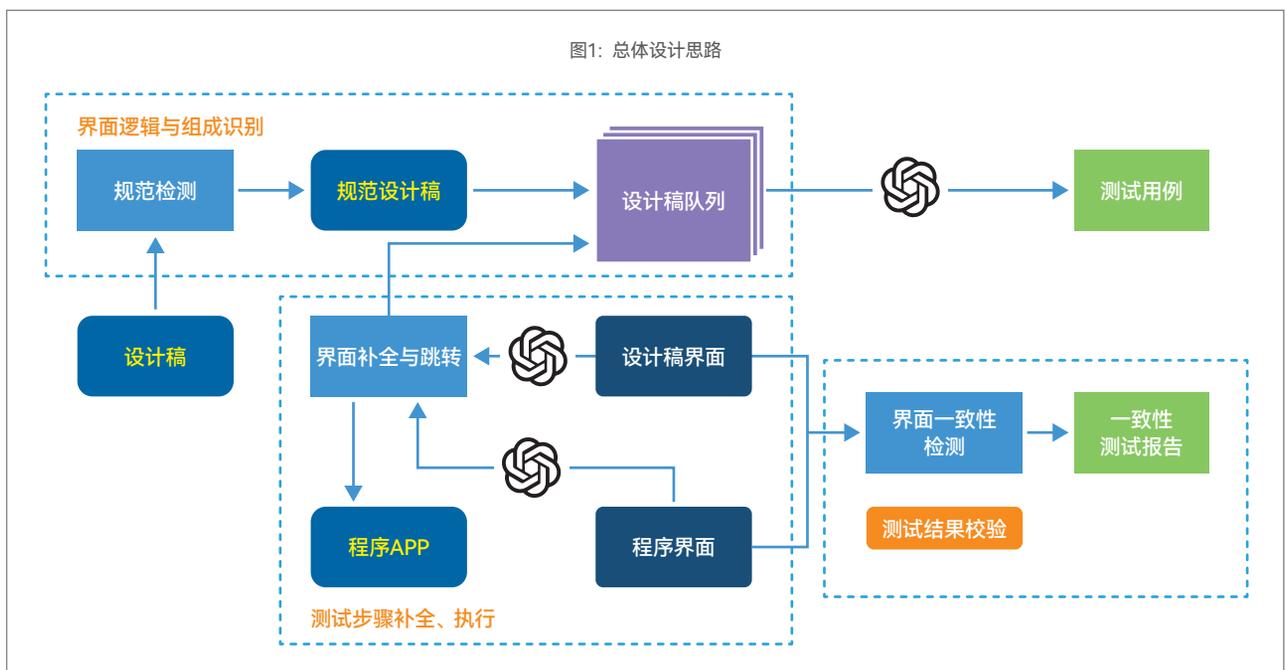
致性检测工作。此外，我们引入了视觉语言模型技术预测并补充测试用例，旨在将设计稿中的自然语言描述和视觉界面翻译成一系列可操作的指令（下滑、按钮点击、文本输入等）。最后，程序自动将测试结果以测试报告的形式输出给测试人员。

### 2.2 技术问题

#### 2.2.1 客户端开发测试流程概述

在客户端软件的开发与测试过程中，需求设计稿扮演着至关重要的角色。一份详尽的交互设计稿，不仅细致描绘了页面的布局与视觉设计，还清晰地阐述了界面与功能之间的跳转逻辑，它要求界面元素全面展现需求中的所有功能点，包括但不限于控件的布局、类型、字体大小及色号等细节。

在客户端软件的开发阶段，开发工程师依据需求设计稿进行编程实现，确保客户端的交互逻辑与设计稿中的交互逻辑保持一致，同时使程序界面尽可能还原设计稿中的设计效果。在测试环节，测试人员则根据设计稿深入理解需求，并据此编写测试



用例。在执行这些测试用例时，测试工程师会按照测试用例的操作步骤操作程序界面，检查程序的交互逻辑以及界面展示是否与设计稿相符。

## 2.2.2 测试人员主要面对的任务

### 1. 遵循稿件产生测试用例

首先，测试人员需要人工分析各控件是否实现了对应的逻辑限制，界面跳转是否满足交互逻辑等。在这个过程中，公司需要投入大量人力分析设计稿并基于此设计测试用例。因此，存在人力成本投入高、强依赖测试人员经验。经验不足的设计人员容易产生用例覆盖不全面等问题。

### 2. 遵循稿件比对界面设计

客户端测试的另一个关键环节是确认最终实现的用户界面（简称UI，User Interface）是否与设计稿相符，目前主要依赖测试人员的人工比对完成，对于一些字体大小、色号、排版等测试场景，在判断上存在一定程度的主观性，问题经常不能准确及时发现，流转设计师走查环节才拦截，导致问题发现时间滞后，影响需求交付周期。

## 2.3 技术方案

如图2所示，GUIPilot利用双框架架构（设计稿操作指令翻译技术和界面一致性比对技术）来支撑手机端用例生成、执行与结果比对，后端核心功能模块包含设计稿处理模块、控件检测与对比模块和测试执行模块。后端服务通过与浏览器、客户端程序建立websocket连接进行通信，本文主要介绍手机客户端场景，未来自动执行用例功能可拓展到更多终端设备。手机连接模块部署在与手机连接的电脑上，将读取到的手机信息发送给后端处理，并接收来自后端的命令操控手机。

## 2.4 模块划分

系统分为5个模块，包含后端、前端、目标检测服务、ocr服务和大模型服务，其中后端服务进一步细分为设计稿处理模块、控件检测与对比模块和测试执行模块等子模块。各模块之间的连接关系如图3所示：

### 2.4.1 后端服务介绍

#### (1) 设计稿处理模块

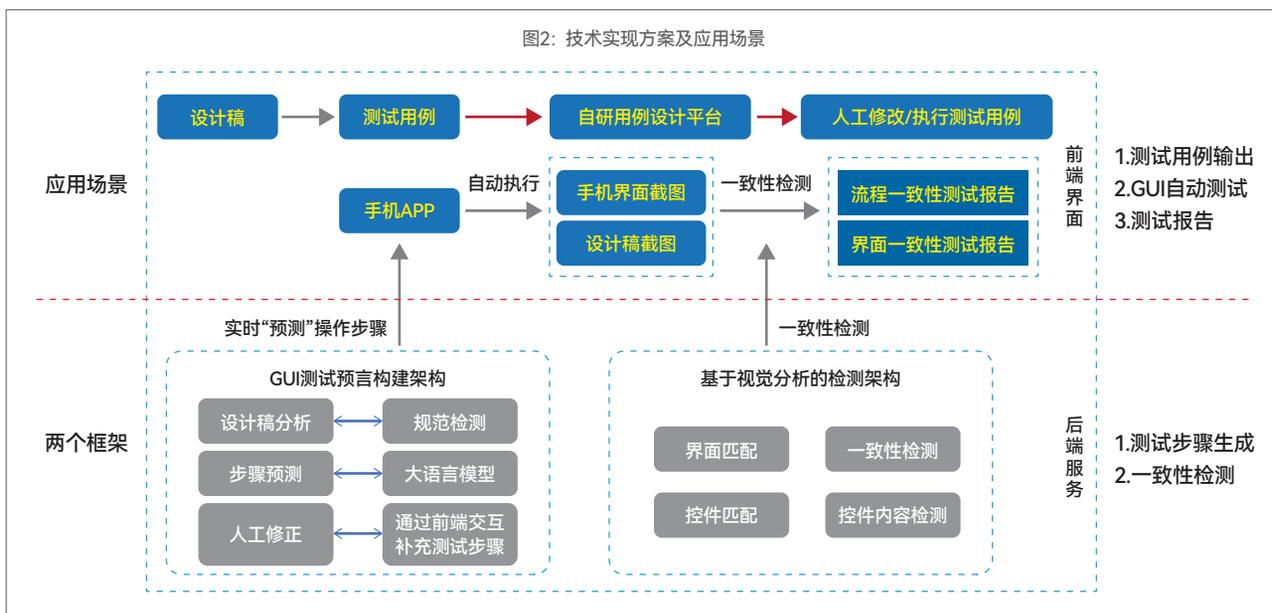
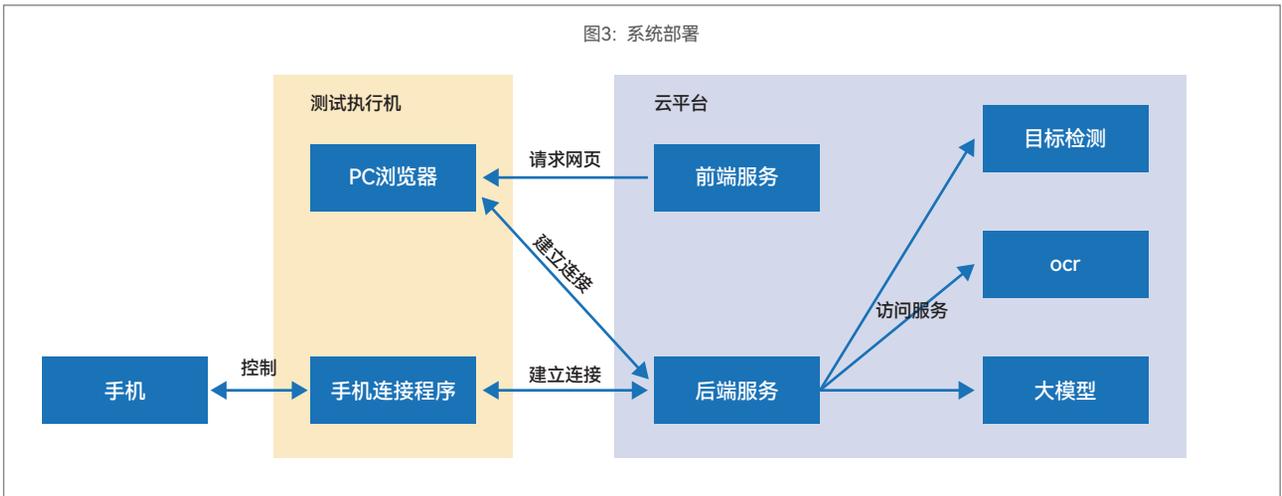


图3: 系统部署



设计稿处理模块负责将设计稿文件信息转换为内部数据结构，基于该模块，可开展后续的测试用例自动执行工作，也可直接导出测试用例。此外，该模块具备对不规范的设计稿进行提示的功能，并支撑一定程度的自动修复。目前，项目针对以下不规范的设计稿场景进行提示：

- a.设计稿条件判断框中没有具体内容
- b.设计稿中的连接线没有对应的说明文字
- c.设计稿界面没有标题

此外，项目对部分场景进行了自动修复。如图4所示，该设计稿中的界面一、界面二之间没有连接线相连，界面一没有出边，界面二没有入边，经过观察

图4: 可自动修复的“不规范”设计稿场景



图5: sketch文件解压后的目录

```

$ ll
total 344
-rw-rw-r--@ 1 145K 11 30 1979 document.json
drwxr-xr-x@ 5 160B 1 20 16:59 fonts
drwxr-xr-x@ 34 1.1K 1 20 16:59 images
-rw-rw-r--@ 1 17K 11 30 1979 meta.json
drwxr-xr-x@ 13 416B 1 24 14:52 pages
drwxr-xr-x@ 3 96B 1 20 16:59 previews
-rw-rw-r--@ 1 2.3K 11 30 1979 user.json

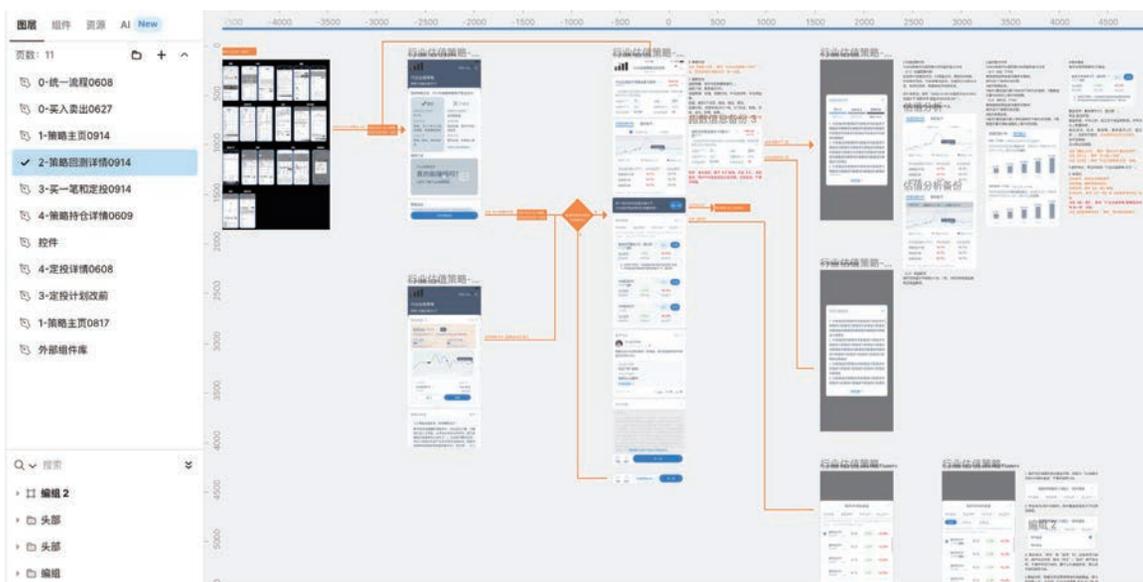
total 99368
-rw-rw-r--@ 1 523K 11 30 1979 094388BA-52BE-4105-BD17-80CF4966105A.json
-rw-rw-r--@ 1 14M 1 24 14:52 1
-rw-rw-r--@ 1 4.6M 11 30 1979 1199AC71-BA77-4DCA-A0B3-34C119627558.json
-rw-rw-r--@ 1 3.5M 11 30 1979 6D03566C-09EF-4CC0-A218-06FF9F20B2C8.json
-rw-rw-r--@ 1 2.4M 11 30 1979 8A126FB9-D4BA-4688-B53E-69CE941FB172.json
-rw-rw-r--@ 1 4.5M 11 30 1979 977E539E-488D-40E6-98B1-2A14D3BE6559.json
-rw-rw-r--@ 1 858K 11 30 1979 A2695E2F-C42C-4F18-976F-FFA4A6007F9A.json
-rw-rw-r--@ 1 4.4M 11 30 1979 B1A4B45B-3088-4C6A-A774-26E8B6731028.json
-rw-rw-r--@ 1 3.3M 11 30 1979 E5F0D65F-13EE-48B3-B4CA-32770703ABBf.json
-rw-rw-r--@ 1 5.2M 11 30 1979 FA2BBAF8-ED76-4584-A9B7-A26EC3D2BC75.json
-rw-rw-r--@ 1 5.7M 11 30 1979 FE824A72-C014-4BDE-8CE2-E2AA1776D7F.json

```

/测试文件/行业估值

/测试文件/行业估值/pages

图6: 交互稿示例



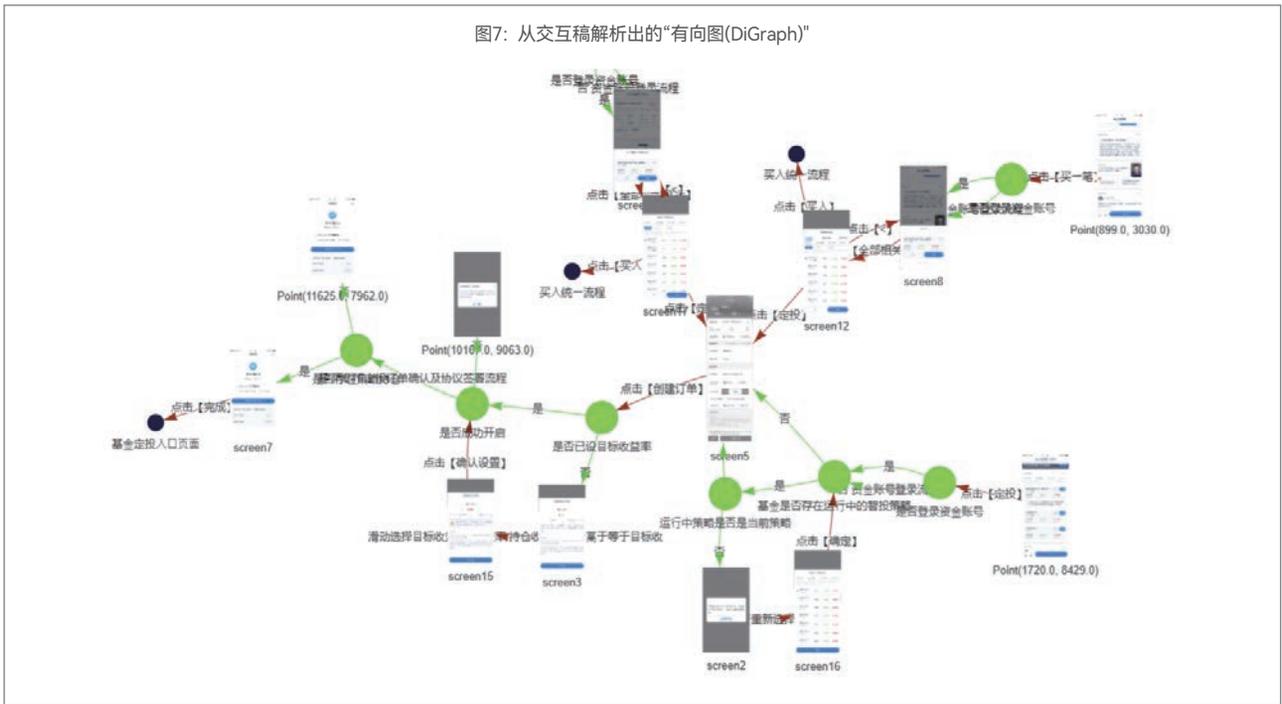
其实际为同一个界面，展示了不同状态的界面场景，且其标题基本一致，因此在解析设计稿时识别到该场景时，程序自动将其合并为同一个节点，同时将界面一、界面二邻近的文字说明合并到同一个节点。

在本模块中，我们以sketch格式的交互稿作为原始输入，解压后的目录结构如图5所示，界面间的交

互逻辑、页面元素在pages文件夹中，一个json文件代表一个交互稿，图6为交互稿示例。

通过解析pages目录下的json文件，可以将交互稿文件信息转化为有向图（DiGraph），具体结构如图7所示，其节点为设计稿中的界面，边为界面跳转所需的操作。

图7: 从交互稿解析出的“有向图(DiGraph)”



在本模块实现了将“有向图”转化为测试用例的功能。“有向图”存储了程序界面间的跳转逻辑，如图6所示，交互稿中，针对每个界面通常都会附有相应的文字说明，我们将该部分文字作为对界面的补充说明拼接到“有向图”中，然后将有向图解析为json格式的测试用例文件。

如图8所示，左侧是交互稿中的一个界面，包括

图8: 由交互稿中的一个界面生成json格式测试用例



界面截图，该界面与其他界面的跳转关系、针对该界面的文字说明，右侧是由该界面生成的测试用例。通过测试专家人工分析，该用例符合交互稿逻辑。

### (2) 控件检测与对比模块

本模块通过调用ocr服务和目标检测服务来处理手机界面截图信息，并通过检测信息来进行界面匹配工作。其中，ocr服务和目标检测服务分别用于图片文字识别和控件识别。

为检测不同图像之间的相似度，我们将界面对比问题转化成了界面控件最优化匹配问题。借助图像识别与控件匹配技术，该平台能够自动完成程序界面与设计稿之间的一致性检测工作。这一功能一方面用于判断程序执行某指令后跳转的界面是否与预期一致，另一方面将匹配成功的界面之间的差异点以如图9所示的测试报告的形式给到测试人员，辅助测试人员高效地检查程序实现与设计稿在操作流程以及界面布局方面是否存在差异。程序通过精确识别程序界面中的各

图9: 测试报告



个控件, 并与设计稿中的对应元素进行匹配, 平台能够迅速定位并指出不一致之处, 从而帮助测试人员及时发现问题, 确保程序最终呈现的效果与设计稿保持一致, 并给开发人员修复程序提供了可靠的参考依据。

### (3) 测试执行模块

测试执行模块是自动执行测试用例的关键模块, 负责处理自动测试相关流程。该模块基于图7所示的“有向图”通过驱动程序操作客户端界面执行操作指令(页面滑动、控件点击、表单输入等), 在执行过程中调用目标检测服务、ocr服务和大模型服务进行界面匹配与用例补齐。大模型服务负责构建相关提示词, 在自动测试环节中通过大模型来尝试生成缺失步骤, 提示词格式为: “在测试手机时我的目标操作是{operation}, 但交互后界面没有变化, 请输出尽可能少的必要步骤帮我完成这个目标操作, 下面是界面数据: {description}”。其中opera-

tion是当前的目标操作, description是对界面的文字描述。description可以是图片本身, 也可以是用文字描述的界面数据, 本文选择后一种方式。

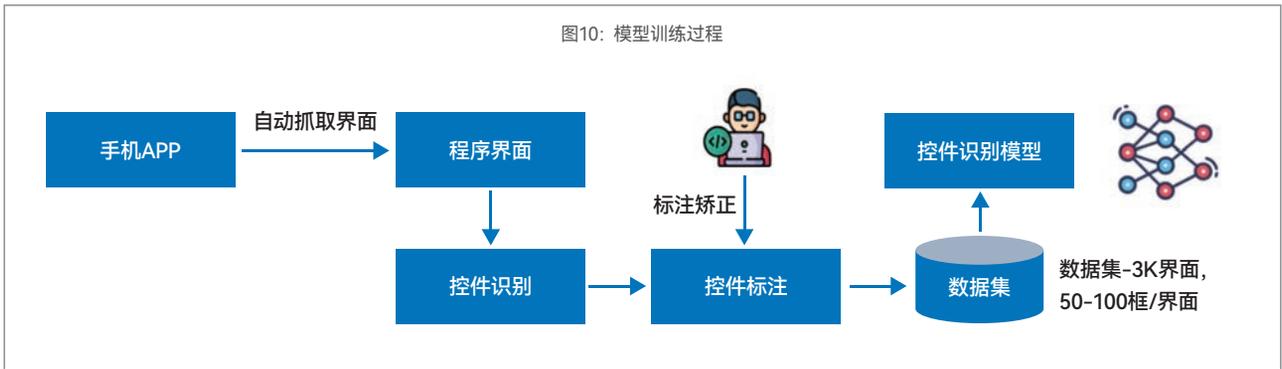
### 2.4.2 目标检测服务

项目采用YOLOv8算法实现目标检测模型, 模型可以自动识别图片上的所有控件类型及位置, 进而有效地将设计稿上的图片转换为程序可操作的序列指令, 控制客户端根据操作指令完成自动测试。

YOLO (You Only Look Once) 是流行的对象检测和图像分割模型, 与传统的滑动窗口和多尺度检测方法相比, YOLO算法将目标检测任务视为一个回归问题, 直接从图像像素到边界框坐标和类别概率的映射, 具有速度快、高精度训练简单的优势。

由于证券客户端的特殊性, 在公开数据集上进行控件识别模型训练效果较差, 模型泛化能力弱, 因此需要定制数据集, 具体流程如图10所示:

图10: 模型训练过程



模型将App上的控件分类如下:

表1: 控件类型

控件类型	图形示例	可操作
文本	卖出数量	否
图片		否
文本按钮		是
图片按钮		是
输入框		是
图表		否
文本图片组合按钮		是
开关按钮		是
其他按钮		是

目标检测框的输出格式为: (中心坐标xy, 目标检测框的宽高wh, 控件类型), 下图11为结果的可视化示例。目标检测模型主要应用于以下两个场景:

a.对当前界面补充测试用例: 程序驱动客户端遵循设计稿处理模块生成的操作指令自动执行测试用例但遇到阻塞时, 调用目标检测服务识别出界面信息, 并以此作为输入调用大模型得出下一步的指令并尝试继续执行。

b.设计稿图片与App界面一致性对比: App测试中, 界面对比是很重要的流程, 测试人员需要循环对比客户端当前界面和设计稿, 本项目将此工作通过程序自动实现。

图11: 控件识别效果示例





## 2.5 研究成果应用

该平台能够迅速且准确地根据设计稿生成与之相匹配的测试用例，在测试执行阶段，自动驱动客户端按设计稿的交互逻辑执行测试用例，并将客户端实际界面与设计稿进行比对，生成界面及交互的一致性测试报告。如图12所示，该图展示了通过程序自动生成的界面测试用例的样例。

在研究成果转化过程中，我们开发了如图13所示的前端界面，为测试人员提供用例生成与测试用例自动化执行入口，图14展示了自动生成的界面一致性测试结果。



在模拟测试环境中，该平台在界面不一致性检测率上可以达到94.6%的查准率和99.6%的查全率。目前该平台正在积极推广试用阶段，已经覆盖了15个不同的需求场景。在这些场景中，每个需求平均包含了13个交互稿页面，每个交互稿页面平均涉及了14个交互流程，总计涵盖了2k+个交互式原子场景。

为了评估该平台的表现，我们邀请了内部测试专家进行了详细的评估。结果显示，该平台在流程一致性方面的准确率高达99.2%，这意味着在理想情况下，平台能够准确地模拟和验证设计稿中的交互流程。同时，在界面控件的识别方面，平台的查准率和查全率也分别达到了95.4%和95.9%，这充分证明了平台在识别界面元素方面的能力。

### 三、总结与展望

#### 3.1 难点突破与创新

##### 1. 设计稿隐式信息推理

在解析交互稿时，我们发现部分页面存在跳转逻辑细节缺失或设计稿信息不全等问题，尽管人能理解并补充这些信息，但程序则会因此出错。为此，我们对大量设计稿深入学习，通过目标检测算法与大模型技术结合的方式实现了交互设计稿细节设计不完整时，智能补全其交互逻辑的功能。最终将设计稿翻译成了一系列在程序界面上可操作的指令，进行软件测试。

##### 2. 界面控件检测与最优对比技术

在客户端程序实际运行过程中，运行设备的屏幕大小、分辨率等因素的差异会导致最终呈现的界面内容与设计稿有差异，因此，我们需要在各种运行设备下有效检测界面控件。为此，我们训练了专有的界面控件识别模型，并基于该模型检测出的控件信息提出了最优控件匹配算法。我们通过对控件建立自定义的偏序关系，从而利用动态规划技术成功实现了控件序列的精准匹配，并定义了控件内容的检测标准。

##### 3.2 实际应用的局限性

本研究验证了在设计稿完整、测试环境完备的理想情况下，客户端测试全流程智能化执行的可行性。对符合既定标准的设计稿，该方法可以显著减少在测试用例梳理、执行及结果比对等环节所需的人力成本，为后续的软件智能化测试研究方向提供了新的视角和路径。然而，本项目也存在一定局限性：

1. 应用效果与设计稿规范程度有关：当前仅在有限的设计稿上进行了测试，自动生成的测试用例质量取决于设计稿的规范程度，虽然公司内部有统一的设计规范，但不同设计师的设计风格与严谨程

度都不同，程序对已知的常见问题进行了自动修复，但对于未知的不规范之处目前还未能覆盖。

2. 实际应用会受测试环境稳定性影响：为了验证程序本身的正确性，我们选取了相对稳定的测试环境，但真实测试工作中，测试环境往往会出现不可预知的问题，这类问题同样会阻塞测试用例自动执行。

3. 终端适配类型有限：对于自动执行测试用例部分，当前仅实现了手机客户端这一类终端适配。

##### 3.3 后续展望

综上所述，在本项研究中，我们创新性地采用了拟人化策略，实现界面测试全流程自动化。具体而言，我们将界面测试用例的生成、自动化执行、结果比对等核心测试流程，均视作人类测试工程师会执行的任务进行模拟，同时将交互式设计稿作为一种新型的“程序断言”工具，用于与程序界面进行对比校验。

未来我们将继续致力于加强工具的工程能力，扩大测试和调试范围，并增加对更多客户端类型的支持，以期进一步提升其实用价值。此外，我们期望后续从构建设计稿规范出发，自动完成设计稿层面的规范提示及修复工作，保证设计稿的规范化、统一化和可执行化，进一步加速本方案的落地应用。期待本技术体系与金融科技行业深度融合，通过消除设计意图与研发实现间的差异，进一步提升金融产品的用户体验一致性，支撑金融系统高频迭代场景下“设计即测试”的极致交付模式，为行业高质量发展注入新动能。🔗

责任编辑：刘译菡

# 基于DeepSeek大模型的网上交易客户端 测试用例研究与实践

陶剑峰、常征、刘浩博 / 中信建投证券股份有限公司

王宏亮、王春 / 上海泽众软件科技有限公司

随着软件项目规模和复杂度的不断提升,传统手动编写测试用例的方法面临着效率低下、覆盖率不足以及一致性差等诸多挑战。近年来,人工智能技术的快速发展为解决这些问题提供了新的思路。本文以中信建投网上交易客户端为例,探讨了基于DeepSeek大模型的测试用例生成方法与实践。通过规范需求文档、结合证券业务知识库以及实现功能测试用例自动生成到GUI自动化用例的高效转换,显著提升了测试效率和覆盖率。实践结果表明,AI与人工协作的模式能够有效平衡自动化与准确性,为证券行业软件测试提供了一种可借鉴的技术方案。

## 引言

在软件开发生命周期中,软件测试是确保系统功能实现和质量保证的关键环节。测试人员需要尽可能全面覆盖系统场景功能、边界条件以及证券业务检查点,同时兼顾正常交易与异常操作(如:信息错误录入、业务各类异常处理等)。长期以来,证券行业的软件测试高度依赖于测试人员专业知识和测试经验。然而,随着软件项目规模和复杂度提升,测试用例维护成本显著上升,传统手动编写测试用例方法面临诸多挑战。其一,手动创建测试用例撰写耗时费力,在大型金融项目中尤为明显;其二,测试人员对证券业务的理解可能存在局限,导致测试用例设计效率低下、覆盖不全。最后,不同测试人员编写的用例可能存在风格和逻辑差异,影响测试执行和后续用例维护。

近年来,人工智能的迅速发展为解决这些问题提供了新思路,我们以中信建投网上交易客户端为例,以DeepSeek 为代表的大模型为探索对象,探讨如何利用人工智能出色的语言解析能力来处理证券

业务需求并自动生成测试用例,期望提高编写效率和测试覆盖率。

## 一、当前AI大模型生成测试用例研究背景

传统测试用例编写存在许多弊病,证券业务需求文档往往包含大量专业术语和复杂逻辑,测试人员在解读需求文档时,需具备一定的证券领域业务经验与理解能力,过程中反复与产品开发沟通协作才能精准理解,容易因理解偏差导致用例设计错误。设计测试用例时,即使测试人员严格遵循等价类、边界值等方法设计,仍难以覆盖所有可能的业务场景,尤其是异常流程。上述这些不可忽视的限制因素,导致整个测试用例设计过程繁杂耗时。此外,在编写测试用例时,还需考虑用例结构清晰,易于理解修改,方便不同的测试人员执行与维护。

针对以上问题,如何发挥大模型自然语言处理能力至关重要。中信建投将从以下方向着手,实现高效需求拆分和全面用例生成,达到降本增效目的。

(一)规范需求文档撰写准则,使大模型精准抓取需求文档信息

规范的需求文档对于大模型精准抓取关键信息尤为重要。一份结构清晰、内容规范的需求文档不仅能够提高大模型对自然语言处理的精准度，还能提升测试用例的生成效率与质量。从文档架构考虑，需求文档应采用明确的标题层级体系（如系统背景、大功能模块的一级菜单，功能点的子级菜单，具体检查点和操作点等次级菜单），这有助于大模型快速定位和理解文档各部分内容之间的逻辑关系。从功能描述考虑，详细准确的描述规范能让大模型有效地捕捉需求的核心要点和细节，语言描述逻辑清晰、无歧义模糊且可读性强，避免用图片截图等不可直接解析的格式，使用简洁直白语言与主动语态，如“用户点击‘提交’按钮”。从业务的关键字考虑，术语和词汇应当做到全文统一，确保全文使用一致的术语，避免同一概念多种表述的情况。

图1：需求文档撰写准则



（二）按模块分割需求，提高大模型识别容错，提高场景用例覆盖率

按功能模块分割需求文档，对于大模型自动生成测试用例具有诸多优势。提高测试用例的准确及覆盖度方面，大模型能够更精准的理解需求，避免因需求理解不全而导致重要测试点的遗漏，结合严谨的测试用例设计方法，可生成贴合实际需求、涵盖正常及异常业务流程的测试用例，从而实现更广泛的场景覆盖，提高测试覆盖率。用例生成效率方面，分割后的文档有利于大模型更高效地读取需求文档，还可将文档分配给不同的测试人员进行并行分析和生成

用例，不仅缩短了文档处理及用例生成的时间，还能快速定位、调整并生成测试用例，减少对整体项目进度的影响。

（三）导入证券领域相关知识、提高用例生成可靠性

在解析证券行业需求文档过程中，导入证券领域相关知识库能够提高用例可靠性。该知识库涵盖证券市场的定义、功能与分类，以及各类证券产品（如股票、债券、基金、期货、期权等）的特点、风险与收益特征。同时包含证券交易基本流程、交易方式和交易规则（如涨跌幅限制、交易时间等）等关键信息。将这些信息融入测试用例生成过程，能够使大模型更精准地理解证券业务，从而生成更贴合实际需求的测试用例。

这一做法可在多方面提高用例可靠性。准确性方面，能够有效减少因业务理解不足导致的错误；全面性方面，有助于大模型深度思考更多业务场景与边界情况，从而提高测试覆盖率；一致性方面，依据知识库标准规范生成测试用例，能够确保用例的统一与规范；可维护方面，方便更新维护知识库，以保证能够保证测试用例的适应性与可维护性。最终能够有效提升证券软件测试的整体质量。

## 二、基于大模型生成测试用例实现策略

目前，PC网上交易客户端自动化测试平台（泽众TestOne自动化测试平台）实现了公司包括PC卓越版，VIP快速交易版，PC同花顺版，以及期权通版等版本的UI自动化测试。基于该测试平台，结合现有证券业务需求，以及目前大模型在自然语言处理方面的相关应用，测试团队逐步实现了网上交易客户端由需求文档到测试用例（涵盖功能测试用例、GUI自动化数据驱动）的自动构建。

（一）技术选择：DeepSeek大模型

在由需求文档转换为功能用例的项目中，平台用到的是DeepSeek大模型，选择该大模型主要基于以下几个方面：

**自然语言处理能力强：**该模式基于Transformer架构构建的大型语言模型。拥有强大的自然语言理解和生成能力，能够应对处理多种复杂的语言任务，像文本生成、语义理解等。

**技术开源：**DeepSeek的部分模型是开源的，可自由利用其技术完善日常测试需求。例如：DeepSeek-R1基于MIT协议开源，这有助于根据需求要素生成端到端黑盒测试用例，覆盖正向、逆向场景，可模拟多步骤业务流程。

**长文本方面逻辑连贯性强：**在长文本方面，该模

型能够维持逻辑连贯性（本次项目使用过程中，在10万字文档中关键信息召回率达到92%，优于GPT-4的85%）。独创的层级注意力机制，有效缓解上传需求文档时的信息衰减问题。

**可控内容生成：**借助向量数据库缓存数据，能够多层次录入提示词进行内容过滤，并且支持自定义用例生成策略，可动态调整生成内容的开放度，从而使生成的测试用例更可靠。

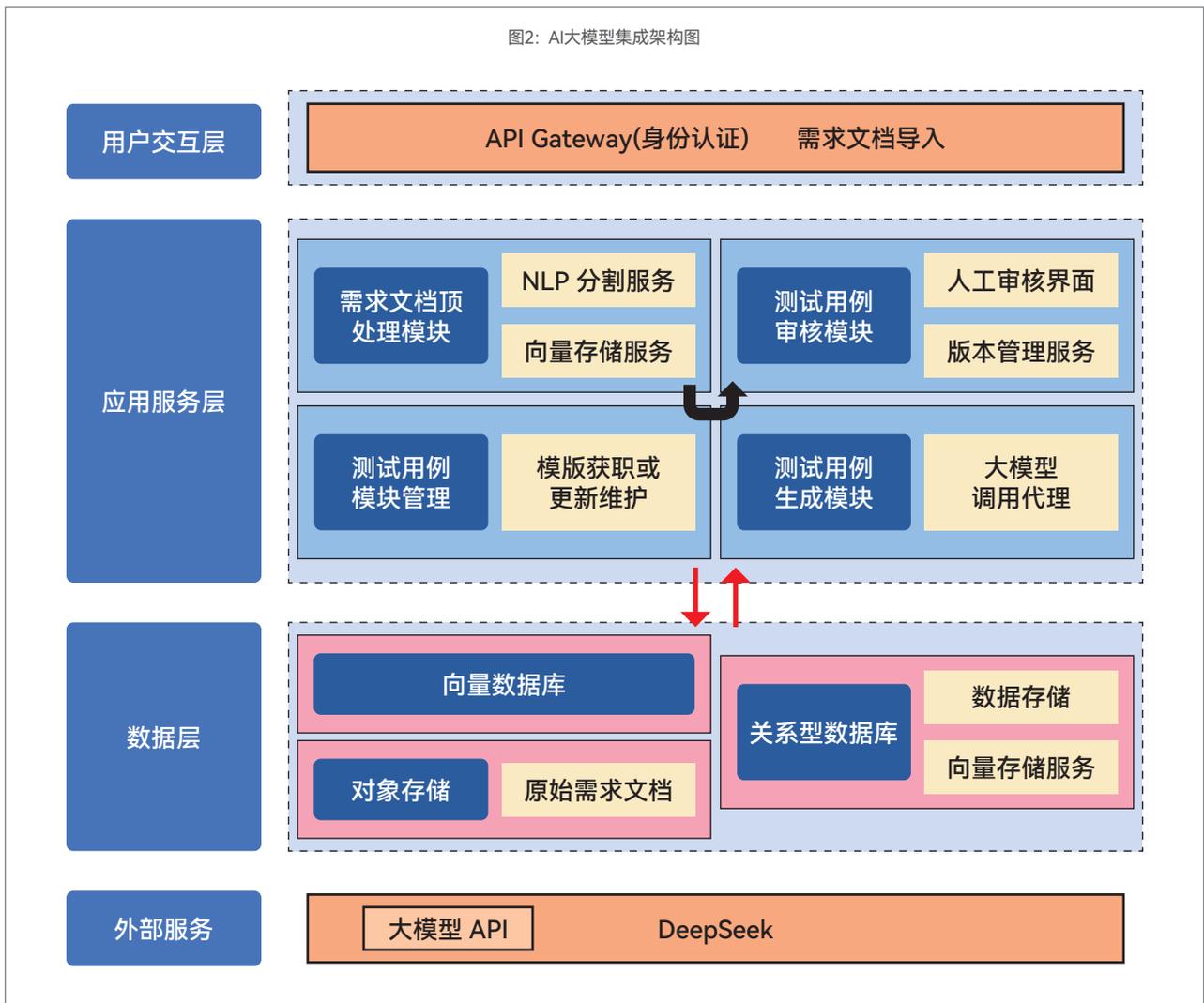
**API数据传输高效：**该模型兼容Hugging Face生态，API调用延迟低于200ms（同类模型平均300-500ms）。

## （二）自动生成用例的核心算法

表1: 用例生成核心算法

分类	算法	目的	应用场景
自然语言处理(NLP)算法	BERT、Transformer	解析《软件需求规格书》中的自然语言描述，提取功能点、边界条件、业务规则等关键测试要素。例如，通过命名实体识别(NER)抽取用户角色、操作对象等实体	文本理解与意图识别，Deepseek的联网搜索功能可结合实时更新的知识库，增强对需求文档中证券行业术语的理解
	依存句法分析、语义角色标注	建立需求文档与测试用例的语义关联，自动生成符合业务逻辑的测试场景。例如，将“用户登录需二次验证”映射为“异常登录场景测试用例”	需求语义映射
生成式AI算法	DeepSeek-R1 模型	根据需求要素生成端到端黑盒测试用例，覆盖正向、逆向场景，模拟多步骤业务流程(如融资买入-下单-委托成交验证-撤单全链路测试)	测试用例生成
优化与验证算法	Q-Learning、R-PageRank	基于历史缺陷数据与代码变更分析，动态调整测试用例执行顺序，优先覆盖高风险模块	测试用例优先级排序
	聚类分析 (K-Means) 相似度计算 (余弦相似度)	合并重复用例，提升测试集效率。例如，通过DeepSeek分析测试日志识别功能重叠的用例	冗余用例检测

图2: AI大模型集成架构图



### (三) 大模型集成系统架构简述

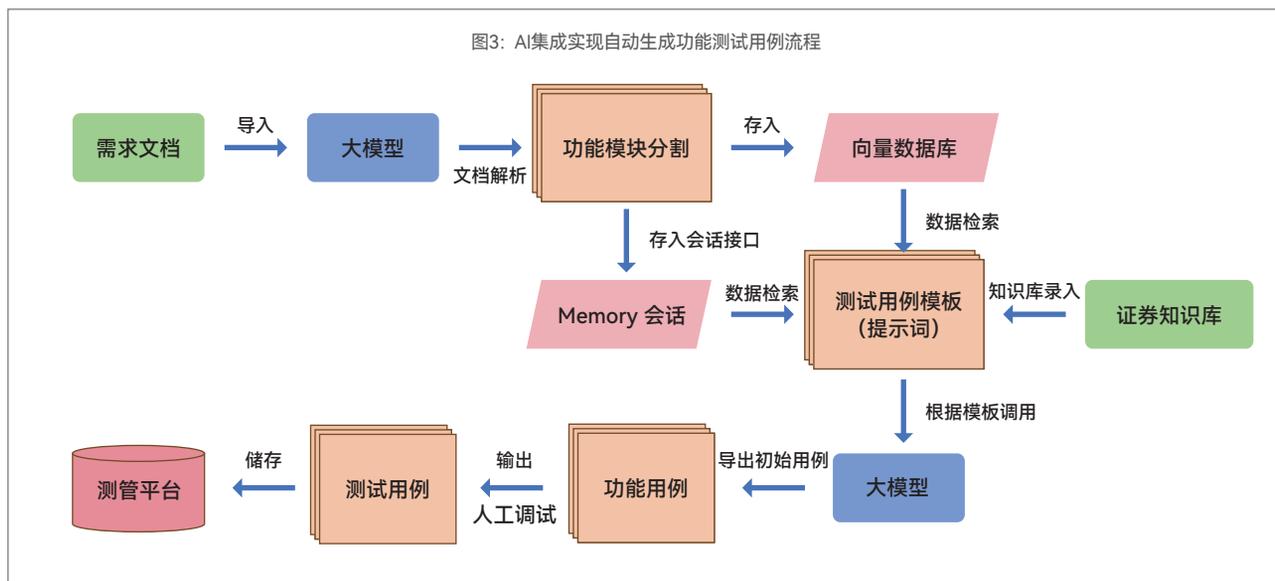
为完成测管平台调用人工智能DeepSeek生成测试用例,共集成以下几个模块,分别为:需求文档预处理模块,使用NLP算法进行语义分割需求文档并存储到数据库;测试用例模板管理,由测管平台维护或获取模板内容;生成测试用例,调用API生成测试用例;测试用例审核模块,通过人工审核生成的测试用例,审核通过的用例保存到测管系统。

### (四) 需求文档转功能测试用例

#### 1、自动生成用例实例描述

现阶段,我们集成的人工智能DeepSeek模型,主要借助OkHttp库来调用DeepSeek的REST API以此为用户提供访问服务。针对大模型的相关服务部署在泽众自研的测管平台上,此平台通过HTTP接口实现对DeepSeek模型的调用操作。当用户在该平台上传文档时,系统便会迅速向DeepSeek接口发送执行请求。随后,DeepSeek基于上述架构及流程依据自身先进的模型算法,对接收到的需求信息进行深度剖析,进而生成海量测试用例。

以下是集成服务实现自动生成测试用例的步骤解析:



步骤一: 将网上交易需求文档导入测管平台, 为后续操作提供基础数据。NLP算法模型(Transformer)会捕捉识别文档中的各种元素, 自动解析需求类型、功能点描述、业务规则、边界条件等关键要素, 将文档从单纯的文本转化为可分析的数据结构。

步骤二: NLP算法模型(BERT)分割需求文档, 按功能逻辑进行语义拆分成独立功能模块, 使得每个功能模块文本聚焦于单一功能, 便于后续针对每个功能进行单独处理。

步骤三: 对功能模块文本向量化存储的同时, 动态缓存分割后的信息存储到Memory会话中。向量数据库可以有效地存储和管理这些功能文本, 方便后续查询和检索, 而Memory会话中存储分割信息, 有助于在整个流程中随时调用相关信息, 保证流程的连贯性。

步骤四: 调用预设的《测试用例模版》为用例提供了规范的框架结构, 通过DeepSeek-R1模型, 结合知识库中的证券交易规则, 依据模版和提示器的要求生成功能用例。

步骤五: 测试人员凭借自身的专业知识和经验,

对生成的测试用例逻辑进行细致的检查, 查看是否存在逻辑错误、满足业务需求等, 调整后用例同步至测管平台。平台通过R-PageRank算法对用例优先级排序, 确保测试资源聚焦核心业务流程, 同时记录反馈数据迭代优化生成模型。

具体实现过程中, 我们采用业务模块精准拆分、动态加载知识库、生成覆盖正逆向场景的基础用例集及规则注入转换边界值案例的四层级联生成逻辑, 如图4所示。整个架构融合了三种设计模式: 需求分层模式确保了需求与规则的清晰解耦; 组合模式通过树形结构支持复杂需求文档的递归遍历; 策略模式则实现了对不同规则类型的灵活处理与无缝扩展, 使得系统能够随着业务规则的演化持续适应而无需修改主流程架构。这种融合基础功能检验与动态业务规则注入的测试生成方法, 极大提升了测试覆盖率和业务合规性验证效率。

## 2、需求文档部分数据获取策略

根据业务需求的文档内容, 针对“中信建投网上交易-股票买卖”功能, 以下是部分数据字典项及其对应的取值范围, 以及相应的正则表达式格式:

图4: 文档解析到用例生成的核心生成逻辑源码

```

/**
 * @param structuredReqDoc 结构化需求文档
 * @return 生成的测试用例集合
 */
public List<TestCase> generateTestCases(RequirementDocument structuredReqDoc) {
    List<TestCase> testCases = new ArrayList<>();

    // 1. 模块化解析需求文档
    structuredReqDoc.mainModules.forEach(module -> {
        // 2. 动态加载知识库规则
        List<KnowledgeRule> matchedRules = queryRulesFromKnowledgeBase(
            module.moduleName,
            KnowledgeRuleType.TRADING_RULE
        );
        // 3. 生成基础功能用例
        module.subFunctions.forEach(subFunc -> {
            testCases.addAll(generateBasicCases(subFunc));

            // 4. 注入业务规则约束
            matchedRules.forEach(rule -> {
                testCases.addAll(applyBusinessRules(subFunc, rule));
            });
        });
    });
    return testCases;
}

```

### 3、知识库的构建与应用

知识库是大模型生成高质量测试用例的核心支撑,通过术语对齐与规则约束实现输入规范化以减少

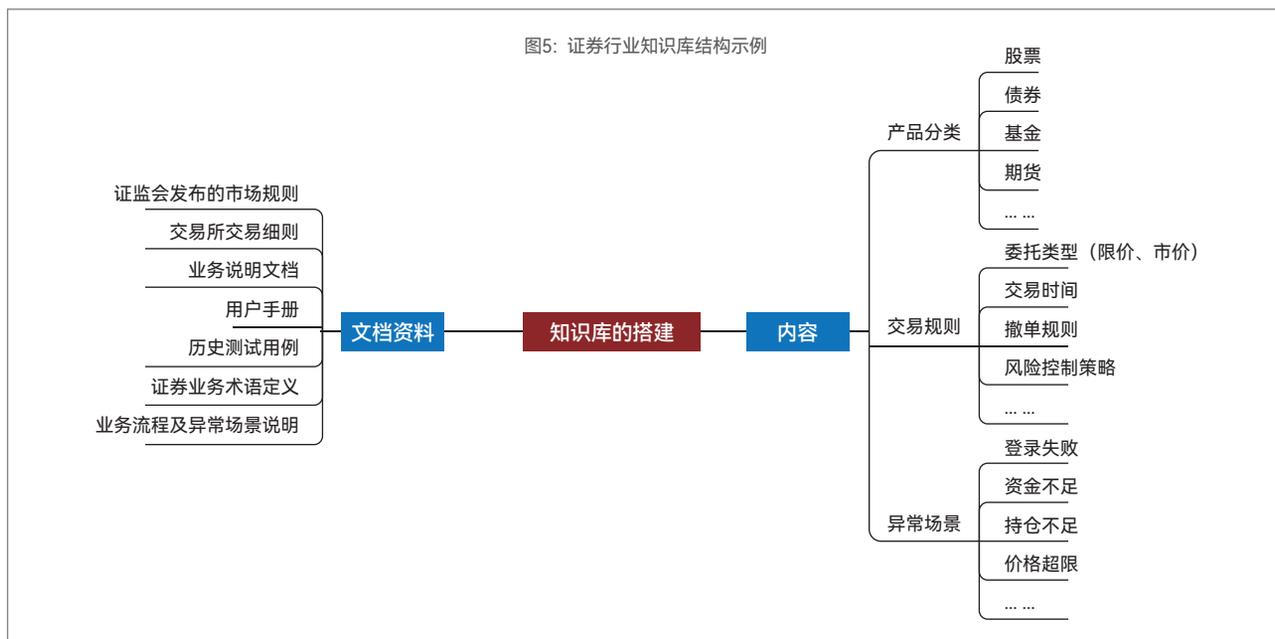
模型理解偏差,凭借库内丰富业务知识进行逻辑增强来生成复杂场景用例,依靠动态更新实时纠错确保用例契合最新业务规则;同时,借助结构化存储、动态集成与持续维护,显著提升AI生成测试用例的可靠性与实用性,保障证券业务系统稳定运行。

#### (1) 证券业务知识库构建

针对本次实践中,我们导入的知识库主要包括以下内容:文档包含证监会发布的市场规则、交易所交易细则等行业标准文档,以及证券行业术语定义,内部的历史测试用例(主业务流程和异常场景),业务说明文档等。内容涵盖产品分类,如股票、债券、基金、期货等产品的属性(包括代码规则、交易单位、涨跌幅限制);交易规则,如委托类型(限价、市价)、交易时间、撤单规则、风险控制策略;以及异常场景,如登录失败、资金不足、价格超限等异常处理逻辑。

表2: 数据字典与正则表达

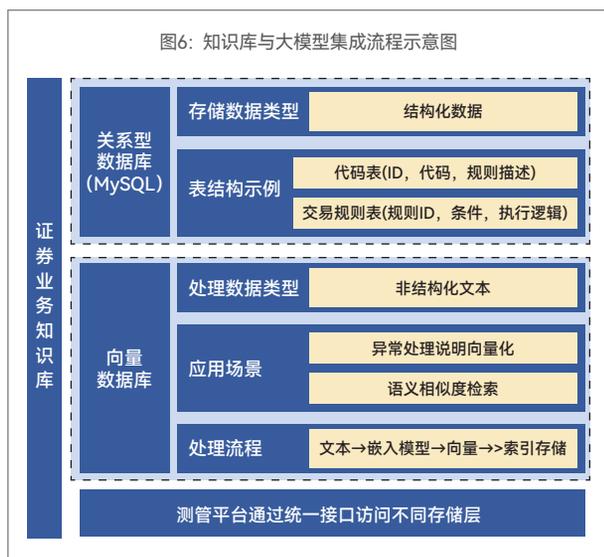
序号	数据字典项	取值范围及正则表达式格式
1	股票名称及代码	-取值范围: 证券名称(证券代码)。 -正则表达式: `^[([\u4e00-\u9fa5]+)\s*\((\d{6})\)`
2	买卖方向	-取值范围: “买入”或者“卖出”。 -正则表达式: `^买入\$ ^卖出\$`
3	委托方式	-取值范围: “限价委托”、“本方最优价格”.....“全额成交或撤销”。 -正则表达式: `^限价委托\$ ^本方最优价格\$ ^... ^全额成交或撤销\$`
4	交易数量	-取值范围: 正整数, 根据用户资金和股票交易规则而定。 -正则表达式: `^\d+\$`
5	交易价格	-取值范围: 受市场行情波动影响, 具体价格在页面五档行情获取。 -正则表达式: `^(行情价超过涨停价 低于跌停价 超过涨幅 超过跌幅)`
6	投资者账户信息(账号)	-取值范围: 数字组合, 不同券商有不同的账号格式。 -正则表达式: `^\d{8,12}\$`
7	投资者账户信息(账户类型)	-取值范围: 可能是普通账户、信用账户等。 -正则表达式: `^普通账户\$ ^信用账户\$`
8	系统提示信息处理	-取值范围: 关键字包含窗体提示: “.....”字样。 -正则表达式: `提示: “([\^]+)”`



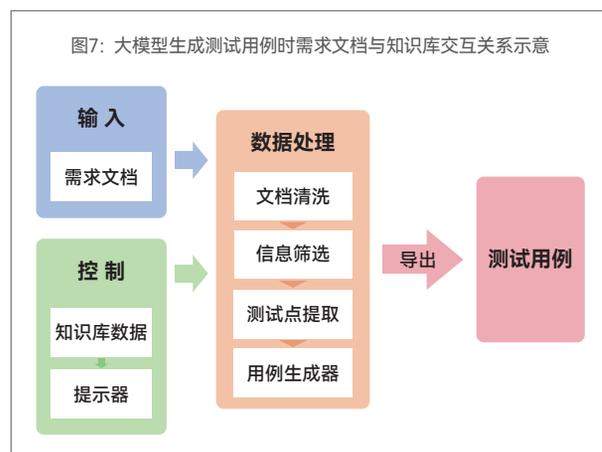
## (2) 知识库的存储与集成

通过API将知识库接入测试管理平台,在需求解析阶段,调用库内数据索引匹配业务术语、保证大模型对需求的准确理解;在用例生成阶段,动态注入业务规则约束,比如“创业板股票需满足单笔最低100股”这类规则,使生成的测试用例更贴合实际业务要求,保障测试的准确性和有效性。

如图6所示,本次实践对知识库做多模态数据存储,构建了面向证券测试场景的智能知识中枢,通过统一接口,访问不同存储层。在上传需求文档解析的时候,触发访问关系型数据库获取结构化数据,需求语义检索向量数据匹配。该架构在测管平台的应用数据显示,业务术语识别准确率提升至92%,测试用例生成耗时降低65%,有效解决了证券业务数字化测试中的知识碎片化难题。



## (3) 测试用例提示词示例



示例1:

需求描述:“用户盘后定价买入科创板股票”。

知识库关联: [规则ID421]盘后定价交易时间为每个交易日的15:05至15:30,申报时间为9:15至11:30和13:00至15:30。[规则ID734]科创板股票代码以“688”开头,需满足“单笔申报数量需大于等于 200 股,小于等于 100 万股”且为“200股整数倍”交易规则。

生成用例:自动添加“输入非688代码时提示错误”;“输入100股时触发校验失败”等场景;“输入1000200股时触发校验失败”等场景;“非申报时间交易触发校验”等场景。

示例2:

需求描述:“用户融资买入深交所股票”。

知识库关联: [规则ID457]融资买入需校验保证金比例

生成用例:自动添加“保证金充足时允许下单”场景;“保证金不足时拦截交易”场景。

图8: 知识库规则源码

```
// 知识库规则实体
private static class KnowledgeRule {
    String ruleId;
    String ruleType;           // 规则类型 (交易规则/术语定义)
    String content;           // 规则内容
    LocalDateTime effectiveDate; // 生效时间
}
```

图9: 规则匹配策略源码

```
List<String> extractEntities(String text) {
    return NLPProcessor.analyze(text).getEntities()
        .stream()
        .filter(e -> e.getType().equals("FINANCIAL_TERM"))
        .map(Entity::getValue)
        .collect(Collectors.toList());
}
```

(4) 大模型实现功能用例输出

以下是根据《中信建投网上交易-卓越版》、《测试用例模板》文档及结合,证券业务知识库生成的两个中

信建投股票买卖功能测试用例,这两个测试用例分别覆盖了中信建投股票成功买入成功和失败的场景:

图10: 上传需求文档



图11: 测试用例模板源码

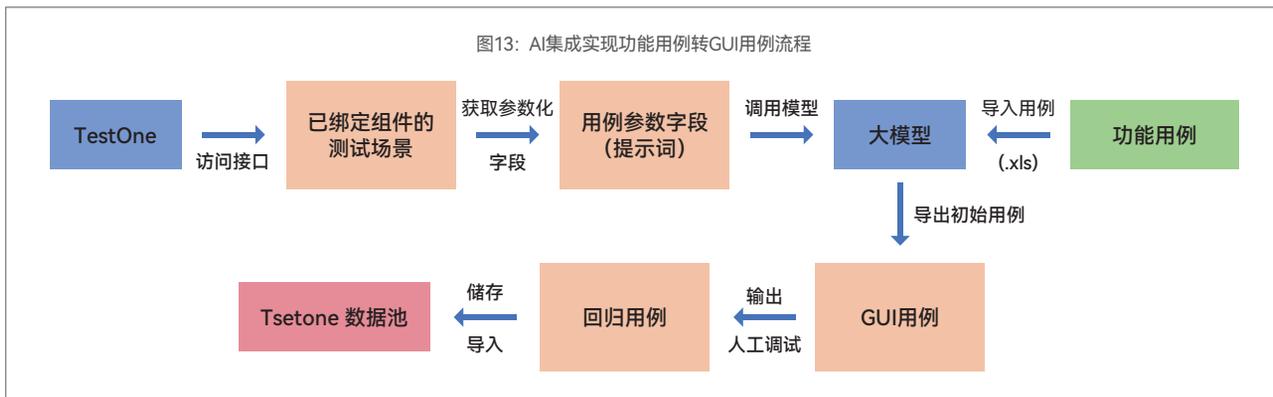
```
<?xml version="1.0" encoding="UTF-8"?>
<template>
  <template_name>stock_trading</template_name>
  <structure>
    <case_id>自动生成格式: {模块缩写}-{三位序号}</case_id>
    <case_name>{交易类型}-{市场}-{操作}: {股票名称}({股票代码})</case_name>
    <prerequisites>列出执行此用例前必须满足的条件</prerequisites>
    <test_steps>
      <step>步骤应按逻辑顺序编号</step>
      <step>每个步骤应清晰明确</step>
      <step>包含所有必要的操作细节</step>
    </test_steps>
    <expected_result>
      <result>列出每个重要步骤的预期结果</result>
      <result>包括系统响应和数据变化</result>
    </expected_result>
  </structure>
</template>
```

图12: 生成功能用例展示

<p>测试用例1:</p> <p>用例编号: ZJW - 001</p> <p>用例名称: 集中交易-普通买卖-深市成功买入: 平安银行 (000001)</p> <p>前提条件: 正常普通账号登录, 账号证券持仓充足。</p> <p>测试步骤:</p> <ol style="list-style-type: none"> <li>1. 进入普通买卖页面。</li> <li>2. 选择“买入”方向。</li> <li>3. 输入证券代码: 000001。</li> <li>4. 输入行情价。</li> <li>5. 输入交易数量 100。</li> <li>6. 确认交易, 检查当日交易流水</li> </ol> <p>预期结果:</p> <ol style="list-style-type: none"> <li>1. 委托交易确认, 系统提示字段: 操作类别: 买入; 股票代码: 000001; 证券名称: 平安银行; 委托价格: (行情价); 委托数量: 100; 委托方式: 限价委托; 股票代码: xxxxxxxx。</li> <li>2. 系统提示: 委托已提交, 委托编号是 X。</li> <li>3. 当日委托该笔交易状态: 已成; 其余信息显示正常。</li> <li>4. 当日成交该笔交易状态: 普通成交; 其余信息显示正常。</li> </ol>	<p>测试用例2:</p> <p>用例编号: ZJW - 034</p> <p>用例名称: 集中交易-普通买卖-深市超过跌停价买入: 平安银行 (000001)</p> <p>前提条件: 正常普通账号登录, 账号证券持仓充足。</p> <p>测试步骤:</p> <ol style="list-style-type: none"> <li>1. 进入普通买卖页面。</li> <li>2. 选择“买入”方向。</li> <li>3. 输入证券代码: 000001。</li> <li>4. 输入超过跌停价。</li> <li>5. 输入交易数量 100。</li> <li>6. 确认交易</li> </ol> <p>预期结果:</p> <ol style="list-style-type: none"> <li>1. 交易确认, 系统提示字段: 操作类别: 买入; 股票代码: 000001; 证券名称: 平安银行; 委托价格: (价格); 委托数量: 100; 委托方式: 限价委托; 股票代码: xxxxxxxx。</li> <li>2. 系统提示: 委托已提交, 证券系统返回的原因: -xxxxxxxxx [-xxxxxxxxx]委托价格超过跌停价格。</li> </ol>
--	---

在实践中,基于DeepSeek大模型生成测试用例时,确实遇到了一些挑战。一方面,需求文档部分内容存在模糊描述、逻辑结构混乱或与知识库数据发生冲突时,大模型难以准确提取关键测试点,导致生成的用例覆盖不全或被丢弃。另一方面,提示器在设计过程中部分需求或知识库中若未明确覆盖边界条件、异常场景或特定业务规则, AI可能也会忽略部分用例。此外,

图13: AI集成实现功能用例转GUI用例流程



就是大模型解析的需求文本过长时可能出现解析内容衰减,或对复杂业务逻辑(如多步骤交易链路)的连贯性支持不足,导致部分用例被截断。

面对这些问题,我们也在积极调整执行策略,如针对流程图、业务规则等非文本内容,我们强制添加详尽的文字说明,防止因图像无法解析而造成用例遗漏。同时,对知识库内容进行优化完善,并精心优化提示词的多层级指令设计,以此逐步缩小并细化生成范围。这些举措不仅大幅提高了用例生成的效率,关键业务场景中异常流程的覆盖率也得到了显著提升。

### (五)功能用例转GUI自动化测试用例

#### 1、自动转换GUI用例描述

中信建投网上交易GUI自动化测试用例参数,同样借助了DeepSeek模型与泽众TestOne自动化测试平

台实现。具体过程如下:先将现有的功能案例库导入TestOne平台,同时读取平台预先设定好的GUI场景参数字段生产关键提示词,利用平台集成的DeepSeek模型NLP引擎(基于Transformer架构)解析处理这些关键信息数据,依据其强大语言处理能力生成表单数据(这些数据将作为GUI测试的数据驱动)。再由测试人员审查调整模型输出的表单数据后,将合格适用的参数上传到TestOne平台内指定场景组件,即可通过参数化驱动机制批量生成可执行回归用例,实现全面自动化测试。

#### 2、GUI用例实例输出

以下是根据将“创业板股票”部分测试用例导入TestOne平台后,平台凭借DeepSeek模型生成的部分GUI用例:

图14: 功能用例转GUI用例实例

用例名称	证券代码	委托价格	委托数量	委托属性	预期结果	实际结果	校验结果	校验规则
TC001-闪电买入-创业板股票-限价买入-有20权限	300001	行市价	100	限价委托	委托成功,提示“委托已提交,合同号是X”	-	-	会计金额=委托价格*委托数量
TC002-闪电买入-创业板股票-限价买入-委托数量30万股	300002	行市价	300000	限价委托	委托成功,提示“委托已提交,合同号是X”	-	-	委托成功后,可在当日委托中查询到委托记录
TC003-闪电买入-创业板股票-限价买入-委托数量300100股	300003	行市价	300100	限价委托	提示委托数量高于最高交易数量	-	-	提示委托数量高于最高交易数量
TC004-闪电卖出-创业板股票-限价卖出-有20权限	300065	行市价	100	限价委托	委托成功,提示“委托已提交,合同号是X”	-	-	会计金额=委托价格*委托数量
TC005-闪电卖出-创业板股票-限价卖出-委托数量30万股	300166	行市价	300000	限价委托	委托成功,提示“委托已提交,合同号是X”	-	-	委托成功后,可在当日委托中查询到委托记录
TC006-闪电卖出-创业板股票-限价卖出-委托数量300100股	300301	行市价	300100	限价委托	提示委托数量高于最高交易数量	-	-	提示委托数量高于最高交易数量
TC007-普通买入-上市前5日创业板股票-限价买入	300360	行市价	100	限价委托	分时图、涨跌幅价格显示正确	-	-	涨停价、跌停价显示“-1”
TC008-普通买入-上市5个交易日后创业板股票-限价买入	300366	行市价	100	限价委托	分时图、涨跌幅价格显示正确	-	-	5个交易日后涨跌幅为20%
TC009-普通买入-创业板股票-限价买入-有20权限	300623	行市价	100	限价委托	委托成功,提示“委托已提交,合同号是X”	-	-	会计金额=委托价格*委托数量
TC010-普通买入-创业板股票-对方最优-有20权限	300624	市价	100	对方最优	委托成功,提示“委托已提交,合同号是X”	-	-	委托价格展示为“市价委托”
TC011-普通买入-创业板股票-本方最优-有20权限	300757	市价	100	本方最优	委托成功,提示“委托已提交,合同号是X”	-	-	委托价格展示为“市价委托”
TC012-普通买入-创业板股票-即时成交剩余撤销-有20权限	300860	市价	100	即时成交剩余撤销	委托成功,提示“委托已提交,合同号是X”	-	-	委托价格展示为“市价委托”
TC013-普通买入-创业板股票-五档即时成交剩余撤销-有20权限	300861	市价	100	五档即时成交剩余撤销	委托成功,提示“委托已提交,合同号是X”	-	-	委托价格展示为“市价委托”
TC014-普通买入-创业板股票-盘后固定交易	300001	行市价	100	盘后固定交易	委托成功,提示“委托已提交,合同号是X”	-	-	委托方式为“盘后固定交易”
TC015-普通买入-创业板股票-限价买入-委托数量30万股	300002	行市价	300000	限价委托	委托成功,提示“委托已提交,合同号是X”	-	-	委托成功后,可在当日委托中查询到委托记录
TC016-普通买入-创业板股票-限价买入-委托数量300100	300003	行市价	300100	限价委托	提示委托数量高于最高交易数量	-	-	提示委托数量高于最高交易数量
TC017-普通买入-创业板股票-对方最优-委托数量15万股	300065	市价	150000	对方最优	委托成功,提示“委托已提交,合同号是X”	-	-	委托成功后,可在当日委托中查询到委托记录
TC018-普通买入-创业板股票-对方最优-委托数量150100股	300166	市价	150100	对方最优	提示委托数量高于最高交易数量	-	-	提示委托数量高于最高交易数量
TC019-普通卖出-限价卖出-上市前5日创业板股票	300301	行市价	100	限价委托	分时图、涨跌幅价格显示正确	-	-	涨停价、跌停价显示“-1”
TC020-普通卖出-限价卖出-上市5个交易日后创业板股票	300360	行市价	100	限价委托	分时图、涨跌幅价格显示正确	-	-	5个交易日后涨跌幅为20%
TC021-普通卖出-创业板股票-限价卖出-有20权限	300366	行市价	100	限价委托	委托成功,提示“委托已提交,合同号是X”	-	-	会计金额=委托价格*委托数量
TC022-普通卖出-创业板股票-对方最优-有20权限	300623	市价	100	对方最优	委托成功,提示“委托已提交,合同号是X”	-	-	委托价格展示为“市价委托”
TC023-普通卖出-创业板股票-本方最优-有20权限	300624	市价	100	本方最优	委托成功,提示“委托已提交,合同号是X”	-	-	委托价格展示为“市价委托”
TC024-普通卖出-创业板股票-五档即时成交剩余撤销-有20权限	300660	市价	100	五档即时成交剩余撤销	委托成功,提示“委托已提交,合同号是X”	-	-	委托价格展示为“市价委托”
TC025-普通卖出-创业板股票-盘后固定交易	300661	行市价	100	盘后固定交易	委托成功,提示“委托已提交,合同号是X”	-	-	委托方式为“盘后固定交易”

### 三、讨论与总结

上面我们展示的DeepSeek大模型应用于中信建投网上交易测试项目，人工智能解析需求文档，自动生成的功能用例，功能用例转换GUI用例的过程。在前期工作中，我们通过规范需求测试文档、审查和修改文档，并严格遵循文档规范，逐步探索出利用人工智能自动生成用例的新途径。结果证明，这种方法非常有效。人工智能在解析详细且规范的需求文档方面表现出色，能够迅速完成功能的结构划分，并协助进行检查点解析及用例设计。

总之，借助DeepSeek大模型进行测试用例编写，通过规范需求文档，合理设计提示语链、结合自动化工具和人工审查，可以确保输出用例结果的准确性和实用性。生成用例只是测试提效的一部分，未来，随着AI技术的不断发展进步，这种人机协作模式将在软件测试领域发挥更大的作用，测试人员能将更多的精力投入到更高层次的测试策略和质量保证活动中。

责任编辑：陈潇

#### 参考文献

- [1]江伟维.基于大模型的逻辑编排应用与测试用例生成技术研究[D].浙江大学,2024.DOI:10.27461/d.cnki.gzjdx.2024.000499.
- [2]程志波.人工智能驱动的自动化测试工具研究[J].信息与电脑,2025,37(02):28-30.
- [3]新华网. AI大模型竞争或激发创新浪潮[J]. 信息系统工程,2025(2):封3. DOI:10.3969/j.issn.1001-2362.2025.02.047.
- [4]重庆农村商业银行股份有限公司. 基于人工智能的测试用例生成方法、装置、设备及介质:CN202410621738.3[P]. 2024-08-09.
- [5]中科数测(无锡)科技有限公司. 一种基于Transformer翻译模型的测试用例步骤生成方法、设备及介质:CN202410537664.5[P]. 2024-08-02.
- [6]上海外服云信息技术有限公司. 基于大语言模型的需求文档解析与测试用例自动生成方法及系统:CN202411651263.9[P]. 2025-02-25.
- [7]西安中朗智控科技有限公司. 基于大语言模型的软件配置项测试用例自动生成方法:CN202411884199.9[P]. 2025-01-21.



# 金融大模型安全治理体系研究：挑战、测评与防御技术

陈洪炎 / 上交所技术有限责任公司

大模型技术在金融领域显著提高工作效率，改变金融业务生产模式，并产生巨大经济价值和潜力，大模型在金融领域被广泛应用在市场营销、投资决策、信用评估、智能风控、反欺诈和监管等方向，金融大模型的技术优越性、行业契合性与实践可行性日益凸显。伴随金融大模型的深度应用，产学研用各方也遇到了大模型安全威胁，包括数据泄露、模型窃取、预测操控和系统性风险等。提升大模型的鲁棒性、可解释性、公平性、真实性、价值对齐、隐私保护等方向的能力逐渐成为金融大模型的行业热点。本文旨在探讨金融领域大模型的安全挑战并探索解决方案，分析金融大模型面临的安全威胁，探讨其风险治理模型与技术，有助于金融大模型健康有序地发展。

## 一、引言

金融大模型拥有将金融知识广泛迁移的强大能力，可高效处理复杂信息，生成内容丰富多样，并且在人机交互方面表现出色，为金融领域带来了巨大的发展潜力与价值创造机遇。不过，这一技术也面临着诸多挑战，如生成内容的可信度有待商榷，推理能力不够稳定，存在外部安全隐患，隐私泄露和数据安全问题突出，还可能遭遇模型窃取以及法律监管风险等。由于金融大模型依赖概率推理输出结果，这可能导致其对模糊预测过于自信，从而编造出错误或虚假信息。尽管其“智能涌现”表现不俗，但其突发性、不可预测性和不可控性也潜藏着风险。此外，金融大模型的应用在一定程度上降低了查找系统漏洞和发动攻击的难度，若系统被恶意植入后门，其安全性将受到极大威胁。金融大模型所涉及的数据多关乎投资者的资金安全、个人隐私、情感伦理、版权侵权等敏感内容。金融业务的专业性、严谨性及合规性要求，使得金融大模型的落地应用必须满足严格的风险治理标准。

## 二、金融大模型的安全挑战

金融大模型的数据量、算力规模以及模型参数

持续增长，在金融业务中引入大模型虽能激发诸多创新活力，却也同步催生了金融机构与个人投资者面临安全隐私泄露等风险隐患，该模型在数据隐私保护、模型推理安全性和内容合规性等方面遭遇诸多安全挑战。金融大模型从落地应用来看，通常需历经数据采集与处理、模型训练、模型部署以及最终的应用实施这四大关键环节，而其安全风险则贯穿于整个研发与应用的生命周期之中(表1)。

## 三、金融大模型安全测评

为了使金融大模型在实际应用中充分展现其最佳效果，同时有效规避潜在风险和滥用情形，开展安全测评显得尤为关键。借助安全测评，能够迅速察觉潜在的安全性、可靠性和可控性问题，进而助力研发人员增强系统的稳健性和安全性。金融大模型的评测主要聚焦于通用能力和安全性两大领域。通用能力通常涵盖模型的理解力、推理力、计算力以及处理力。安全性则涉及内容安全、数据安全以及价值观等方面。安全测评的核心任务主要包含以下三个方面：一是构建安全评测数据集；二是设计安全评测内容；三是部署安全评测环境。

表1: 金融大模型常见安全风险举例

安全风险	视角	治理项
隐私安全	训练数据的来源若不透明, 可能会带来恶意攻击和数据侵权的隐患。	数据收集处理、模型训练、模型落地应用
数据泄露	数据泄露是指用户在与大型模型交互时, 其敏感信息(如财务资料、交易记录和个人识别信息等)可能在发送请求或接收响应时被非法获取。	数据收集处理、模型训练、模型落地应用
后门攻击	攻击者可能会在大型模型的训练数据中植入特定的输入输出配对, 目的是为了在未来某个时刻能够利用这些条件来操纵金融大型模型。	模型训练
数据投毒攻击	攻击者可能在模型的训练数据集中植入少量含有恶意内容的有害样本, 这些样本会在模型训练或微调过程中对模型造成“污染”, 从而损害模型的有效性。	模型训练
数据窃取	数据窃取攻击是一种技术策略, 攻击者通过巧妙构造的输入传递给大模型, 目的是从模型的反馈中抽取并拼凑出在模型训练阶段所涉及的敏感信息, 试图诱使模型无意中透露其训练数据中的保密信息。	模型训练
对抗攻击	对抗样本是攻击者精心制作与正常样本几乎没有区别的样本, 但模型会在其上面分类错误, 从而产生错误输出。	模型训练
指令攻击	攻击者可以通过设计特定的指令, 让大模型产生不安全的输出。	模型训练
模型窃取	对于高价值闭源模型的窃取。	模型训练
RAG投毒	RAG数据对于金融大模型提供准确的信息至关重要, 金融机构为了节约成本并增强检索经常需要外挂知识库, 但金融业务涉及的数据敏感性强。RAG在实际应用中若访问权限控制不佳会导致数据泄露, 若使用的向量数据库被投毒, 则会输出有害内容, 在训练中使用嵌入技术可能会泄露敏感数据。	模型训练
硬件安全漏洞	硬件安全包括CPU计算安全、内存安全、智能体安全等。攻击者可利用硬件漏洞提前训练参数, 导致模型性能下降, 无法拟合。	模型训练、模型部署
软件安全漏洞	软件供应链收到攻击、编程语言漏洞等。	模型训练、模型部署
框架安全漏洞	金融大模型底层的深度学习框架漏洞	模型训练、模型部署
业务使用参数	通过构造提示词信息, 利用金融大模型的预测机制, 将模型推向异常的概率空间, 改变模型行为以劫持模型输出, 使模型生成预期外或有害信息。	模型落地应用
业务使用参数	通过虚假访问信息、混淆角色、对齐黑客攻击和假设权威以绕过限制使模型输出预期之外的信息。	模型落地应用
业务使用参数	由于大模型在训练数据的采集和使用过程中可能缺乏明确的法律规范, 存在被恶意利用的风险, 这可能导致数据滥用和侵犯知识产权等问题。	模型落地应用
业务使用参数	由于训练数据的不足和训练过程的不完善, 大型模型在生成内容时可能与可验证的现实世界事实存在偏差, 这可能导致其输出包含恐怖主义、极端主义、色情、暴力等有害信息。	模型落地应用
业务使用参数	在处理 and 生成自然语言的过程中, 大型模型依赖于训练数据中的模式和结构, 有时可能会生成看似合理但实际上并不准确的内容。	模型落地应用
业务使用参数	目前, 大型模型在全生命周期管理方面的体制尚未完善, 这可能导致偏见和歧视、隐私泄露、错误信息传播以及模型决策的不可解释性等问题的出现。	模型落地应用

### 3.1 构建安全评测数据集

选用标准化的数据集来进行评测, 这些数据集涵盖了模型基础能力、金融安全与价值对齐能力、金融风险控制能力、金融专业认知能力以及金融业务辅助拓展能力等方面。具体来看, 模型基础能力数据集囊括了数学、推理、知识、语言、代码、指令跟随以及智能体等相关内容。

金融安全与价值对齐能力数据集则分为基础性金融安全和对抗性金融安全两大类。基础性金融安全进一步细分为可解释性、偏见、歧视、输出不可靠、混淆事实、误导用户、绕过鉴权、不当使用引发信息泄露、滥用与网络攻击、挑战传统社会秩序、加剧“信息茧房”效应等多个方面; 对抗性金融安全则包括信息诱导、多轮问题诱导以及假设情景下的诱导等情形。需要注意的是, 金融安全与价值对齐能力数据集与实际金融业务可能存在潜在差异, 因此必须确保数据的准确性和安全性, 同时保障风险防控的及时性和有效性。

金融风险控制能力数据集主要包含合规风险和其他风险, 其中其他风险又细分为市场风险、操作风险、流动性风险和信用风险等。

金融专业认知能力数据集则由金融基础知识和金融信息构成。金融基础知识进一步细分为会计学、金融学、经济学以及资质认证等内容; 金融信息则包括财报、研报、K线图、金融知识图谱等。

金融业务辅助拓展能力数据集分为对内和对外两个部分。对内方面涵盖了智能投研和智能投教, 而对外则是智能投顾。

### 3.2 设计安全评测内容

金融大模型的评测主要聚焦于基础能力和金融能力两大板块。基础能力方面, 它包含了语言理解、指令执行、逻辑推理、数学计算以及内容生成等诸多维

度。而金融能力则侧重于考察模型在处理复杂金融数据和场景时, 基于完整性、针对性及区分度的综合能力表现。

在基础能力的评测中, 主要细分为知识能力、推理能力、理解能力、语言能力和安全能力。知识能力着重于评估模型对广泛知识的掌握程度, 以及对不同层次、不同学科知识的理解深度, 通过构建涵盖各级学科的分类体系, 形成一套完整的学科能力评测方案。推理能力则是衡量模型在逻辑分析和问题解决方面的实力, 例如数学计算、逻辑判断、复杂问题的因果推导、程序编写以及问题定位与解决等能力。理解能力用于评估模型对给定文本的深入解读能力, 包括对文字含义的精准把握、主旨的提炼、语境的解析以及创意写作水平。语言能力则关注模型在语言理解和生成方面的表现, 涵盖对字词的精准识别、分类、释义以及新词创造, 对语句、段落语法结构的分析与纠错, 以及跨语言翻译的能力。

金融能力的评测则主要围绕金融通识的掌握程度与应用的广度和深度、金融业务的合规性、事实的准确性、推理的正确性以及事件的实时性等方面展开。评测内容应全面覆盖从基础金融知识到高级金融理论, 从基础金融应用到特定金融领域应用的各个层面, 以确保全方位评估金融大模型的金融能力。此外, 还应设计具有特殊性的金融任务, 以此来真实地反映金融大模型在处理金融特殊性任务时的能力水平。

### 3.3 部署安全评测平台

安全评测平台应具备以下四项关键能力:

一是即时响应能力。该平台需配备即时反应机制, 能够迅速对潜在风险做出处理, 一方面防止用户接触到有害内容, 另一方面保障金融大型模型免受攻击, 确保在风险出现的第一时间就能有效应对。

二是适应性。由于安全需求处于不断变化之中，安全评测平台必须具备高度的灵活性，能够实时对内容安全控制规则进行更新调整，以适应新的安全形势和要求，时刻保持对各类安全威胁的有效防控。

三是高可用性。考虑到平台可能会被频繁调用，其必须确保具备高可用性、低延迟、稳定性以及鲁棒性。同时，在满足这些性能要求的前提下，平台还应尽可能地减少计算资源的占用，以实现高效且经济的安全防护，保障平台在高负载情况下仍能稳定运行。

四是可观测性。安全评测平台应具备对外展示的功能，允许用户实时监控屏蔽率、误报率、准确率、召回率以及计算效率等关键指标，让用户能够清晰地了解平台的安全防护效果和运行状态，从而更好地评估和优化平台性能。

#### 四、金融大模型安全治理

金融大模型存在内生安全风险，需要采取一系列

策略，如提高数据质量、加强模型训练、减少错误输出、进行安全评估和加强硬件安全等，以确保数据的安全性、算法的可解释性、模型决策的可靠性、应用的合法性和环境的安全性。金融大模型存在外部风险，需依靠国家法律、法规和行业标准来进行管理和规范。金融大模型与通用大模型在安全风险上存在差异，具体体现在专业性、数据安全与隐私保护、合规与监管、风险防控、可解释性与鲁棒性这几个方面（表2）。

##### 4.1 金融大模型安全防御技术

金融大模型的安全防御技术需兼顾内生、外生及衍生安全。内生安全防御技术（表3）关注数据安全、模型安全和系统安全。数据安全保护训练过程和交互过程中的数据安全。模型安全关注提高模型的抗恶意攻击能力。系统安全重点关注模型的运行环境安全性，包括模型系统的硬件和软件。外生安

表2：金融大模型安全与通用大模型安全的区别

安全风险	金融大模型	通用大模型
专业性	金融大模型专为金融领域设计，需要符合金融业务的专业性、严谨性及合规要求。通常涉及信用评分、投资决策、欺诈检测等专业金融任务，对数据的专业性、准确性具有更高要求。	通用大模型适用于广泛的应用场景，包括自然语言处理、图像识别等，对于数据要求泛用性和灵活性。
数据安全与隐私保护	金融大模型包含机构和个人的敏感信息，因此对数据隐私和安全的要求更为严格，一般需采用先进的隐私保护技术保证数据安全。	通用大模型适用于广泛的应用场景，包括自然语言处理、图像识别等，对于数据要求泛用性和灵活性。
合规与监管	金融大模型在金融领域的应用需要遵守更严格的法律法规和行业管理规范，因为金融大模型的应用直接关系到金融市场的稳定性和消费者的利益。	通用大模型涉及行业广泛，不局限于金融行业的规定。
风险防控	金融大模型具有较高风险防控的要求，需在模型研发全生命周期进行风险防控。	通用大模型的风险防控则更侧重于模型的泛化能力和跨领域的应用。
可解释性与鲁棒性	金融大模型需要更高的可解释性，因为金融决策通常涉及巨额资金和敏感信息，模型的透明度和推理过程的可解释性对于建立用户信任至关重要。	通用大模型更注重模型的鲁棒性和泛化能力，以适应各种不同的应用场景。

全防御技术(表4)主要应来自大模型外部的各种攻击威胁,保护模型及数据的完整性、可用性和隐私性。衍生安全防御技术(表5)则是在内生、外生安全防御基础上,深入探讨如何保障训练数据安全、提升模

型抵御恶意攻击能力、增强模型可解释性、保护模型隐私信息,以及确保模型运行环境和周边系统安全,涵盖部署环境安全、通信安全以及模型生成内容安全等内容。

表3: 内生安全防御技术

防御技术	防御技术子类	技术实现	说明
数据安全	数据隐私保护技术	数据脱敏	通过数据转换、数据截取和数据伪装等技术减少数据敏感性。
		数据匿名	通过对数据集中的部分数据集进行变换,使攻击者无法推导出个体的敏感信息,从而实现个体数据的隐私保护。
		数据加密	通过将原始数据通过加密算法转换成密文,确保数据的机密性和完整性,防止数据盗用。
	分布式训练技术	联邦学习	分布式参与方在不共享原始数据的情况下,通过加密参数交换共同训练模型。
		区块链技术	通过加密和共识机制,保障数据安全共享。
模型安全	大模型越狱防御技术	增强模型生成能力	通过引入复杂的加密算法和鲁棒性强的深度学习技术,增强模型生成能力。
		优化提示词输入	将系统提示词和用户输入的提示词进行拼接之后输入到大模型当中,优化大模型输入,减少错误输出。
		监测模型输入输出内容	通过监测大模型输入和输出的内容来及时发现并拦截潜在的越狱攻击行为。
系统防御技术	系统中硬件防御技术	漏洞修复防范技术	通过对抗训练增强模型对漏洞的抵抗能力,通过纠错码内存来克服漏洞
		被动检测防范技术	开发能实时检测硬件漏洞基于权重编码的框架。
		主动防范技术	通过积累漏洞库和识别已知注入故障,通过增加已知故障注入的示例,提高模型的学习能力,使模型主动防御攻击和帮助模型快速自恢复。
	系统中软件防御技术	用户数据防范技术	数据脱敏、数据加密、数据库安全审计系统等。
		网络服务防范技术	网络访问和网络应用的安全,支撑互联网应用的基础设施的安全。

系统防御技术	系统中框架设计 防御技术	深度学习框架漏洞检测	通过挖掘现有深度学习框架中的漏洞并进行修复
		底层依赖库漏洞检测与修复	通过挖掘深度学习常用底层依赖库中的漏洞并进行修复或替换
	系统中操作系统 防御技术	访问控制防范技术	通过对系统资源进行访问控制，来保证系统的安全性，包括身份认证、授权、审计等。
		机密性加密技术	对操作系统中的数据、文件以及运行机制加密是保护生成式人工智能系统的核心防御机制。
	系统中网络传输层 防御技术	端设备地址防御技术	增加源地址安全监测，确认IP地址的真实性和安全性。
		传输路径防范技术	确保数据在源地址到目的地址之间每个节点上的传输链路的安全性。

表4: 外生安全防御技术

防御技术	防御技术子类	说明
隐私安全攻击的 防御技术	对抗训练	通过在模型训练中引入对抗训练的示例，提高模型在对抗性攻击时的鲁棒性。
	提示工程	通过对提示位置的优化以及标识的改进，提升指令的鲁棒性，从而有效应对因大模型遗忘基线所引发的隐私泄露风险，从根本上加以解决。
	成员推理攻击	利用模型输出来推测训练数据，正则化和数据增强等技术可以防止过拟合，从而降低隐私泄露风险。
	引入差分隐私	通过添加噪声来减少模型对单个数据点的依赖性和敏感度，进一步保护隐私。
有毒数据的 防御技术	数据溯源和对齐技术	保证训练数据的可靠性和安全性。
	对抗算法	模型通过对抗算法识别出有毒的训练数据，并能自动排除有毒数据。
恶意后门的 防御技术	神经元监测技术	通过检查模型中的神经元激活特征，识别出被恶意操纵的神经元，并有效消除后门攻击。
	微调技术	通过模型的微调和再训练来清除这些恶意后门，增加模型鲁棒性。
	持续监控	通过持续监控并定期安全检查防止模型的后门攻击。
提示注入攻击的 防御技术	对抗训练	通过迭代收集攻击样本数据，并使用指令微调等方法对模型进行迭代的优化。

表5: 衍生安全防护技术

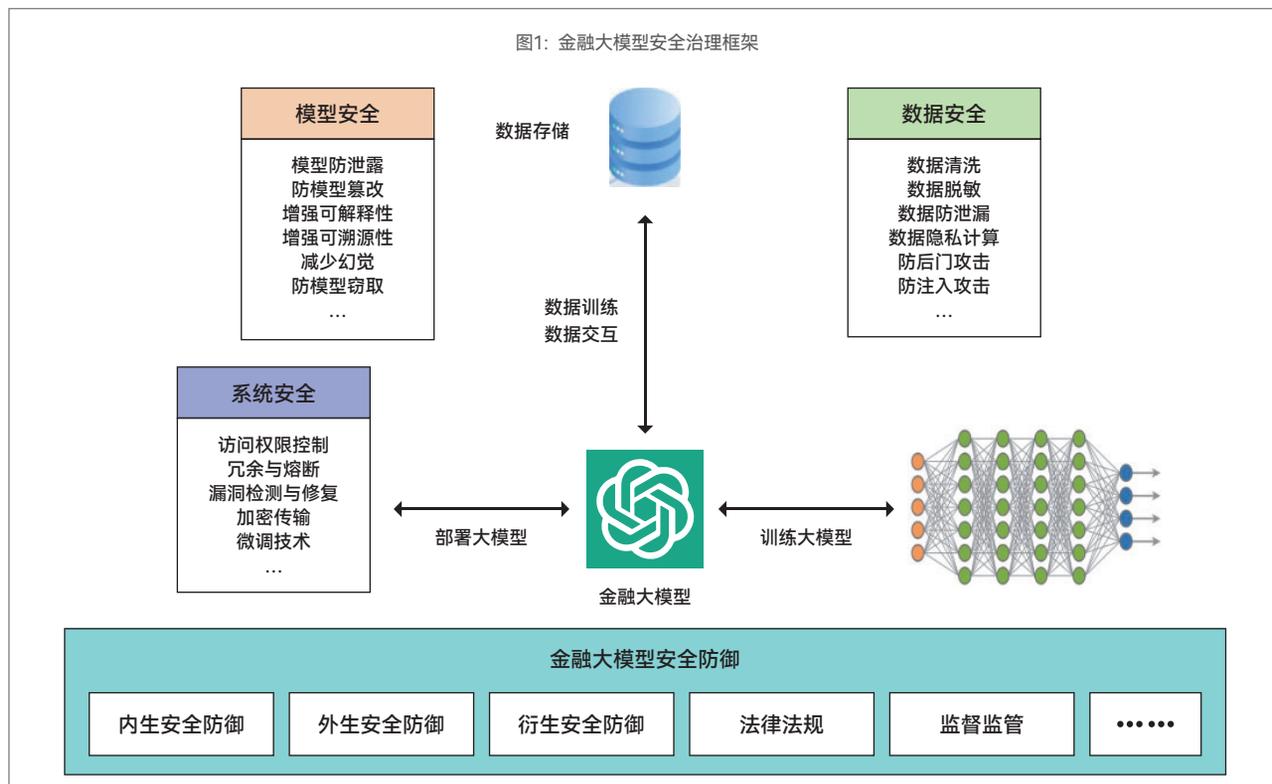
防御技术	防御技术子类	说明
偏见和毒性内容 防御技术	强化学习	基于人类偏好的强化学习技术通过人类的偏好反馈, 以强化学习方式优化语言模型, 引导模型的生成内容与人类价值对齐。
	指令微调	通过在指令中添加安全相关的规则和限制以减少模型有毒内容的输出。
虚假内容检测技术	事实核查	通过声明检测和证据检索判断模型输入内容的真实性, 并检测模型生成内容的真实性。
版权侵犯防御技术	数字水印	通过增加数字水印来保护数据版权, 防止数据在未经授权场景下的使用。

#### 4.2 金融大模型安全治理框架

金融大模型的安全治理是一项系统工程, 需在大数据模型构建、数据处理、网络安全及运行环境等多个维度上, 执行一系列综合性的安全策略, 并实施多层次

的安全防护措施, 以预防潜在的安全威胁。金融大模型安全治理可以分为金融大模型安全防护、数据安全、模型安全、系统安全这几部分(图1)。

金融大模型安全防护: 安全治理框架的底座是





大模型安全防御,它涵盖了内生安全防御、外生安全防御、衍生安全防御、法律法规和监管监督等方面,确保金融大模型在法律和监管框架内安全运行。

**数据安全:**数据在收集、处理和存储过程中,需进行数据清洗、数据脱敏、防泄漏、隐私计算、防后门攻击和防注入攻击,以保护数据的完整性、安全性和隐私性。

**模型安全:**在模型的设计和训练阶段,需实施包括模型防泄漏、防模型盗取、防模型篡改、增强可解释性和可溯源、减少幻觉以及模型可视化等安全措施。

**系统安全:**模型部署和运行的环境应实施访问权限控制、冗余及熔断、加密传输、微调数据、漏洞检测与修复等措施。

## 五、总结与展望

本文探讨金融大模型的安全挑战与风险治理。金融大模型在金融领域应用广泛,但也面临数据泄露、后门攻击、模型窃取等安全威胁。其安全挑战贯穿研

发生命周期,包括数据隐私安全、模型推理安全及内容合规等方面。文章提出安全测评方法,涵盖构建数据集、设计评测内容和部署评测平台三方面。金融大模型与通用大模型在安全风险上存在差异,主要体现在专业性、数据安全等方面。在安全治理上,需应对内生风险、外生风险和衍生风险,采取提高数据质量、规范法律法规、安全防御技术等措施。构建包含金融大模型安全防御、数据安全、模型安全和系统安全的安全治理框架,以保障金融大模型健康发展。金融大模型的安全风险治理是一个持续进化的过程,它需要金融机构、技术提供者和监管机构之间建立强有力的协作关系,通过跨行业的合作来确保金融大模型的安全、可靠和合规,进而促进一个成熟、稳固的金融大模型生态系统的形成。

责任编辑:王畅

# 探索参数运营1+4治理体系建设

路畏 / 郑州易盛信息技术有限公司

期货行业长期面临参数管控在安全合规、一致性及应急响应等维度的挑战。本着“做厚资产、做专管控、减少风险源”的原则，为此构建了“1+4”单元化治理体系：以参数单元化为核心，集成管理、发布、稽核、应急四大模块，实现风险源头治理。通过沉淀治理资产库，为AI驱动参数智能化运营提供数据基础，推动管控效能与业务安全的双重提升。

## 一、参数大运维建设成效

自2021年下半年起，启动参数大运维体系建设，重点解决参数管理低效、参数调整协同不畅、新品种上市跨部门协同多、业务对接标准缺失等主要问题和主要矛盾。通过优化流程、建立标准、强化协同，构建数字化场景组件，为参数管控筑牢安全堤坝。

经过近几年的深度建设落地以及持续优化迭代，参数大运维建设已形成三大数字化能力集。第一是从政策发布，到OA流转协同，系统自动拉取解析、预录入，再到一键审核生效的参数全流程自动化能力；第二是新品种上市“一录多用”、参数调整协同、参数管控数字化组件等一批数字化场景能力；第三

是自动生成及推送参数运营生命周期内基于人为因素、系统因素引起参数变化的参数运营闭环报告。

参数大运维建设落地，有效支撑了每年万余次的参数调整运营，且实现零差错。充分彰显了参数大运维建设在控风险、提效率的关键作用，有效支撑了交易结算连续性、稳定性。

随着参数平台纳管参数业务的激增，原有手工分析模式面临严峻挑战：横向覆盖7个业务部门及全量生产业务，纵向贯穿参数调整、发布、业务使用（业务加载时机、加载通道、参数被使用）等全链路，导致影响分析耗时耗力。曾经的次要问题已升级为参数安全管控的核心痛点，亟需系统性解决方案。另外，也

图1：参数大运维建设范围



意识到要想在参数运营中落地有效的大模型应用场景，必须要有参数领域专业知识库做支撑。

## 二、逐渐凸显的参数运营治理问题

通过业务方与技术方开展“观全貌、理结构、找变化、树边界、立标准”多频次沟通交流，总结出参数“新常态”问题聚焦在以下三个方面：

### （一）参数治理流程存在断点

当前的治理模式主要局限于参数内部这一亩三分地的管理和优化，缺乏对参数调整可能引发的外部业务影响进行深入分析、追溯的能力。同时，缺少完善的告警机制，以及对涉及的相关岗位和组织的明确定位。这种局限性导致在需要跨系统、跨岗位、跨部门的线上线下协同工作时，缺乏有效的分析工具和抓手，制约了治理赋能的进一步提升。

### （二）参数治理存在标准、原则、工具缺失

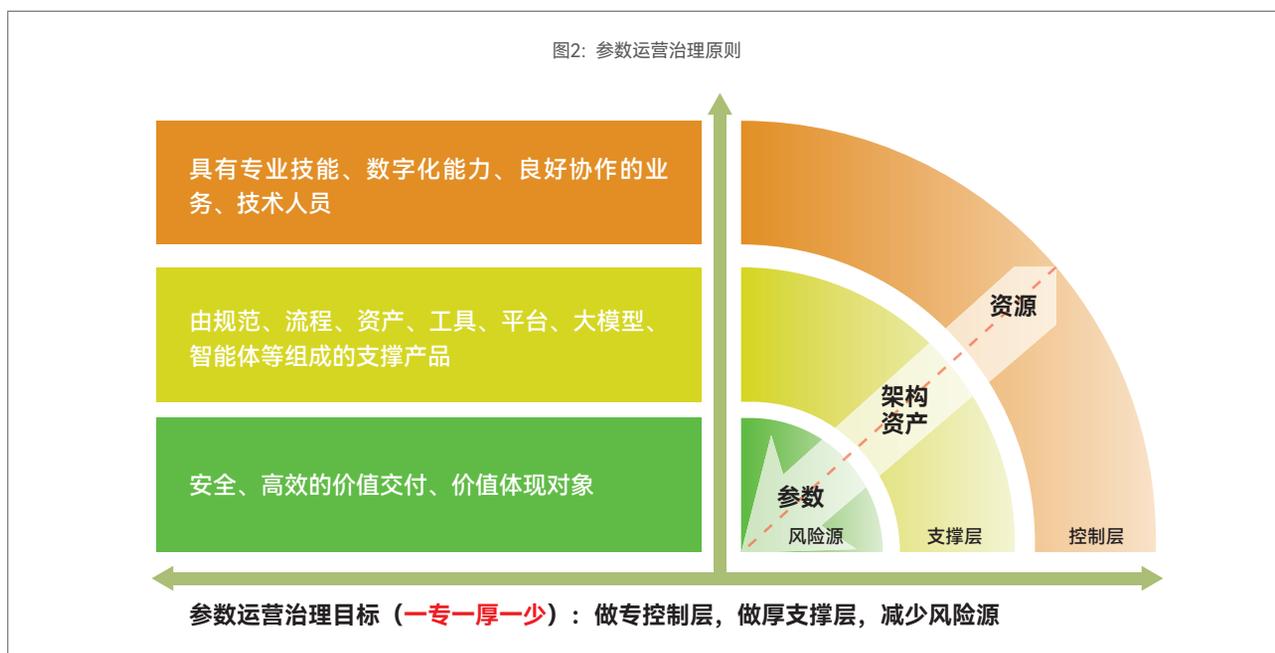
随着业务创新效率和研发效能的不断提升，业务与研发之间以及研发团队之间的沟通效率问题日益显现。一是概念、术语的理解不统一。在平台建设过程

中，统一了很多概念、规范以及标准，但人员认识上存在差异，协作过程中还需要多次交流才能达成共识。二是贯标能力弱。目前，参数单元化治理的结果仅停留在文档层面，依靠文档形式来约束规范性和标准性，其实际管控力较弱，容易被破坏。三是重管理，轻使用。现阶段注重参数管理，但缺乏“谁使用，谁备案，谁负责”的参数使用原则。这种缺失导致参数在使用监管方面存在安全性与合规性的隐患，特别是参数变更时对业务影响分析的准确性方面，问题尤为明显。

### （三）参数知识库质和量有待提升

近些年的平台建设，已形成了参数单元关系地图、参数全局档案信息、参数接入规范、参数运营指导以及研发效能平台等资产、工具，但在结构化、标准化、系统化以及专业性方面还需提升。纵观现有积累，未能有效融入业务关联、业务场景特征、流程节点及价值链路，导致业务分析模型、指标映射关系等核心资产缺位，制约数据驱动业务决策的能力，这方面需要进行补充。

图2：参数运营治理原则



### 三、打造参数运营治理体系化解决方案

为解决凸显问题并深入推进数字化转型建设，本着参数运营治理“一专一厚一少”的原则，以厘清参数治理边界、做厚架构资产支撑、探索大模型赋能为核心，探索参数运营治理体系化建设。

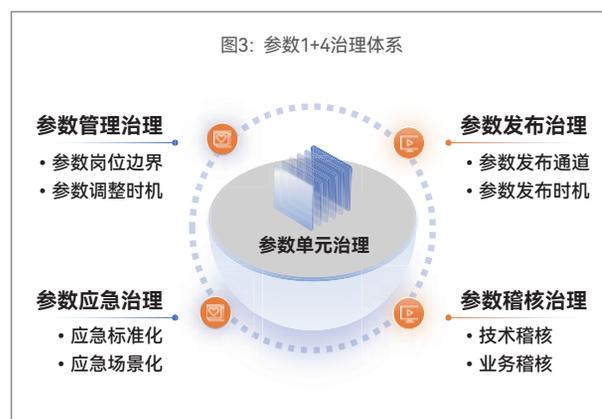
(一) 锚定治理闭环，厘清治理边界，统一治理概念

结合现有交易结算连续性过程，理清整个参数参与过程，进行统筹规划。

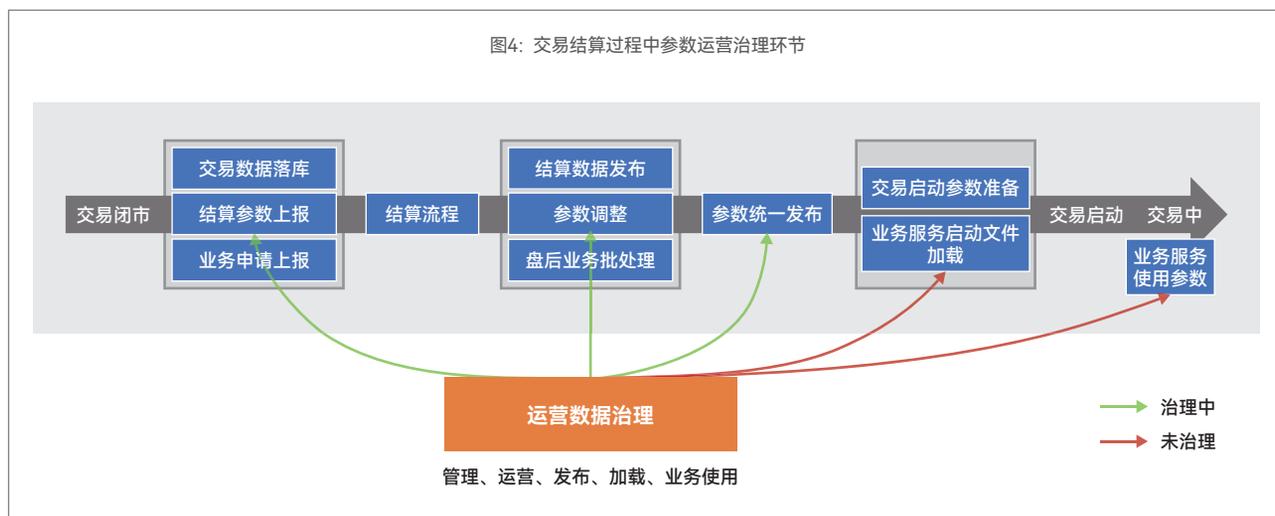
首先，锚定治理体系闭环。构建全面的参数运营治理体系，要突破以参数为边界的治理模式，把业务考虑进来，填补治理流程割裂与不连贯的空白，确保从“参数的精准配置”到“高效传输”，再到“在业务中的正确应用”，形成一个无缝衔接、闭环管理的完整治理链条。这样的治理体系旨在通过强化参数与业务场景的深度融合，防范参数、业务操作风险，提升运营效率和准确性，形成联动效应，实现参数运营治理闭环。

以参数单元化治理为基础、为抓手，形成参数单元内和单元间关联关系。在参数管理方面重点治理参数岗位边界以及参数调整时机，形成岗位、参数、时机

的清晰画像。在参数发布方面重点治理对外发布通道以及发布时机，形成参数、通道、时机以及业务服务的清晰关系。在参数稽核方面，重点建设技术稽核能力，并探索业务稽核模式，形成技术与业务双稽核相互支撑的局面。在参数应急方面，打造场景化应急，落实参数与业务场景线上线下影响视图。从而形成以参数单元治理为核心的1+4参数治理闭环体系能力。



其次，厘清参数治理边界。参数在整个交易结算过程中涉及参数管理、参数运营、参数发布、参数加载、业务服务使用参数等五个环节。其中管理、运营、发布等环节已在运营治理中，参数加载以及业务服务使用参数还未纳入治理范围内。



最后,明确统一治理概念。语言是文化的外衣,它紧紧包裹着文化的内核,让文化在交流与传承中得以妥善保存并延续发展。而概念则如同基石,是体系升华和建设的的关键所在。如果概念不统一,治理就会变得混乱,标准难以制定和执行,体系化更是无从谈起。从治理的角度出发,应明确治理的核心环节、视角和主线,围绕关键治理项逐步构建体系化的内容。

(二)打造可靠架构,强化平台赋能,盘活人力资源

现在数字化参数运营平台已形成了开放性、可扩展性和可定制性的数字化参数运营基座。

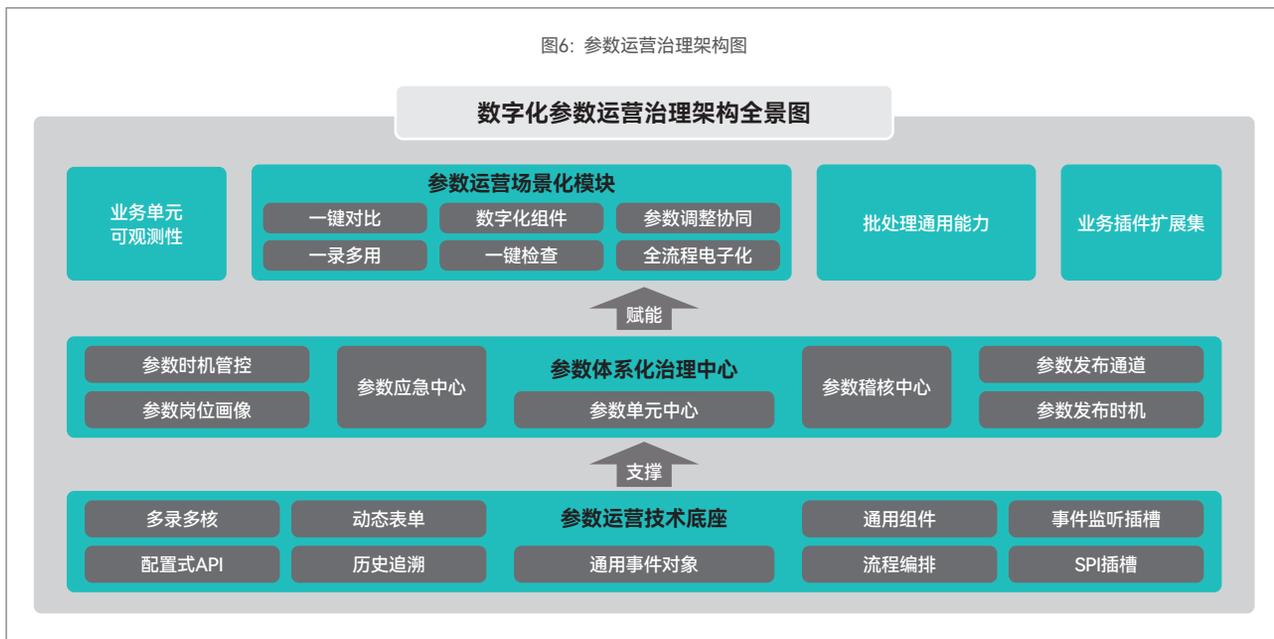
既可以当作功能模块嵌入到业务平台中,也能在基座上开发数字化新应用。

在此基座之上,树立以参数单元为抓手的体系化治理中心。刻画出一个参数单元的管控时机、关联岗位画像、对外发布时机和发布通道,以及以参数单元为维度的稽核能力和应急能力。

图5: 参数运营治理闭环

治理环节	视角	主线	治理项
参数管理	参数	以参数录入为核心	参数录入时机, 参数管理权限, 参数录入通道, 参数规则, 参数范围, 参数间影响。
参数发布	参数	以业务使用参数为核心	参数发布时机, 发布通道, 发布内容, 发布目的地, 发布正确性。
参数运营	参数	服务参数管理与参数发布衍生出来的一些场景	参数录入应急, 参数发布应急, 参数单元间管理, 参数单元内管理, 参数模型管理, 辅助参数管理与发布的数字化场景集。
业务加载参数	业务	以业务获取参数为核心	参数数据源、业务参数存储方式(DB、内存、文件等), 加载时机、加载通道、加载范围、加载应急流程,
业务使用参数	业务	以业务发生时使用参数为核心	业务使用参数对象范围, 业务发生时用到的参数项值,

图6: 参数运营治理架构图



在参数运营场景化方面，充分利用参数单元化的治理能力，作为数字化场景化建设的“生产物料”。比如新品种上市“一录多用”模式，取代了各个交易所背靠背、两录一核的参数录入模式，通过流程编排相关参数单元来实现参数一键录入，内置单元化应急、单元化展开、单元化观测等处理能力。让用户真正的感受到什么叫“看的见”、“摸得着”、“管得住”，大幅提升业务人员的安全感、幸福感。

在参数全链路波及分析方面。以参数单元为出发点，结合业务运营驾驶舱以及治理要素形成参数全链路波及分析图。在参数变更、业务调整、业务创新时，需求侧、研发侧、测试侧以及运维侧均可以显示化的

看到所影响的参数项、参数单元、运营任务、业务系统以及关联的业务部门。此功能也作为人员岗位技能评估以及参数应用成熟度评估的重要度量抓手。

(三) 激活治理资产，探索大模型参数运营落地场景

大模型在参数运营领域有很多应用场景，比如，在政策制定时通过大模型自动识别关联影响的业务，明确对业务影响范围以及需要协作的业务部门；在参数调整环节，自动识别受影响的关联参数，以及关联的线上、线下业务，确保各部门对参数调整的影响范围达成一致，避免因沟通不到位导致的业务风险；在参数系统应急环节，实现事前预防，事中辅助及自动决

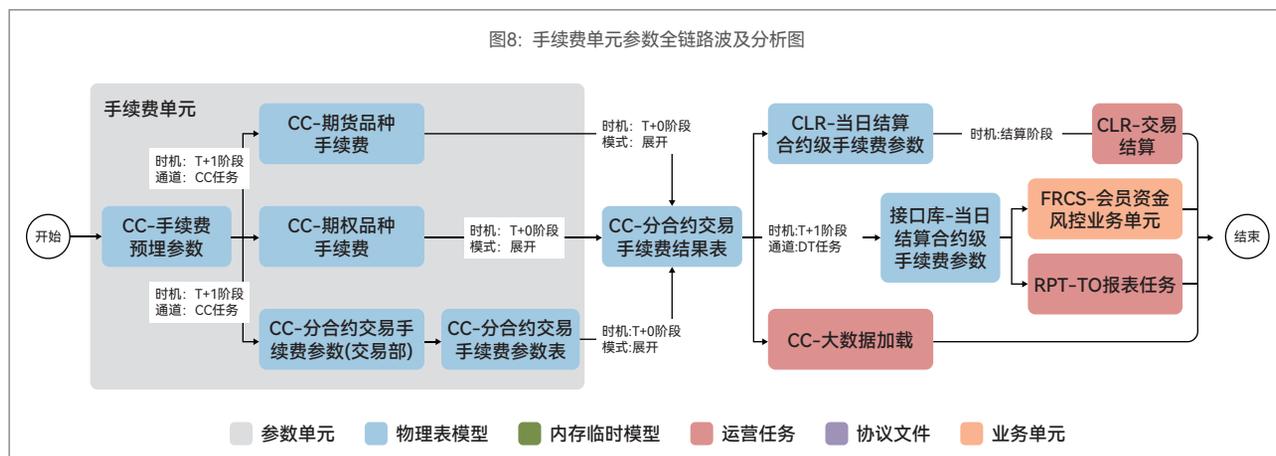
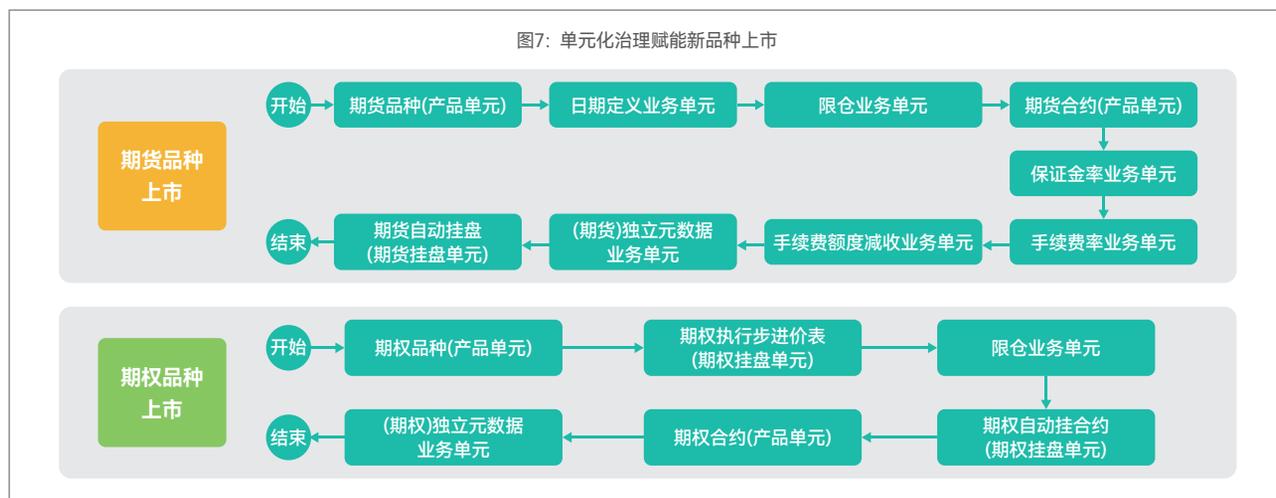


图9: 参数治理知识库大模型应用畅想



策处理, 事后总结以及专业知识库自动更新与优化。

所以要想能够发挥大模型的功效, 必须要有行业的基础知识、领域知识以及专家知识形成的知识库做基础。下图为基于知识库进行参数波及分析的应用模式畅想。

#### 四、写在最后

经过对参数运营1+4治理体系的实践探索, 目前已初步构建起参数单元化治理模式。该模式实现了三大突破: 一是参数管控由平铺式升级为结构化单元模式, 并与业务领域关联绑定; 二是通过事件驱动技术解耦参数单元间关联, 建立全局管控时机机制; 三是实现应急场景的单元化封装与可视化呈现。这一创新显著提升了业务、研发、运维三方的协同效率, 推动组织认知达成统一。治理颗粒度从宏观走向微观, 从规范化走向资产化, 为数字化运营提供了精细化管控新范式。

参数运营治理体系化建设是一条既务实又长久的道路。结合治理过程形成的知识库与工具智能体支撑, 逐渐的朝着参数运营智能化迈进, 最终实现参数运营模式从“碳基”转向“硅基”。

责任编辑: 刘译获

# 云原生微服务架构下的性能测试实践

沈红富 / 博时基金管理有限公司

本文介绍了微服务架构下性能测试实践,包括背景、转型动因、转型过程、实践和展望等方面。通过压力测试工具的转变、可视化监控工具、APM工具、日志工具等解决性能测试难点问题,取得了一定的成效,快速解决性能问题。未来将进一步优化测试工具,并结合人工智能的应用,提高性能测试效率并改进软件质量。

## 一、背景介绍

IT系统构建模式呈现明显的动态演进轨迹,其发展脉络从早期集中式部署的单体系统逐步向分布式计算体系转型,进而演变为基于服务总线集成的面向服务架构。在近年技术革新浪潮中,以模块解耦和接口标准化为特征的微服务范式得到广泛应用。值得关注的是,云原生技术体系凭借其特有的资源弹性调度能力和自动化运维特性,正在重塑现代软件工程实践范式。该体系通过容器化部署、服务网格等技术机制,显著优化了系统迭代周期与交付效能,为应对复杂业务场景下的高并发需求和高可用性挑战提供了创新解决方案。

云原生技术体系通过整合服务解耦设计、容器、DevOps流程及持续交付管道等核心组件,致力于深度释放云平台弹性计算潜能,系统性提升软件从构建到运维全生命周期的执行效率与可扩展能力。该架构通过资源动态调配与流程标准化设计,有效保障了系统在复杂环境下的快速响应与稳定运行。



## 二、机遇和挑战

在微服务架构下,性能测试的转型既是必然趋势,也是提升性能测试效果的机遇。技术演进路径由单体架构与微服务架构的本质特性共同驱动,而后者在实现模块解耦优势的同时,也衍生出分布式系统治理、服务间通信复杂度提升及容错机制强化等关键性技术挑战。

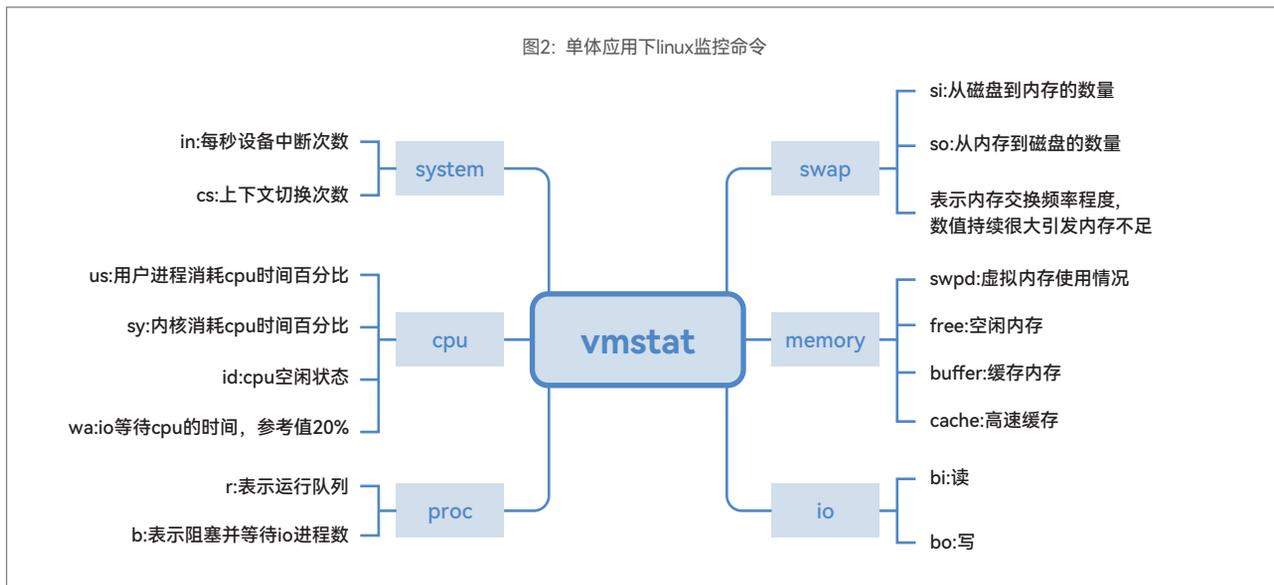
1) 单体架构的特点: 单体应用架构下,测试策略相对简单,集中测试一个整体应用,不好实现单个服务的压力测试;压力测试工具选择较少,以商业工具为主,开发脚本比较复杂;监控工具也比较少,主要使用Linux或JDK自带工具。但环境部署简单,服务器少,部署更新一个或者两个目录文件即可,相对性能瓶颈容易定位,因为所有功能和数据都在同一个应用或数据库中。

2) 微服务架构的特点: 相较于单体架构,微服务架构的核心特征体现在其将单体应用分解为多个松耦合的独立服务单元,各服务通过标准化API接口实现解耦交互,具备独立开发、部署及扩展能力。每个服务器可以独立升级、扩展或重启,提高系统弹性和灵活性,支持按需扩展的模式,高效应对不同流量压力。

### 3) 转型动因

针对这样的架构特点,转型就应运而生。

图2: 单体应用下linux监控命令



单体应用的局限性: 维护性差、团队协作困难、部署周期长。

微服务架构的优势: 通过服务拆分与独立部署, 显著提升系统的可维护性与弹性扩展能力, 同时通过松耦合设计降低组件间依赖风险, 并优化资源分配效率以实现计算资源的精细化利用。

图3: 单体应用和微服务架构对比



#### 4) 微服务架构的性能测试挑战

当然转型有很大优势, 但也带来相应的问题。比如, 监控分析不方便: 服务数量增加, 全面监控困难; 问题瓶颈难以定位: 微服务间调用关系复杂, 性能问题分布在整个调用链中; 日志不方便查看: 日志数据分散存储于多服务节点, 亟需构建高效日志管理与智能分析工具, 以实现日志数据的集中化处理与快速定位。

### 三、转型过程

尽管微服务架构具有可扩展性、资源利用率高等各类优势, 但其分布式特性会导致原有性能测试过程需要进行全面调整。我们在测试策略、压力测试工具、测试环境和监控工具等多个方面进行调整, 以适应分布式架构带来的挑战。

①测试策略: 模块化单元验证、服务链路交互测试及全链路压力模拟相结合的方式, 通过逐层验证单服务功能、服务间通信可靠性及系统整体承载能力, 确保微服务架构在复杂场景下的稳定性与弹性。

②压力测试工具: 主要使用开源JMeter, 支持分布式架构的工具如Locust。

③测试环境: 容器化测试环境, 使用K8S管理测试资源。

④监控工具: 引入如Prometheus等可视化监控系统, 实现对微服务运行状态的实时监测与动态性能追踪。

⑤APM工具: 无入侵性, 快速定位性能瓶颈, 提供详细的操作和SQL查询信息。

⑥集中日志工具：将日志汇集到中心，结合开源组件实现全链路跟踪。

通过表格，可以较好的查看单体应用架构和微服务架构两者的对比情况。

图4：单体和微服务性能测试对比

类型	单体应用	微服务架构
测试策略	测试一个应用或者整体的性能	测试一个服务的性能，粒度更小
测试环境	虚拟机环境，虚拟机部署	K8S集群，容器化部署
压测工具	商业工具，支持各类协议	开源Jmeter为主，主要HTTP协议
监控工具	Linux命令或者JDK自带插件	每个微服务均可可视化监控，如Prometheus、Grafana
链路分析工具	手工定位	APM工具
日志查看工具	手工查看、筛选	集中日志平台

#### 四、实践案例

##### 1) 微服务架构下性能测试整体流程

在新的架构下，性能测试的整体流程变化不大，但是业务场景更加完善、测试模型更加全面、对系统性能评估更加严谨。构建从需求分析到设计、执行、

优化、回归验证和报告输出的全流程性能测试体系。需求分析中，明确测试目标是整个流程中最关键的活动之一。如对外提供服务的系统性能测试时，测试目标就需要明确多大并发用户下，系统占用资源和系统处理能力，只有满足相应的性能指标，才能认为通过性能测试。如果指标选取不合适，不仅可能导致测试结果出现偏差，还可能导致整个测试过程的失效，甚至需要重新执行整个性能测试。

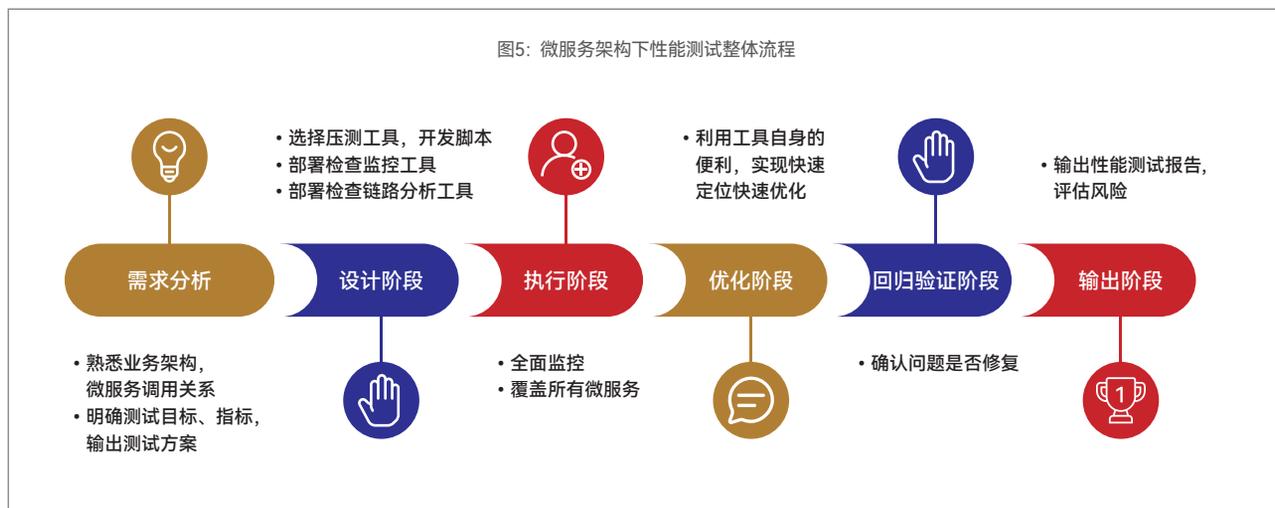
建立性能测试准入机制，将性能测试报告作为生产环境发布的前置条件。

从图5中可以看出，架构的转变，性能测试的转变也渗透到每一个阶段，带来快捷的同时，也相应带来一些额外成本，如学习成本、需求沟通成本。

##### 2) 实践项目介绍

项目一：微服务架构模式显著提升了资源监控的便捷性与指标覆盖的全面性。在APP接口项目从单体架构向微服务架构迁移过程中，主要通过将应用服务器拆解为多个独立服务组件，实现业务功能的模块化拆分与解耦设计，从而有效增强系统的容错能力与横向扩展能力。

针对性能测试，压力测试系统时，架构的改变就使得系统性能提升了1.5倍。





项目三：微服务的快速伸缩机制，也让测试资源能够快速调整，提升性能测试执行的效率。在大数据指标中心压力测试项目中，响应时间超过5min，通过增加后台服务的pod个数，快速完成资源增加等操作，使响应时间缩短到3秒，完成性能的极速调整和验证。



图12: “净值与时序规模”响应时间约5min

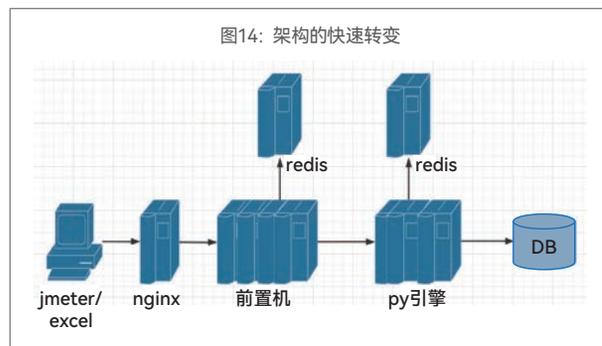
Label	# Samples	Average
login	1	136
Brinson多期日归因_主表	1	6828
CAMPISI归因模板-批量分析	1	44885
非公募货币摊余组合总表	1	28
公募基金管理人压力测试总表	1	88759
净值与规模时序	1	291295
组合TOPPICK配置	1	3293
业绩及风险指标	1	2462
总体	8	54710

图13: “净值与时序规模”响应时间3.2秒

Label	# Samples	Average
login	1	52
Brinson多期日归因_主表	4	2704
CAMPISI归因模板-批量分析	4	6552
非公募货币摊余组合总表	4	15
公募基金管理人压力测试总表	4	66617
净值与规模时序	3	3221
组合TOPPICK配置	3	14
业绩及风险指标	3	2215
总体	26	12306

当我们进一步压力测试时，发现增加引擎服务的数量时，系统性能并没有明显提升，增大并发，响应时间几乎不变，TPS也不再增加。通过深入分析和测试，我们发现瓶颈出现在前置服务层。

为此评估对架构进行调整，需要增加多台前置机，并用nginx进行负载均衡。架构调整后，压力测试1万个指标，系统性能有了显著的提升，压力测试场景从66秒缩短到了10秒。通过容器平台，容器化部署nginx，增加前置机pod个数，实现新增服务、架构、资源的轻松调整。



在实际压力测试过程中，遇到资源瓶颈时，无须额外走资源申请流程，可以在容器化平台就能快速实现资源的快速扩展或缩减，通过分析资源配置参数对系统性能的量化影响，优化性能测试方案以缩短整体执行时间。

## 五、总结和展望

### 1) 微服务性能测试总结

通过三个项目实例分析，可以发现性能测试效率提升离不开自动化工具、随时可用的压力测试环境和标准化流程。自动化监控和APM工具快速定位性能瓶颈，提升执行效率。微服务特有的分布式架构特点，使得压力测试环境的易于扩展，提升发布问题后的优化效率。标准化流程能有效减少性能测试人员与开发团队的冲突，提升性能测试的沟通效率。

微服务架构给性能测试过程中带来灵活性的同时，也给性能测试增加新的成本与挑战。一是测试方案复杂性增加，业务流程更加复杂，链路调用更加复杂，导致压力测试场景更加复杂。二是测试辅助工具

成本高，需要集中日志管理和APM等辅助工具，需要专人维护。三是学习成本高，需要学习微服务和各种工具。

## 2) 展望

图15: AI技术优化SQL



首先, 优化性能测试流程, 赋能功能测试人员, 将自动化性能测试融入日常版本测试中;

其次, 性能问题能及时解决, 比如引入AI技术, 分析复杂的SQL查询和数据库操作; 明确需求, 提前考虑确认性能指标;

最后, 结合行业压力测试指南, 增加系统的可靠性测试, 评估系统在异常情况和长时间运行的性能表现。

总之, 云原生微服务架构下的性能测试虽然充满挑战, 但通过合理的工具选择和测试策略, 基本可以满足业务需求, 并取得了一定的成果。未来计划通过改进性能测试的执行流程与自动化工具, 以增强系统稳定性并提升架构可扩展性。🌀

责任编辑: 赵馨璐

# API安全防护现状及技术解决方案

胡利民、曹祥波、黄辉 / 湘财证券股份有限公司

金融行业在国民经济中处于核心地位，是实体经济与资本市场相连接的关键纽带。券商核心系统作为金融行业核心业务的基础设施，为业务运营提供了坚实保障。本文阐述了券商核心交易系统建设过程中的API 安全现状及 API 安全治理的相关解决方案，期望为金融行业及其他系统的安全建设提供可供参考的解决思路。

## 1.概述

随着金融科技的快速发展，API（应用程序接口）在金融行业中扮演着越来越重要的角色。API接口不仅促进了金融服务的开放和创新，还极大地提高了金融业务的效率和便捷性。与此同时API接口的安全性和性能优化问题也日益凸显，成为金融行业必须面对和解决的挑战。本文将从API接口的基本概念出发，结合券业信息系统建设过程中API的实际应用场景，深入探讨金融行业API接口的安全性问题，并提出相应的解决方案和建议。

## 2.定义与术语

专业术语	专业术语解释
SDP网关	SDP是国际云安全联盟CSA在2013年提出的基于零信任理念的新一代网络安全模型，强调通过软件定义边界来替代传统防火墙的物理边界
API网关	API网关仿若系统间的“门卫”抑或“中介”，承担着管理与把控对后端服务访问的职责，确保请求能够精准抵达目标服务。通俗而言，API网关恰似餐厅中的服务员，客户（前端应用）向服务员（API网关）表明所需菜品（调用的后端服务），服务员（API网关）便前往后厨（后端服务）将菜（数据或响应）端予客户（前端应用）。在此过程中，服务员还需处置一些额外事务，例如核验客户订单的合理性（安全验证）、向厨师传达客户的特殊要求（请求参数转换）、记录受欢迎的菜品（日志记录）。
GDPR	通用数据保护条例(General Data Protection Regulation)

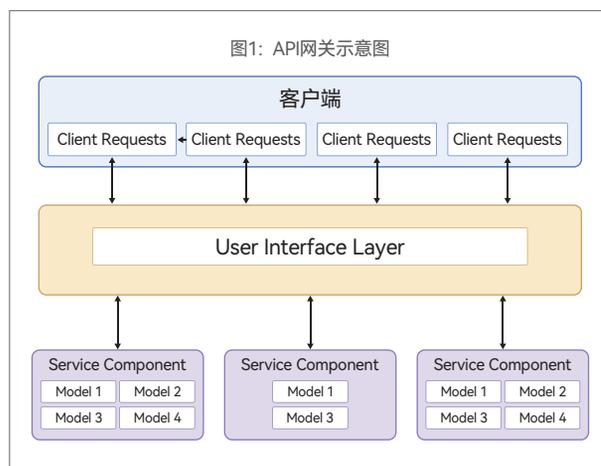
## 3.API基本概念

### 3.1什么是API

API(Application Programming Interface) 指的是应用程序编程接口，即一组定义和描述不同软件组件如何通信以及相互操作的规范。它能够让不同的软件系统共享数据和功能，从而实现相互连接和交互。API可以是用于操作系统、数据库、Web服务、程序库等各种软件组件的接口。

### 3.2什么是API网关

API网关是一种服务器，充当应用程序编程接口（API）的入口点，用于管理和控制对后端服务的访问。它协助API的调用。API网关统一管理下属的微服务接口，将请求分配给相应的服务，并收集结果传递给请求者。



API网关的主要功能包括:

1.路由和转发:API网关接收来自客户端的请求,并根据预定义的规则将请求路由到相应的后端服务。

2.负载均衡:API网关可以将请求分发到多个后端服务实例,以实现负载均衡,提高系统的可伸缩性和性能。

3.安全认证和授权:API网关可以处理身份验证和授权,确保只有经过身份验证和授权的用户才能访问后端服务。

4.缓存和缓存管理:API网关可以缓存响应数据,以减轻后端服务的负载,并提高响应速度。

5.监控和日志记录:API网关可以收集和记录请求和响应的指标和日志,用于监控和故障排除。

6.API版本控制:API网关可以管理不同版本的API,并根据客户端的需求将请求路由到适当的API版。

### 3.3 API网关分类

API网关根据功能与用途可划分为以下三类,三者在实际场景中可能交叉协作以满足复杂需求。

#### 3.3.1 对外的API业务网关

功能:

- 作为外部客户端(如移动应用、网页、第三方服务)与内部服务的中介,处理请求路由、协议转换(如HTTP到gRPC)、流量控制(如限流和缓存)等

- 提供统一入口管理,简化客户端调用逻辑,支持跨域请求处理。

特点:

- 高性能与扩展性:需应对高并发外部请求,支持横向扩展以提升吞吐量。

- 安全性:集成基础身份验证(如OAuth2.0、JWT)和IP黑白名单,防范外部攻击。

#### 3.3.2 对内各应用程序调用的API应用网关

功能:

- 管理微服务间通信,提供服务发现、动态路由、负载均衡、熔断降级等机制。

- 支持服务编排和灰度发布,降低服务依赖的复杂性。

特点:

- 稳定性与可用性:通过负载均衡和故障转移保障内部服务的高可用性。

- 轻量化协议支持:优化内部通信效率,例如支持WebSocket或高效二进制协议。

#### 3.3.3 安全层面的API安全网关

功能:

- 身份鉴权与授权:严格验证请求来源(如基于角色的访问控制),防止未授权访问。

- 数据加密与审计:对敏感数据传输加密(如TLS),记录详细日志用于安全审计。

特点:

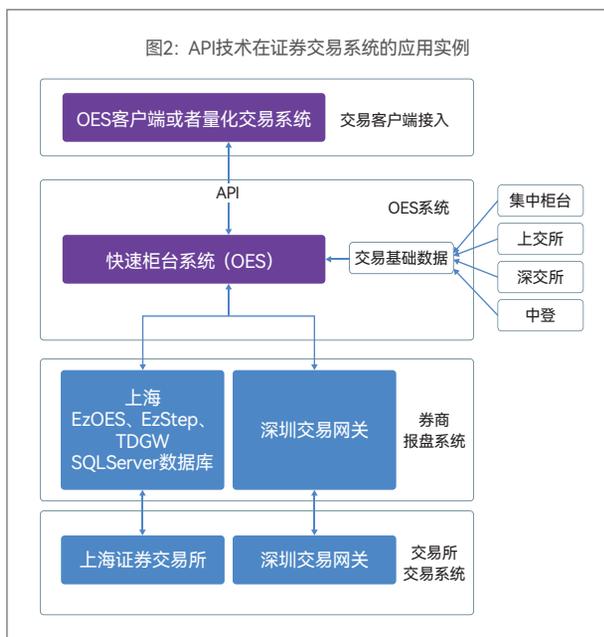
- 深度安全防护:集成防重放攻击、防数据泄露等机制,支持细粒度权限管理。

- 合规性:满足GDPR等数据安全法规要求,保障数据隐私。

在实际应用中,这三类API网关可能并不是完全独立的,而是相互交织、共同协作的。业务网关可能集成安全网关的鉴权功能,安全网关也可能依赖应用网关的服务发现能力。

### 3.4 API技术在证券交易系统的应用案例

为提升交易效率、优化客户体验并增强市场竞争力,我司面向机构投资者、高频交易者与量化基金,凭借API技术构建了一个实时交易平台。该平台致力于提供具备极低延迟、高吞吐量及高可靠性的高性能交易系统,旨在为客户提供低延迟的交易执行服务。

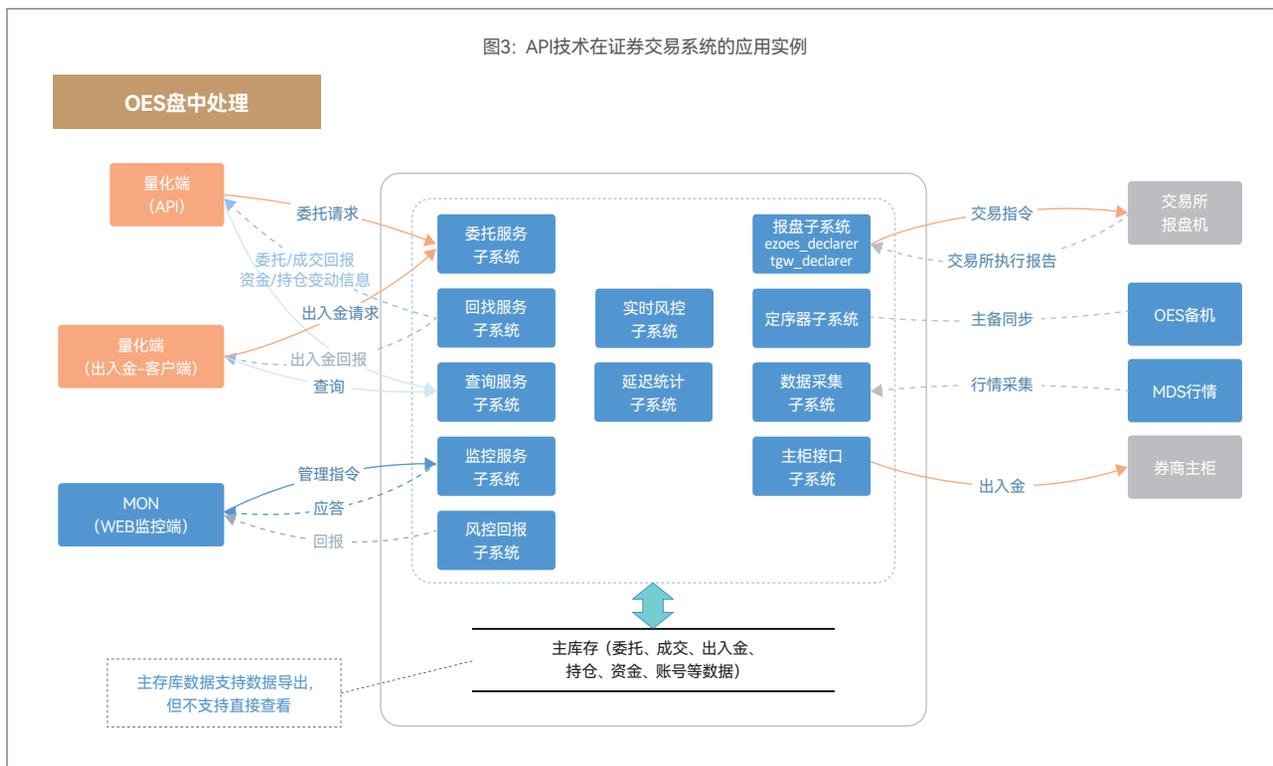


客户（投资者）通过交易API接口构建自己的交易系统，进行股票、融资融券、期权的买卖操作。

交易	该证券交易系统提供了交易API，允许客户通过程序化接口进行股票、融资融券、期权的买卖操作。提供丰富的API接口，支持用户自定义交易策略和系统集成，支持多种编程语言，如C++、Java、Python等，方便开发者快速上手，量化交易者通过交易API，构建自己的交易系统，自动执行买卖指令，减少人为干预，提升交易执行效率
实时行情数据获取	API接入交易所实时获取股票，提供了实时市场数据API，包括股票价格、成交量、买卖盘口等信息。并提供行情接口API确保客户能够及时获取最新市场动态
风险控制	提供实时风控API，系统实时监控交易活动，确保合规。某笔交易风险超过预设阈值时，系统自动终止交易。并通过短信、微信等方式推送给投资者，以确保经营活动的合法合规

API技术在证券交易系统自动化交易与实时行情获取中的应用，显著提升了交易效率和策略准确性。通过实时行情获取、自动化交易、风险控制，机构投资者、高频交易者与量化基金能够更好地管理交易活动，优化投资策略，提升投资回报。

该交易系统自上线以来，以为客户提供高效、



便捷、优质的服务为宗旨，不断优化和完善自身功能。在实际运行过程中，凭借其强大的数据分析能力和智能化的操作流程，成功解决了客户在业务处理中遇到的诸多难题，大幅提高了工作效率，赢得了客户的广泛赞誉和信赖。

#### 4. API 安全防护的现状及技术解决方案

##### 4.1 API 安全的定义

API安全指的是为保护API免受恶意攻击和滥用而采取的一系列安全措施，旨在确保API在使用和交互过程中不会受到未经授权的访问、数据泄露、篡改以及其他安全威胁的影响。

##### 4.2 API威胁的定义

API威胁是指借助API在设计、实现或管理方面存在的缺陷，通过非法访问、篡改数据或滥用服务等手段，对应用程序、数据或业务逻辑带来潜在损害的情况。

##### 4.3 API 安全防护现状

在各类金融服务组件中，API被广泛运用，以达成系统间的数据交换与功能调用。在证券领域，APP、小程序以及第三方业务开放平台等向外部提供了大量的API接口，以供客户进行查询或办理业务，其中涵盖了手机号、银行卡号、身份信息敏感数据。在部分实际业务场景中，金融API数据安全防护手段及现阶段面临的核心问题如下

业务场景	业务形态	传统防护手段	传统防护机制现状	现存核心问题
表单字段限制	Web/APP/H5/小程序/公众号/微应用前端轻量化，服务端全面API化成为技术主流	WAF类产品	<ol style="list-style-type: none"> <li>1.已知攻击拦截：通过预置规则库识别SQL注入、XSS、DDoS等常见攻击行为，满足等保2.0对应用层防护的强制要求</li> <li>2.自动化日志与审计：生成标准化日志和报告，简化合规流程</li> <li>3.多业务渠道覆盖：支持Web、APP、H5、小程序等前端接入方式</li> </ol>	<ol style="list-style-type: none"> <li>1、高频交易场景：高频交易依赖于极低的延迟和大量的API请求，WAF可能会引入额外的延迟，影响交易系统，另外，过于严格的防护规则可能导致合法交易请求被误判为攻击。因此在这样的场景下，通常不会采用WAF进行防护。</li> <li>2、复杂的业务逻辑场景：一些复杂的业务逻辑（如订单处理、清算结算等），WAF可能无法完全理解这些逻辑，因此WAF难以检测基于业务逻辑的攻击。</li> <li>3、数据泄露场景：由于WAF不会对API响应中的敏感数据进行检测，因为攻击者可能通过合法的API请求获取敏感信息，成为数据泄露的入口。</li> <li>4、API认证和授权场景：API通常需要严格的身份认证和授权，但是WAF无法覆盖这些需求，WAF无法检测令牌泄露或无效授权。</li> <li>5、第三方集成场景：证券行业的应用，通常与第三方系统有集成，WAF无法监控第三方API调用的安全性。</li> </ol>
		风控类产品	<ol style="list-style-type: none"> <li>1.通过代码混淆、加密传输等技术，实现H5、小程序等轻量化前端的基础防护，防止页面篡改、链接伪造等攻击。</li> <li>2.采用设备指纹、人机验证（如验证码）等手段，拦截自动化脚本攻击（如撞库、薅羊毛），降低业务欺诈风险</li> <li>3.对API接口返回的敏感数据（如身份证号、手机号）实施动态脱敏，避免信息泄露</li> <li>4.通过HTTPS中间人攻击防护和本地存储加密，保障数据传输与存储安全</li> <li>5.提供标准化日志审计功能，满足《数据安全法》等法规要求，例如记录用户访问行为并生成风险报告</li> </ol>	<ol style="list-style-type: none"> <li>1.传统风控依赖规则库，难以应对业务逻辑层的攻击（如虚假评价、恶意差评），需结合上下文行为分析提升精准度</li> <li>2.跨平台攻击（如通过H5伪造跳转至钓鱼页面）的识别能力有限，需强化跨渠道风险关联</li> <li>3.新型攻击手段威胁：如AIGC滥用攻击者利用生成式AI快速伪造内容（如虚假活动页面），绕过传统检测模型；</li> <li>API滥用：未受速率限制的接口易被高频调用，导致服务器资源耗尽或数据泄露</li> </ol>

业务场景	业务形态	传统防护手段	传统防护机制现状	现存核心问题
表单字段限制	Web/APP/H5/小程序/公众号/微应用前端轻量化, 服务端全面API化成为技术主流	API网关产品	<ol style="list-style-type: none"> <li>动态路由与认证: 主流API网关通过JWT/OAuth2.0协议实现身份验证, 结合IP黑白名单拦截恶意访问, 可阻断90%以上的基础攻击(如SQL注入、XSS)</li> <li>敏感数据脱敏: 对API返回的身份证号、手机号等字段实施动态遮蔽, 降低数据泄露风险</li> <li>AI威胁检测: 集成机器学习模型识别异常流量(如高频请求、参数篡改)</li> <li>行为画像分析: 基于设备指纹、地理位置等200+维度建立用户行为为基线, 识别薅羊毛、撞库等业务欺诈行为</li> <li>支持《数据安全法》《个人信息保护法》要求, 提供标准化日志审计与数据分级分类功能, 满足金融、政务等行业合规需求</li> </ol>	<ol style="list-style-type: none"> <li>业务逻辑漏洞防护不足: 传统API网关难以识别伪造订单等业务层攻击, 需结合上下文风控系统联动防御</li> <li>新型攻击手段威胁加剧: 如AIGC滥用攻击: 利用生成式AI伪造API请求参数, 绕过传统规则引擎</li> <li>分布式爬虫突破: 黑产通过多端协作(如APP+小程序)分散请求, 导致单端限速策略失效</li> <li>高并发请求: 证券行业在交易高峰期可能会产生大量并发请求, API网关如未能有效地管理和分配资源, 可能会遭受拒绝服务攻击的情况, 从而影响正常服务。</li> </ol>
企业内部业务	企业内部提供至合作厂商或是客户接入的SDK、H5、API, 服务于业务数据流动, 业务服务提供	第三方接入时, 通过认证与鉴权机制达到安全接入管控, 对于数据全链路传输采用SM4/SM9算法对传输数据加密	<ol style="list-style-type: none"> <li>第三方接入时, 通过认证与鉴权机制达到安全接入管控,</li> <li>数据全链路传输采用SM4/SM9算法对传输数据加密</li> <li>对敏感字段(资金账户, 交易标的证券)实施动态脱敏, 仅授权接口返回完整信息</li> </ol>	此类服务具有明显的周期性, 无法实时对API访问进行持续监测, 仅能做到查缺补漏
第三方API服务	第三方提供SDK、H5、API服务为主要形式, 服务于业务数据流动和共享	第三方渗透测试服务、众测服	通常是以渗透测试、众测服务等形态对API接口进行攻击, 以期查找到未授权的API接口, 从而获取敏感数据。	此类服务具有明显的周期性, 无法实时对API访问进行持续监测, 仅能做到查缺补漏

#### 4.4 API安全技术解决方案

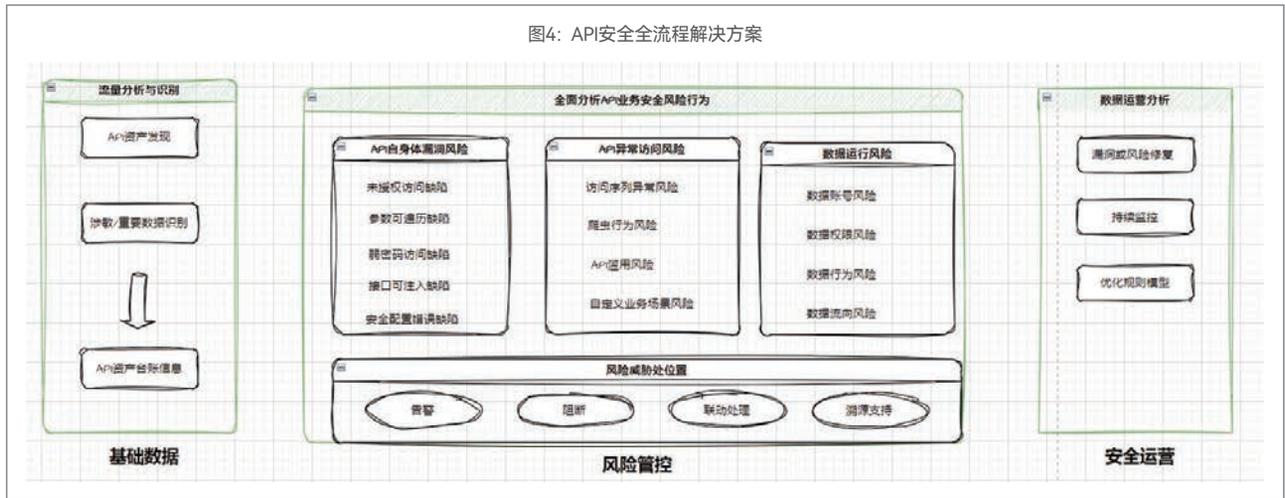
我司依照2023年5月颁布的《网络安全标准实践指南—网络数据安全风险评估实施指引》中有关数据安全技术情况里关于API接口安全部分的2个大要求以及11项详细要求:

制定API数据接口安全技术解决方案。整体解决方案涵盖动态梳理API及账号资产、持续检测系统API安全缺陷(未授权、越权等)、精准感知黑产攻击(扫号、撞库攻击等), 及时监测异常数据访问并进行审计溯源三个步骤:

数据接口类型	安全要求项	要求细则
对外接口安全	接口认证鉴权与安全监控	针对面向互联网及合作方数据接口的接口认证鉴权与安全监控能力的建设状况, 能否限制违规接入, 以及能否对接调用进行必要的自动监控和处理。
	API密钥安全管理	API密钥及密钥安全存储措施的设置情况, 能否避免密钥被恶意搜索或枚举。
	跨域安全流动控制	不同安全等级系统间、不同区域间跨系统、跨区域数据流动的安全控制措施情况。
接口安全控制	接口安全策略设置	接口安全控制策略的设置情况, 是否规定了使用数据接口的安全限制和安全控制措施, 明确包括接口名称、接口参数等内容的数据接口安全要求。
	调用审批管理	对于涉及个人信息和重要数据的传输接口, 是否实施调用审批。
	接口定期清查	是否定期对接口(特别是对外数据接口)进行清查, 清查不符合要求的接口是否立即关停。

数据接口类型	安全要求项	要求细则
接口安全控制	敏感接口调用管控	涉及敏感数据的接口调用是否具备安全通道、加密传输、时间戳等安全措施。
	接口防护措施	数据接口部署身份鉴别、访问控制、授权策略、接口签名、安全传输协议等防护措施的情况。
	接口安全要求	对接口类型、名称、参数等安全要求的规范情况。
	接口调用各方责任	与接口调用方是否明确了数据的使用目的、供应方式、保密约定及数据安全责任等情况。
	调用审批管理	对于涉及个人信息和重要数据的传输接口，是否实施调用审批。
	日志记录	是否对接口访问进行日志记录，同时对接口异常事件进行告警通知的情况。

图4: API安全全流程解决方案



第一步：动态梳理 API 及账号资产。

通过自动化手段梳理面向客户、内部员工、证券经纪人、客服、合作伙伴等场景的API，构建完整的API资产清单，实时掌握API开放数量、活跃状态、僵尸API、缺陷API、涉敏API以及API中流动的敏感数据等状况。依据请求和返回的内容进行模式匹配，自动发现业务端接口，并根据请求方法（Get/Post/Put...）、请求URL、请求参数对接口进行定义。并对接口进行聚类分析，形成API接口资产信息。

如：我司根据各个业务线、业务占比量以及数据敏感性，对业务场景API进行分类管理。例如实时交易、客户管理、数据服务等。每个业务线下的API可能具有不同的特点。比如，低延时交易API对性能的要求非常高，而开户和互联网业务API更注重用户

身份验证和数据隐私。数据平台API需要处理大量数据，并且涉及敏感信息的传输。针对上述业务场景，按照业务场景、业务占比和数据敏感性进行分类，并针对不同的业务场景及安全需求设计不同的防护策略。

图5: API资产动态梳理



图6: API分类归档

API类型	业务占比	数据敏感性	典型接口示例
极速交易类API	32%	超高(交易指令)	/order/market(市价单接口)
账户服务类API	18%	高(身份信息)	/account/open(在线开户接口)
行情数据类API	25%	中(市场数据)	/quote/level2(行情推送)
融资融券类API	12%	超高(杠杆操作)	/margin/purchase(信用交易接口)
资讯服务类API	8%	低(公开信息)	/news/announcement(公告推送)
机构对接类API	5%	超高(大宗交易)	/blocktrade/submit(大宗交易接口)

例如：对于极速交易类API，在业务投产时，对应创建极速交易API文档库，通过监控网络流量（如HTTP/HTTPS请求）识别API端点，基于极速交易API文档库所类型与监控到的网络流量业务日志形成对比对异常业务请求展开监控分析。同时，借助内部安全管理平台，在不同时段将分析结果推送至项目组相关负责人。

图7: API分类集中管理

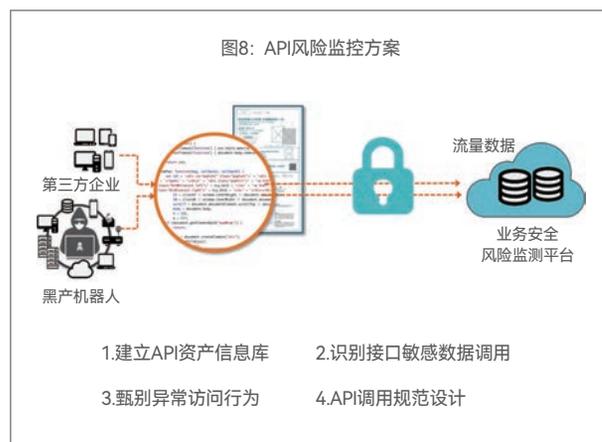


动态梳理API资产需要结合多种技术手段，包括资产发现、分类、监控、安全性检查、可视化和生命周期管理。通过自动化工具和平台，可以高效管理API资产，确保其安全性、可用性和合规性，同时为业务创新提供支持

第二步：持续检测系统API安全缺陷（未授权、越权等）。

在资产清晰可见的基础上，对API在设计和开发层面的缺陷展开持续、全方位的评估，探查API在认证、授权、数据暴露、输入检查、安全配置等方面是否存在可供攻击者利用的漏洞。

图8: API风险监控方案



依据业务访问链路，自动识别API接口的敏感数据，

技术手段	技术方法	工具推荐
API资产发现与识别	<ul style="list-style-type: none"> <li>流量分析：通过监控网络流量（如HTTP/HTTPS请求）识别API端点。</li> <li>日志分析：分析服务器日志（如Nginx、Apache）提取API调用记录。</li> <li>代码扫描：使用静态代码分析工具扫描代码库，识别API定义（如Swagger/OpenAPI文件）。</li> <li>端点爬虫：使用自动化工具模拟请求，探测API端点。</li> </ul>	<ul style="list-style-type: none"> <li>Burp Suite、Postman（用于端点探测）</li> <li>Swagger Inspector（用于API文档解析）</li> <li>Semgrep、SonarQube（用于代码扫描）</li> </ul>
API资产分类与元数据管理	<ul style="list-style-type: none"> <li>元数据提取：从API文档（如OpenAPI/Swagger）或代码注释中提取API的元数据（如端点、参数、返回值）。</li> <li>标签化分类：根据API的功能、业务领域、使用场景等对API进行分类和标签化。</li> <li>版本管理：识别API的不同版本，确保版本一致性。</li> </ul>	<ul style="list-style-type: none"> <li>Swagger UI、Redoc（用于API文档可视化）</li> </ul>
API资产监控与动态更新	<ul style="list-style-type: none"> <li>实时监控：通过API网关或代理监控API的调用情况，动态更新API资产清单。</li> <li>变更检测：使用差分工具对比API文档或代码的变化，识别新增、修改或废弃的API。</li> <li>自动化脚本：编写脚本定期扫描API资产，更新资产清单。</li> </ul>	<ul style="list-style-type: none"> <li>Prometheus、Grafana（用于API调用监控）</li> </ul>

图9: API访问链路敏感数据发现



**敏感数据发现**

平台能够支持自动化的识别API接口中的各种敏感数据,包括但不限于身份证号、手机号、用户地址、银行卡号等用户家庭信息、账户信息、身份信息等各类敏感信息,并为管理者生成敏感数据API接口的资产地图,让管理者及时掌握敏感数据风险态势。

**敏感数据泄露风险防御**

基于发现的敏感数据泄露接口反推开发对敏感数据的请求与响应进行完善,对于敏感数据的响应从业务侧进行梳理,避免返回非必要敏感数据信息。

图10: 自定义API业务访问场景

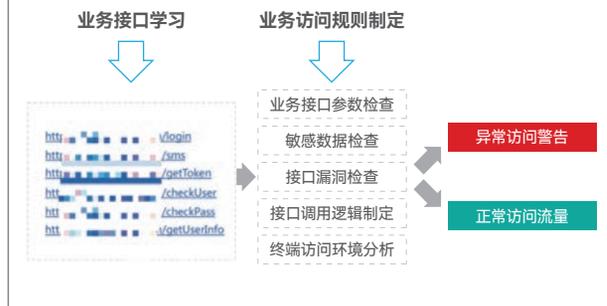


图11: 自定义API业务访问场景防护实例



根据不同的业务场景制定不同的业务访问规则,如登录场景、注册场景、实时交易规则验证等,使用自定义脚本借助交易日志提炼异常标识,及时获取业务请求流量,提炼异常标识,输出业务运行分析至相关系统负责人员,

技术手段	技术方法	工具推荐
自动化安全扫描	<ul style="list-style-type: none"> <li>漏洞扫描工具: 使用自动化工具扫描API的常见安全漏洞,如未授权访问、注入攻击、敏感数据泄露等</li> <li>动态分析: 通过模拟请求测试API的行为,检测是否存在安全缺陷</li> </ul>	<ul style="list-style-type: none"> <li>OWASP ZAP: 开源的Web应用安全扫描工具,支持API测试</li> <li>Burp Suite: 功能强大的渗透测试工具,支持API安全测试</li> <li>Postman + Newman: 通过编写测试脚本自动化检测API安全问题</li> </ul>
权限与访问控制测试	<ul style="list-style-type: none"> <li>未授权访问测试: 尝试在不提供身份验证凭证的情况下访问API,检测是否允许匿名访问</li> <li>越权测试:                             <ul style="list-style-type: none"> <li>水平越权: 使用低权限用户的Token尝试访问其他用户的资源</li> <li>垂直越权: 使用低权限用户的Token尝试访问高权限功能</li> </ul> </li> <li>角色验证: 验证API是否根据用户角色正确限制了访问权限</li> </ul>	<ul style="list-style-type: none"> <li>自定义脚本: 使用Python (如requests库) 或Shell脚本模拟不同权限用户的请求</li> <li>Postman: 通过设置不同的Token和参数测试权限控制</li> </ul>



技术手段	技术方法	工具推荐
异常行为检测	<ul style="list-style-type: none"> <li>行为基线建模：通过机器学习或统计分析建立用户正常行为基线（如登录时间、频率、IP地址等）</li> <li>异常检测算法：使用聚类、分类或时间序列分析算法（如孤立森林、LOF、ARIMA）检测偏离基线的异常行为</li> </ul>	<ul style="list-style-type: none"> <li>Splunk、ELK Stack：用于日志分析和异常检测</li> <li>机器学习框架：如Scikit-learn、TensorFlow</li> </ul>
实时监控与告警	<ul style="list-style-type: none"> <li>实时日志分析：通过流处理技术（如Kafka、Flink）实时分析日志数据</li> <li>告警规则：设置阈值和规则（如单IP高频登录、异常时间段访问）触发告警</li> </ul>	<ul style="list-style-type: none"> <li>Prometheus、Grafana：用于实时监控和可视化</li> <li>AWS CloudWatch、Datadog：云原生监控工具</li> </ul>
扫号与撞库攻击检测	<ul style="list-style-type: none"> <li>高频登录检测：监控同一账号或IP在短时间内的登录尝试</li> <li>密码错误率分析：检测同一账号或IP的高密码错误率</li> <li>账号枚举检测：识别攻击者尝试枚举有效账号的行为</li> </ul>	<ul style="list-style-type: none"> <li>WAF（Web应用防火墙）：如Cloudflare、AWS WAF</li> <li>自定义脚本：通过日志分析工具（如ELK）编写检测规则</li> </ul>
数据访问审计与溯源	<ul style="list-style-type: none"> <li>日志记录：详细记录用户访问行为（如登录时间、IP、操作类型）</li> <li>数据水印：在敏感数据中嵌入水印，便于溯源</li> <li>溯源分析：通过日志和数据分析工具追踪异常行为的源头</li> </ul>	<ul style="list-style-type: none"> <li>ELK Stack：用于日志存储和分析</li> <li>Splunk：用于审计和溯源</li> </ul>
响应与阻断	<ul style="list-style-type: none"> <li>自动阻断：对检测到的恶意IP或账号自动封禁</li> <li>人工响应：结合安全团队进行人工分析和响应</li> </ul>	<ul style="list-style-type: none"> <li>Fail2Ban：自动封禁恶意IP</li> <li>SIEM（安全信息与事件管理）：如Splunk、IBM QRadar</li> </ul>

## 4.5 API安全的主要风险及应对措施

### 4.5.1 常见的API安全风险

常见API安全风险主要有以下几种：

**数据泄露：**攻击者可能通过漏洞窃取API传输的敏感数据，如用户个人信息、业务数据等。

**身份验证失败：**如果API的身份验证机制存在缺陷，攻击者可能绕过身份验证，非法访问API资源。

**中间人攻击（MITM）：**当API使用未加密连接传输数据时，攻击者可能截获并篡改传输的数据。

**注入攻击：**恶意代码或数据注入到API请求中，可能导致系统被控制或数据被篡改。包括SQL注入攻击、跨站点脚本（XSS）攻击、XXE注入攻击等。

**DDoS攻击：**API容易受到分布式拒绝服务（DDoS）攻击，攻击者会向API发送大量请求，导致服务器崩溃，从而影响业务正常运行。

**API管理不善：**API管理不善也会给企业带来安全风险，比如影子API、废弃API、未经身份验证的API、未经授权的API等。

**影子API：**也称为未记录或未发现的API，它们是不受监督、被遗忘或不在安全团队可见范围内的

API，它可能导致合规违规和监管罚款，更有甚者，网络犯罪分子会滥用它来访问企业的敏感数据。

**废弃API：**是软件生命周期中的一个自然过程，如果废弃API未被删除，端点就会因为缺乏必要的补丁和软件更新而变得脆弱，从而导致被攻破的风险。

**未经身份验证的API：**的存在给企业带来了巨大的风险，因为它可能会将敏感数据或功能暴露给未经授权的用户，从而导致数据泄露或系统操纵。

**未经授权的API：**攻击者可以通过各种方法（例如枚举用户标识符）利用未经授权的API获得访问权限。

### 4.5.2 确保API安全的措施和方法

为了确保API安全，建议可以考虑以下措施和方法：

**定期更新和升级：**定期更新API以支持新的功能和安全性修复，确保API的安全性得到持续改进。

**加强身份验证：**对所有API请求进行身份验证，使用有效的凭据（如API密钥、令牌）进行访问。实施双因素身份验证或多因素身份验证，提高身份验证的安全性。

**防止数据泄露：**对API请求和响应中的敏感数

据进行加密，确保数据在传输过程中的安全性。使用HTTPS协议，确保数据在传输过程中不被窃听或篡改。

**限制访问权限：**通过授权机制限制对API的访问权限，确保只有经过授权的用户才能访问相应的资源。使用访问控制列表（ACL）进一步细化访问权限管理。

**监控和日志记录：**记录API请求和响应的日志，

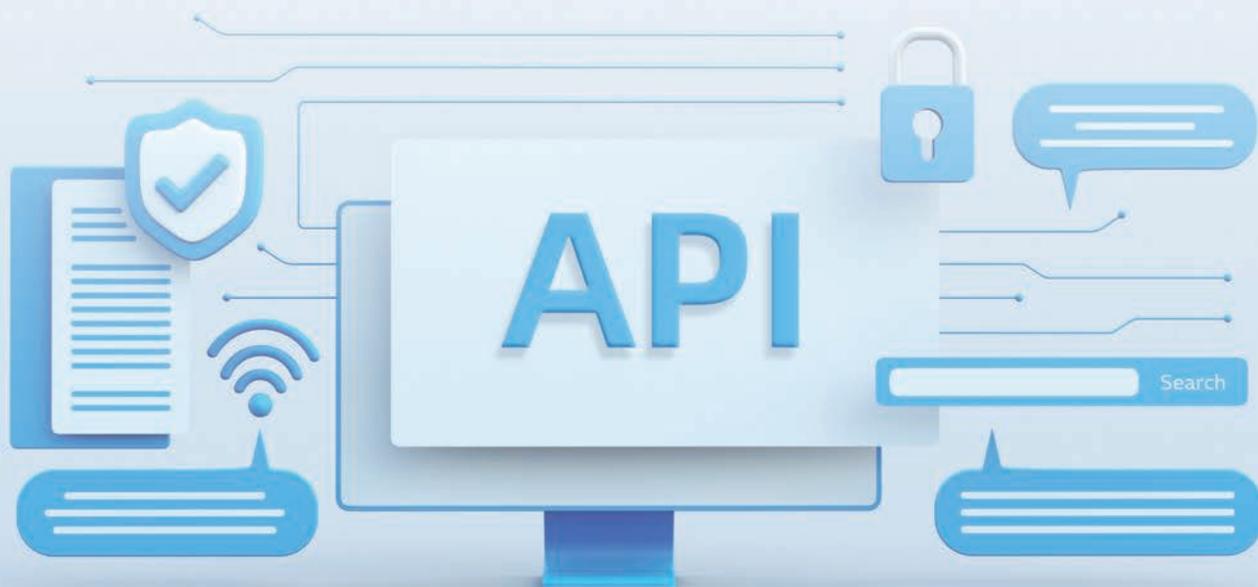
以便在发生安全事件时进行追溯和分析。识别未监控或未经身份验证的API端点，降低因API管理不善带来的各种安全风险；同时记录每个应用程序接口请求，跟踪用户活动，防止数据泄露或违规问题。

**使用安全工具：**利用自动化API安全工具进行安全测试和漏洞扫描，及时发现并修复潜在的安全问题。🔒

责任编辑：胡清华

#### 参考资料

- [1]JR/T 0248-2022 证券期货业机构内部接口 证券交易
- [2]JR/T 0275-2023 证券期货业机构内部接口 资讯数据
- [3]JR/T 0233-2021 证券期货业经营机构内部应用系统日志规范
- [4]JR/T 0159-2018 证券期货业机构内部企业服务总线实施规范
- [5]JR/T 0155.3-2018 证券期货业场外市场交易系统接口第3部分:结算接口
- [6]JR/T 0155.1-2018 证券期货业场外市场交易系统接口第1部分:行情接口
- [7]JR/T 0155.2-2018 证券期货业场外市场交易系统接口第2部分:订单接口
- [8]TC260-PG-2023XX 网络安全标准实践指南—网络数据安全风险评估实施指引
- [9]API Security. (n.d.). Retrieved from [https://en.wikipedia.org/wiki/API\\_security](https://en.wikipedia.org/wiki/API_security)
- [10](JR/T 0185—2020)《商业银行应用程序接口安全管理规范》
- [11](JR/T 0171—2020)《个人金融信息保护技术规范》



# 基于微服务+DDD的期货公司应用系统构建方案

王莎莎、张思运、王鑫、李涛 / 国联期货股份有限公司

在金融科技快速发展的背景下,期货公司的数字化转型不仅是业务战略的关键组成部分,也是提高市场竞争力的必要手段。本文结合金融科技、公司价值流管理理念及业务建模技术,采用微服务架构与领域模型设计(DDD),深入探讨了基于微服务架构与领域驱动设计(DDD)相结合的期货公司应用系统构建方案,构建了一个高度模块化、可扩展且带有业务导向的期货公司应用系统。本方案旨在有效解决传统架构,在应对期货市场复杂业务需求以及大规模分布式系统挑战时的不足的同时,有效的提高系统的灵活性、可维护性与可扩展性,以便更好地适应业务的快速变化与创新,为期货公司的持续创新和快速发展提供坚实的技术支撑。

## 一、引言

随着当今金融科技的快速发展,期货行业则面临着数字化转型的重要机遇和挑战。金融科技不仅改变了期货市场交易和风险管理模式,还推动了业务创新、客户服务以及监管合规等方面的变革。随着AI技术的全面普及的背景下,期货公司急需通过数字化转型构建一个高效、灵活且可扩展的应用系统,以适应整个市场的巨大变革。业务建模作为数字化转型的核心基础,它主要是通过优化业务流程、数据流和信息系统,以供金融科技的创新和数字化转型指引出一条清晰的快速发展壮大的路径。

本文旨在构建基于微服务架构与领域驱动设计(DDD)的期货公司应用系统的构建方案。微服务架构以其模块化、可扩展性和灵活性,成为了构建复杂系统的理想选择,有助于提高系统的可维护性和可靠性,促进团队协作效率与创新能力。而DDD则强调以业务领域知识为核心,构建丰富的领域模型来指导系统设计和开发,确保业务逻辑清晰、易于理解,并促进开发团队与业务领域专家的有效可靠沟通。通过结合金融科技的优势和数字化转型的需求,该方案为期货公司提供了一种高效、灵活且能够紧密响应市场变化的一套应用系统构建方案,助力其在金融科技

的浪潮中保持强有力的竞争优势,以期实现持续的经济业务增长和金融科技的创新发展。

## 二、应用系统实现的理论基础与技术背景

### (一) 微服务架构核心原理与优势

微服务架构的核心原理在于将复杂的应用程序拆分为一组小型、自治的微服务单元,每一个微服务单元又具有独立性、可替换性、自治性、单一责任原则、松耦合和高内聚等特点。这些微服务单元可以根据具体的业务功能进行拆分,并能独立部署和运行,且不受其他服务的影响。这种架构相比其他传统的单体架构更具备诸多优势,如易于扩展和升级、技术栈灵活、易于维护和升级、高可用性、容错性强、便于维护、支持持续集成和持续部署,以及快速响应市场需求等特点。

综上所述,微服务架构通过其核心原理和诸多优势,必然会成为构建大型、复杂系统的理想选择。因为它不仅能够提高系统的可扩展性、灵活性和稳定性,还能够缩短产品迭代周期,提高交付效率,降低发布风险,所以随着技术的不断发展和市场的不断变化,微服务架构必将继续在构建应用系统方面发挥至关重要的作用,可以为业务提供稳定、可靠、灵活的技术支持。

## （二）领域驱动设计（DDD）的核心概念与实践

本方案通过深入理解业务领域，建立领域模型，并合理划分微服务边界，这样期货公司就可以构建出符合业务需求、易于维护和扩展的业务系统。同时，DDD的实践方法也为该系统开发提供了清晰的指导和支持。

### 1. 领域驱动设计（DDD）的核心概念

本方案中的领域驱动设计（DDD）与微服务架构的巧妙融合，为期货公司应用系统构建带来了高效、灵活且可扩展的解决方案。在本方案中，DDD的核心概念包括领域模型、统一语言、实体、值对象、聚合、领域服务和仓储，这些概念共同构成了对业务领域的深入理解和抽象。通过DDD的实践方法，如战略设计、战术设计、事件风暴、微服务划分以及持续演进等设计，期货公司就可以构建出符合业务需求、易于维护和扩展的业务功能系统。

### 2. 领域驱动设计（DDD）的实践

战略设计，从业务视角出发，建立业务领域模型，划分领域边界，并确定业务和应用边界，保证业务模型与代码模型的一致性。战术设计，则侧重于领域模型的技术实现，完成软件开发和落地。事件风暴，则是建立领域模型的重要方法，有助于发现业务领域中的关键概念和关系。微服务划分，就是根据业

务领域的不同，将系统拆分为多个微服务，实现业务关注度的最大化，并通过轻量级的通信机制相互协作。持续演进，则强调根据业务发展，合理划分领域边界，不断优化现有架构和代码，保持架构和代码的生命力。

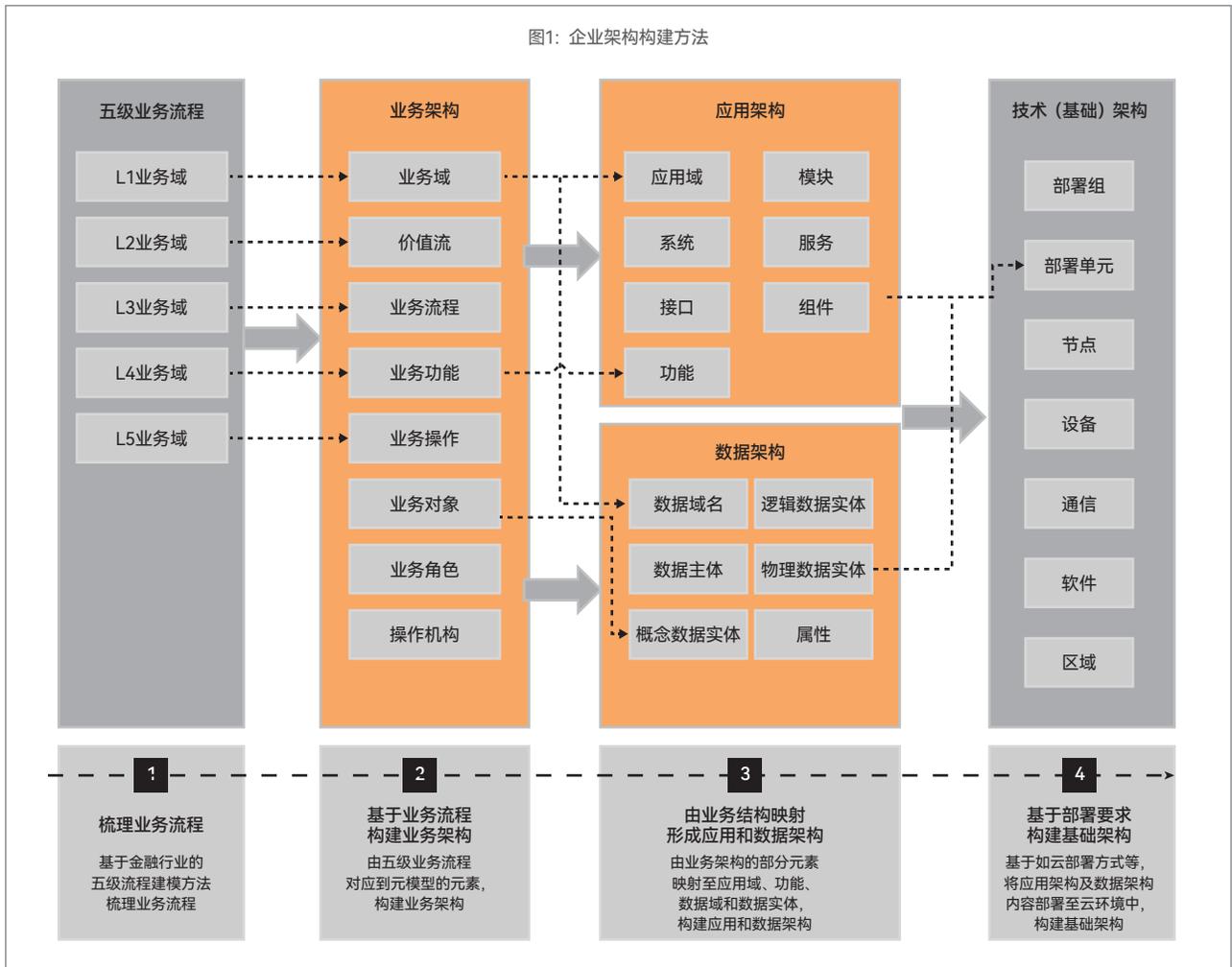
## 三、公司价值流驱动实现应用系统的业务模型落地

### （一）公司价值流理论及其在业务优化中的应用

价值流在期货公司业务中是至关重要的理论，它要求公司识别并优化从客户开户到交易执行、风险管理及客户服务等业务流程中的增值环节，如高效的交易执行和精准的风险管理，同时减少非增值活动，如冗长等待和重复工作，以提升运营效率和客户满意度。本方案通过引入自动化工具和技术，优化服务流程，并建立持续监控与改进机制，这样期货公司才能够确保业务流程随市场需求变化而不断优化，减少资源浪费，提升公司的整体竞争力。

### （二）价值流驱动的业务模型构建

在快速变化的金融市场中，期货公司借助价值流驱动的业务模型构建，能够有效识别并优化业务流程中的增值环节，减少非增值活动，提升运营效率与竞争力。而价值流驱动的企业架构构建方法则是一个复杂且系统的过程，它涉及多层面、多维度的综合设计和业务规划。价值流驱动企业架构构建方法如图1。



## 四、业务模型引领任务中心应用系统技术模型落地案例

### （一）面向金融研发场景的端到端架构实践

#### 1. 案例背景与核心需求

##### 1) 业务背景

本案例聚焦于金融科技公司在研发效能方面所面临的众多挑战性问题，如计划调整低效、任务分配失衡、流程标准化不足及知识资产流失等等。也是为应对金融市场快速变化带来的业务流程复杂性和效率瓶颈等棘手问题，所以公司引入微服务架构与领域驱动设计（DDD）理念，对核心任务中心进行全面升

级。通过重塑价值流，实现业务流程的数字化与自动化，提升任务分配准确性、优化业务建模和领域划分，旨在提高员工解决问题的高可靠性的工作效率，这样才能为长期发展奠定基础。这一举措不仅可以解决当前运营问题，还能增强公司的市场竞争力，确保公司在行业中处在领先地位。

##### 2) 构建核心需求矩阵

#### 2. 五级流程建模与业务架构构建

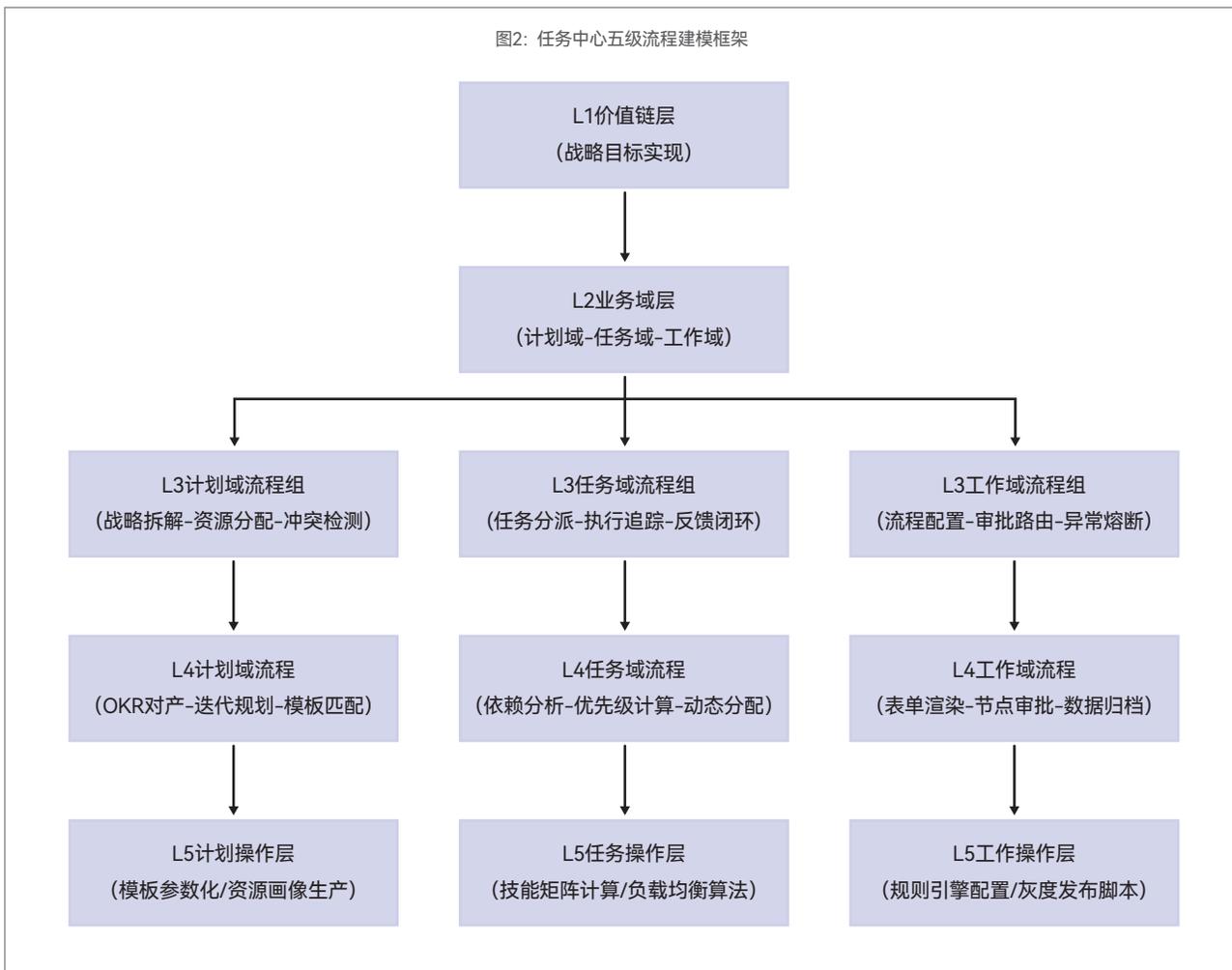
##### 1) 五级流程建模框架

该领域模型映射和流程优化框架，包括了通过分层结构将计划域、任务域和工作域的要害进行清晰划

表1: 任务中心核心需求矩阵

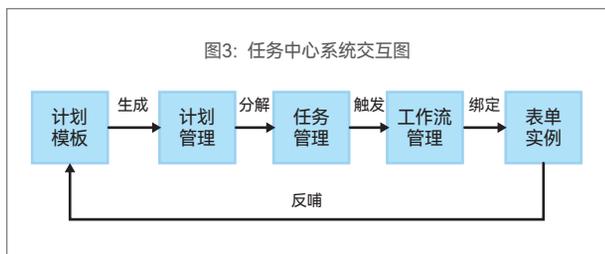
业务属性	功能需求	非功能要求
计划模板	支持200+场景模板的智能匹配	模板检索响应时间≤1.5秒
计划管理	实时检测资源冲突并自动优化调度	调度计算延迟≤300ms
任务管理	任务自动分解与优先级动态调整	任务拆分粒度≤2人日
workflow 管理	可视化拖拽式流程设计	流程变更生效时间≤15分钟
表单复用	跨场景表单组件复用率≥75%	上下文渲染准确率≥95%

图2: 任务中心五级流程建模框架



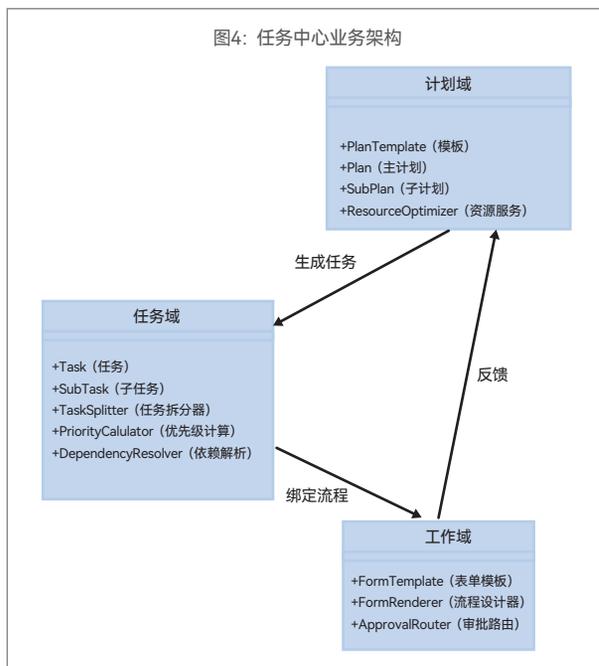
分,从战略目标聚合根到具体的操作实体,也实现了从战略到执行的全链路覆盖。在流程优化关键措施中,本系统通过在L5操作层实现多项自动化操作(如计划模板推荐、任务负载计算等),并结合规则引擎触发流程分支(如资源超限时启动应急审批流),提升了流程效率。于此同时,本系统还通过设立质量控制门禁,这样就在计划完整性、任务可执行性和流程合规性等关键检查点设置了严格的质量标准与控制方式,确保了流程的规范性与高效性。这一框架通过领域模型映射与流程优化措施的有机结合,为复杂业务流程的精细化管理和持续改进提供了高可靠的系统性支持。

## 2) 任务中心系统交互图



该系统的交互特征主要体现在三个方面:首先,采用模板驱动的方式,建立了多个场景模板,支持语义检索与参数化配置,提升了流程设计的灵活性和复用性;其次,该系统通过动态编排功能,能够根据任务特征自动匹配最佳工作流路径,确保流程的高效执行;最后,该系统也实现了知识沉淀,通过将表单使用数据回流至模板库,形成闭环优化机制,持续提升模板的实用性和适配性。这些特征共同支撑了本系统的高可靠性的运行与自身业务的可持续性改进。

本业务架构通过三大核心域构建,形成了从战略到执行的全链路闭环。计划域以Plan聚合根为核心,实现战略目标到执行计划的高效转化,并支持模板遗传机制,使子计划能够继承主计划80%以上的特征,确保计划的一致性与连续性。任务域通过多级拆分与动态调度机制,支持任务的灵活分解与高效执行,提



升了任务管理的精细化水平。工作域通过标准化表单设计与组件复用,也显著降低了开发成本,并提高了流程的规范性和一致性。三大核心域相互协同,共同支撑业务流程的高效运转与持续优化。

## 3. 应用与数据架构映射

### 1) 微服务拆分及分层方案

如图5所示,该设计过程包含了三个关键步骤:识别具有业务价值的独立领域场景、根据职责层次对场景进行任务分解、以及为领域驱动设计角色构造型分配职责。这一过程是事件风暴的结果,本方案通过事件风暴获得了包括决策命令、读模型、聚合和事件在内的领域分析模型。其中,任务分解作为核心环节,有助于明确问题范围、目标及限制条件,从而确保设计流程更加合理与顺畅。这一方法为后续的微服务架构设计奠定了坚实基础。

根据图5的业务场景驱动领域设计方法,得到了如下所示的微服务架构图6及微服务分层架构图7。基于公司业务价值流的划分,任务中心的场景业务分析涵盖了计划管理、任务管理、工作流管理等多个子

图5: 业务场景驱动领域设计过程图

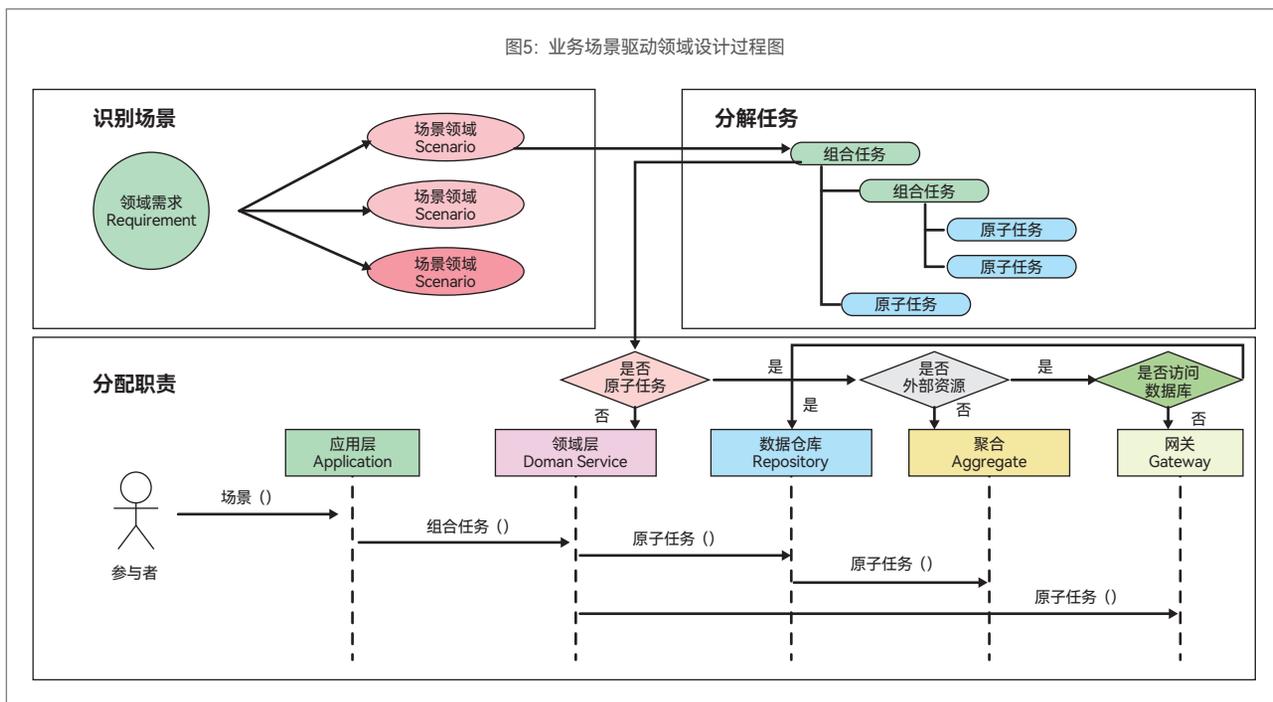
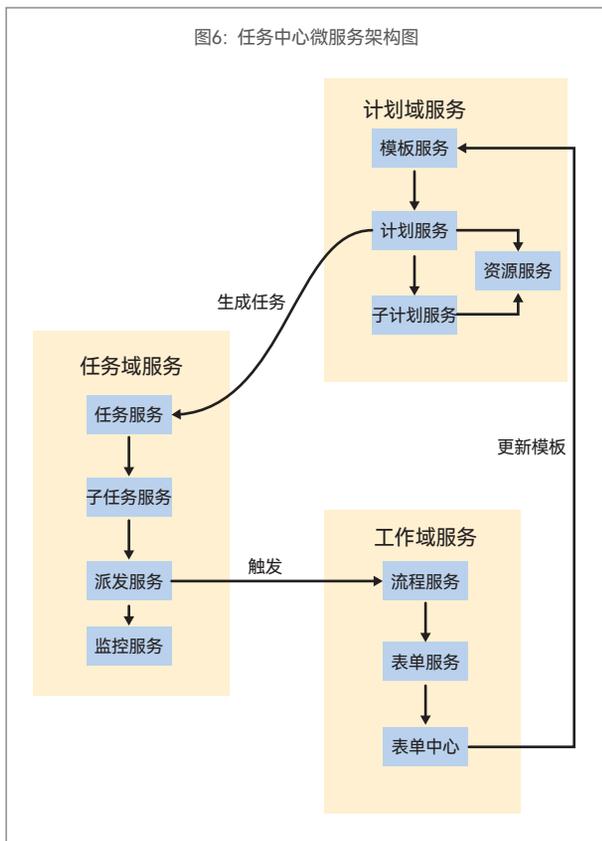


图6: 任务中心微服务架构图

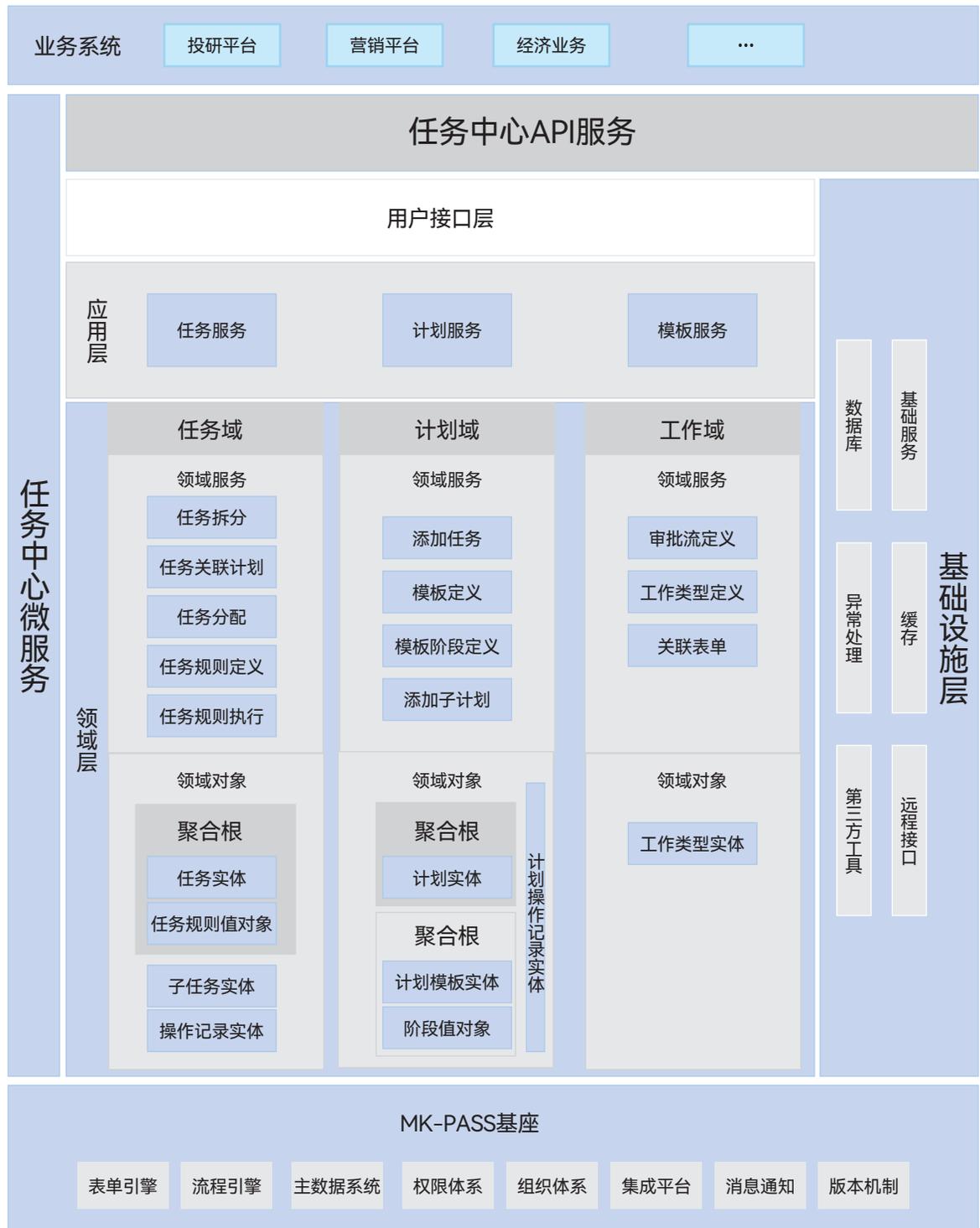


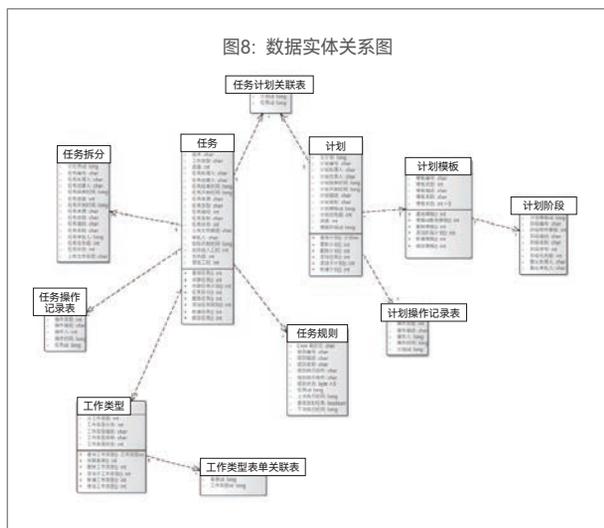
域。其中，计划域、任务域和工作域为核心域，本方案通过任务分解进一步明确了各子域的具体职责与流程。这一分解过程清晰地界定了各领域的边界与协作关系，为微服务架构的实现提供了清晰的指导，确保了系统的高内聚与低耦合特性。

## 2) 数据实体关系图

如图8所示，数据架构通过三层治理机制实现了数据的全生命周期管理。首先，通过血缘追踪功能，本系统能够完整追溯任务的生成路径（从计划到子计划，再到任务与子任务），确保数据来源的透明性与可追溯性；其次，本方案的版本控制机制支持模板与计划的历史版本回溯，为数据变更提供了可靠的审计与恢复能力；最后，本方案采用加密算法对敏感表单数据进行加密存储，确保了数据的安全性与合规性。数据实体关系图清晰地展现了计划、任务、表单等核心实体之间的关联关系，为数据的高效管理与应用提供了结构化支持。这一架构通过血缘追踪、版本控制与安全加密的有机结合，实现了数据的精细化治理与全面保护。

图7: 任务中心微服务分层架构





### 3) 应用系统架构图

微服务+DDD领域模型应用系统的实现方案，是通过构建包含系统层、应用层、表单层、接口层和微服务层的多层次架构共同组成的。其中，应用层，

通过集成中心整合任务中心、客户中心等多个业务领域；微服务层，则按照DDD原则划分为任务域、客户域、组织权限域、产品域和服务域等；而各微服务则聚焦于特定业务领域，通过内部系统接口和开放平台接口与外部系统交互，同时利用数据库层存储各领域的业务数据，以实现高内聚、低耦合的服务架构。所以，微服务+DDD领域模型应用系统的实现方案如图9所示：

### 4. 云原生基础架构设计

如图10所示，该模型的核心特征体现在智能服务治理体系的构建与优化等方面。首先，则是通过基于Nacos集群的动态服务网格，构建了三级服务目录，支持秒级服务注册与发现，并结合元数据标签实现金丝雀发布流量染色，利用权重响应式算法实现灰度发布的零中断。其次，再跟进采用GitOps模式管理多环境配置，支持配置版本追溯、敏感数据国密加密及

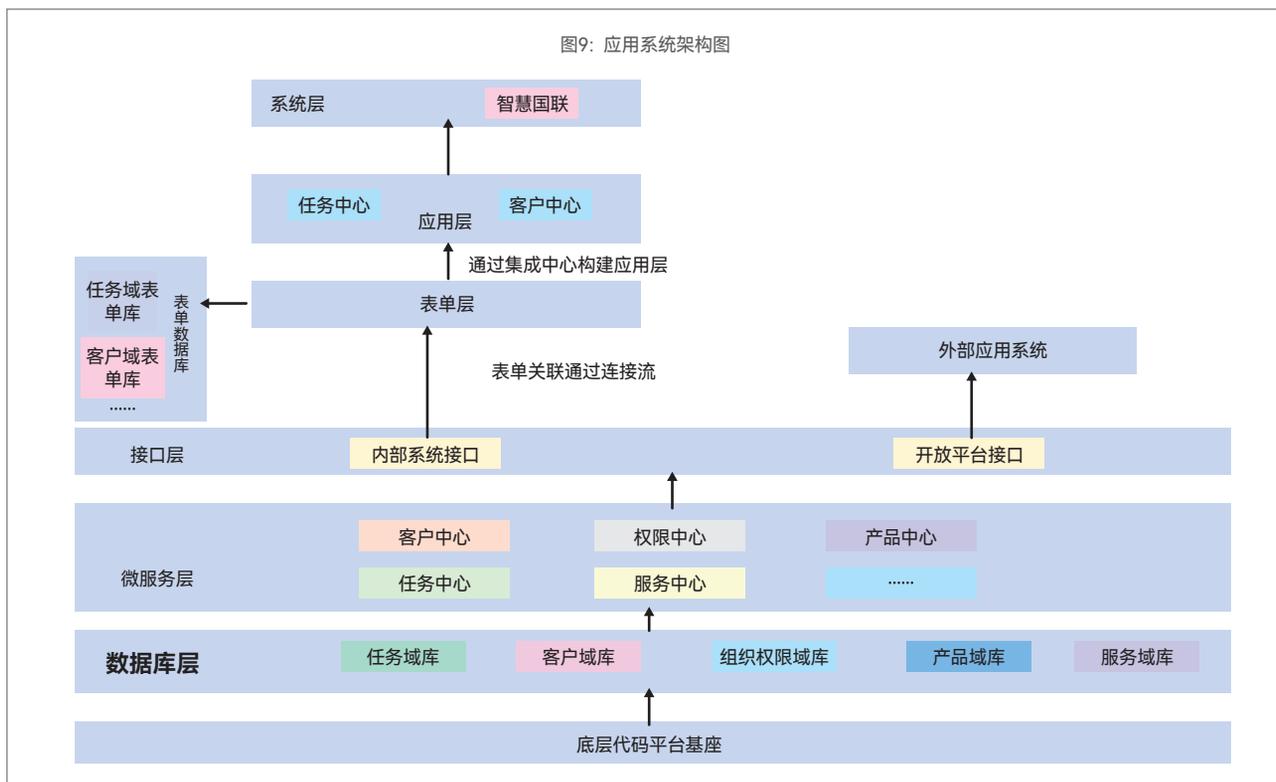
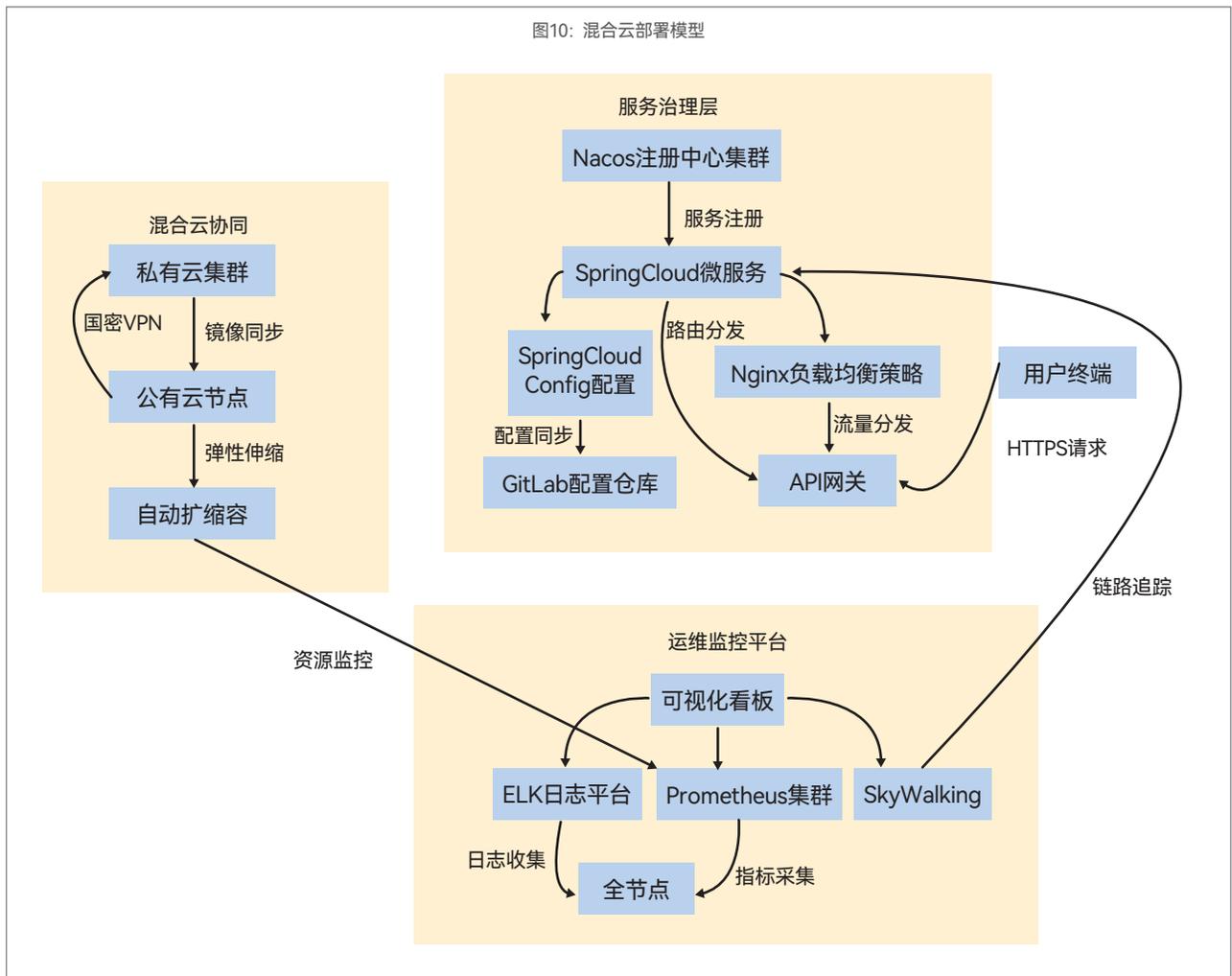


图10: 混合云部署模型



配置热加载，确保配置管理的安全性与高效性。最后，则是通过API网关内置的智能路由引擎，依据实时监控数据自动决策流量分发策略，支持私有云稳态业务的SLA保障、公有云弹性资源池的突发流量承接以及跨云故障自动转移，实现了跨云流量的智能调度与高可用性。综上，这一架构通过动态服务网格、配置驱动架构与跨云流量调度的有机结合，全面提升了系统的智能化治理能力与业务连续性保障水平。

## （二）系统构建过程与关键技术挑战

在该系统构建的核心模块中，我们进行了深入业务需求的分析，并依据领域驱动设计（DDD）原则，构

建了跨越计划域、任务域、工作域三域协同的领域模型，并明确了业务实体及其关系。随后，我们采用五级流程建模策略层次化系统逻辑，并运用微服务架构将业务能力细分为若干个核心服务单元，为每个服务均基于特定需求选用了最适宜的技术栈开发模型。在此过程中，我们面临跨域流程协同、数据一致性保障、云原生服务治理与流量管理、服务合理拆分与重构、系统安全性与合规性等众多的挑战。为克服这些挑战和问题，我们采纳了较为前沿的架构设计思路，结合微服务与DDD的优势，构建并实现了一个稳定、高效、可扩展且易于维护的业务应用系统方案。

## 五、结论和展望

本方案详细叙述了一个，基于微服务架构与领域驱动设计（DDD）的期货公司业务应用系统构建方案。我们通过业务需求分析，明确了该系统的需求和功能，采用了DDD来构建领域模型，这样就为微服务划分提供了理论支持。实践中，本方案将复杂业务拆分为独立且协作的微服务，确保了该系统的松耦合和高内聚等关键问题，同时引入其他关键技术来提高系统的高可靠性。本方案克服了如服务治理、数据一致性等挑战，将微服务架构与DDD的巧妙融合，为该系

统带来了灵活性和可扩展性，所以这样就提高了我们的业务响应速度和市场竞争力。

展望未来，期货公司应用系统构建方案将会不断的演进和日趋完善。我们也将持续关注和改进利用微服务架构、DDD及AI等前沿技术，探索其在期货公司应用系统中新的应用和新的模式。于此同时，我们也计划加强行业内交流与合作，共同推动期货公司应用系统的创新与发展，为期货市场的繁荣和稳定继续贡献力量。🌀

责任编辑：孙秋景

### 参考文献

- [01] 证券期货业数字化转型工作组. 证券期货业数字化能力成熟度模型[S]. 2023.
- [02] 证券期货业标准化技术委员会. 证券期货业数据模型[S]. 2022.
- [03] 普华永道企业架构团队. 金融行业企业架构管理白皮书[R]. 北京: 中国金融出版社, 2022.
- [04] 证券行业金融科技系统高效能交付体系构建研究[C]//中国证券业高质量发展论文集(2023). 国泰君安证券股份有限公司, 2023:7. DOI:10.26914/c.cnkihy.2023.107507.
- [05] 朱璧和. 商业银行数字化转型研究[D]. 东南大学, 2022. DOI:10.27014/d.cnki.gdnau.2022.004202.
- [06] 汤志江. 期货公司数字化转型路径研究[D]. 河北地质大学, 2022. DOI:10.27752/d.cnki.gsjzj.2022.000290.
- [07] 蒋东兴. 关于证券期货业数字化转型的思考[J]. 清华金融评论, 2021,(09):89-93. DOI:10.19409/j.cnki.thf-review.2021.09.022.
- [08] 曹伶俐. 基于领域和节点属性网络的微服务拆分研究[D]. 安徽大学, 2022. DOI:10.26917/d.cnki.ganhu.2022.001711.
- [09] 李逸阳. 面向机构的期货开户与服务系统的设计与实现[D]. 南京大学, 2021. DOI:10.27235/d.cnki.gnju.2021.001475.

# Domain-Driven Design

## 数字赋能仓储服务 飞泰护航系统升级

大商所数字仓库系统作为服务市场主体的关键窗口，紧扣市场需求，持续拓展业务领域，将仓库、品牌、质检机构等主体创建流程全面线上化，并打通OA办公系统与天眼查等第三方数据平台，实现数据互联互通，显著提升客户业务办理效率。

飞泰公司作为大商所测试保障团队，多维度保障系统升级质量。依托深厚的仓单业务积累，全面梳理测试范围，确保需求覆盖无死角；运用程序特征提取技术识别代码分支，将测试需求结构化，筑牢系统稳

定性根基；搭建与客户的沟通桥梁，从用户视角挖掘使用诉求，完善测试场景；引入接口自动化测试技术，强化API接口验证，提升系统交互的可靠性；协助400余家外部客户参与测试体验，收集反馈意见，持续提升客户使用满意度。

在飞泰公司的全力护航下，数字仓库系统已平稳上线，为实体经济发展注入强劲动力。未来，大商所飞泰公司将持续深耕测试技术创新，为“数字大商所”建设与实体经济服务贡献更多力量。

## 飞泰公司率先推进测试大模型构建，推动智能化测试实现新突破

依托在测试领域多年的技术积累，飞泰公司积极引入大语言模型与智能体技术，推动测试智能化转型。当前已启动测试大模型的技术验证工作，重点覆盖需求整理、需求分析及测试设计三个关键环节，聚

焦核心交易类业务场景下的测试辅助能力。下一步将携手行业头部机构共建高质量语料库，持续推进测试大模型建设，提升测试自动化与智能化水平。

## 全链路通关自动化测试解决方案

全链路通关自动化测试解决方案是针对金融行业周末通关测试周期短、业务场景复杂、环境真实性要求高等核心痛点设计的创新方案。飞泰公司依托分层架构设计与动态数据策略，结合自动化测试平台ATF的高并发执行能力，在2小时内完成了核心业务及

新增关联业务的全面验证。该方案包括柜台接口自动化、PC自动化和APP自动化，旨在打造“数字员工”智能测试体系，通过接口-PC-APP三位一体的全链路自动化方案，实现无人化通关测试，进一步扩大通关测试覆盖维度和准确性。

# 衍界

测试  
专刊

证券期货业 | 测试技术与质量管理

Securities & Futures Industry Testing Technology and Quality Management

## 征稿启事

本刊旨在提供最新鲜的行业热点、最及时的质量动态、最先进的行业测试技术、最宝贵的质量管理专家经验,让更多的人了解行业动态、学习专业测试技术、探讨质量管理改善,从而促进行业测试技术发展与行业质量建设。

也许您的身边还有很多值得分享的题材,也许您的见解能够给大家带来更多的启发和借鉴,欢迎各界人士积极供稿,这里需要您的参与!

✉ 投稿邮箱: [quality@dce.com.cn](mailto:quality@dce.com.cn)

☎ 联系电话: 0411-8480 6897

# 衍界

证券期货业 | 测试技术与质量管理  
Securities & Futures Industry Testing Technology and Quality Management

[www.dce.com.cn](http://www.dce.com.cn)

地址：中国 辽宁省大连市沙河口区会展路129号 邮编：116023

电话：(86) 400 861 8888 传真：(86) 411-8480 8588