目 录

# 系统总体设计+应用模式+使用要求---招标评议响应情况对照表

**“软件测试工具”研制招标评议项目表-系统总体设计部分+应用模式+使用要求**

| 类别 | 序号 | 项目 | 要求 | 分值 | 响应情况 | 对应章节 | 页码 |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 技术部分 | 1 | 系统总体设计 | 系统总体架构设计合理，采用“高性能计算资源+成熟云基础平台+灵活云测试服务+高质量软件测试工具”的技术架构进行技术系统设计；能够满足非嵌入式软件测评需求、指挥信息系统软件全系统测试需求、典型FPGA软件的混合仿真验证需求、软件资产管理需求。 | 4 | **完全响应**  对系统进行了总体架构设计，按基础硬件设施层、IaaS层、PaaS层、SaaS层、云桌面端机应用层分层的技术架构，软件测试工具涵盖了嵌入式软件、非嵌入式软件测试工具；能够满足非嵌入式软件测评需求、指挥信息系统软件全系统测试需求、典型FPGA软件的混合仿真验证需求、软件资产管理需求。 | 3.2.1  5.2 |  |
| 系统工作流程、应用模式描述清晰,满足软件试验和软件测评等多方面应用需求。 | 1 | **完全响应**  对系统使用的业务流程、系统的工作流程、应用模式均进行了有效的阐述和设计，可以完全满足软件试验和软件测评等多方面应用需求。 | 5.5  5.6  5.7 |  |
| 系统实现关键技术解决或验证充分，解决措施合理可行。 | 1 | **完全响应**  共分析出6大项系统实现的关键技术，对所有的关键技术均进行了技术攻关，给出了技术的解决方案。 | 6 |  |
| 具有云测试平台、软件测试工具的研发经验和技术积累；具有相关发明专利或软件著作权；具有软件评测经验和软件实验室建设经验。 | 2 | **完全响应**  XX | 7 |  |
| 软件测试工具优先选用成熟的、行业认同度高的主流软件测试工具；单机版测试工具的 License不少于2个；除FPGA混合仿真验证工具外,其他软件测试工具均能够与软件测试云平台集成,测试数据能与平台共享。 | 3 | **完全响应**  XX | 8  9 |  |
| 对集成已有的软件测试工具,产品选型合理,测试工具 License数量、功能与技术指标的符合性论述清晰；对需研发的软件测试工具,需求理解准确、功能设计合理,测试工具功能设计与技术指标的符合性论证清晰。 | 3 |  |  |  |
|  | 7 | 应用模式 | 对系统建设需求理解清晰,云测试平台应用模式论证充分、应用模式设计合理、技术可行；软件测试流程、信息交互关系阐述清晰；应用模式涵盖软件试验和软件测评两种应用。 | 2 |  |  |  |
|  | 8 | 使用要求 | 使用要求环境适应性、可靠性、维修性、保障性分析论证清楚，指标合理，指施落实得当。 | 2 |  |  |  |

# 项目背景分析

## 项目名称

项目名称：软件测试工具研制

## 业主方面临的任务分析

随着装备信息化建设任务的要求和技术的发展，军用电子信息系统装备和系统的呈现出体系化、服务化、快速迭代等技术特点，按照装备鉴定试验的新要求，对装备性能试验、作战试验、以及在役考核等提出了新的更高的要求，中国人民解放军63880部队承担着全军电子信息装备和系统软件试验和软件测评的重要任务，需要一套具有试验和测试环境快速设计、快速部署、试验和测试任务快速规划设计、试验和测试任务自动/半自动开展、试验和测试过程控制执行和分析评估的云平台系统，支持所承担的任务的高效、高质量完成，以更好的保证电子信息装备软件和系统的质量。

## 电子信息装备的发展趋势

当前，军用电子信息装备的发展趋势进呈现为体系化、国产化、服务化、系统虚拟化、大型分布化、云化等特点，研制和建设呈现边建设、边试用、边试验/测试的特点，传统的瀑布模型使用的越来越少，试验和测试更早期的参与到系统和装备的研制过程中，试验任务频次较之前的模式也越来越多。

从电子信息装备系统的组成看，主要包括3类：一是非嵌入式系统/软件，主要完成系统的指挥控制、态势处理、情报融合处理等能力；二是嵌入式系统/设备主要完成系统所需信息的采集、传输和处理；三是前板卡级的设备，主要完成信息的采集和收集，主要功能大多由FPGA软件来实现。

这些都需要试验单位能够快速响应，快速部署、快速组织实施、快速给出评价结论，同时试验和测试环境能够满足未来装备发展的需要，具有快速升级和扩展的能力。

## 现有的试验与软件测评基础

经对甲方现有系统的调研，甲方前期通过上级配发和自研项目支持两个途径，初步形成了C/C++语言的软件静态代码分析和单元测试能力、设备级嵌入式软件接口测试能力，具备了一定的软件试验和测评能力，主要包括：

（1）LDRA Testbed静态单元测试工具（C/C++语言）

LDRA Testbed是英国利物浦数据研究协会所研制的静态单元测试工具，该工具提供强大的包括编码规则检查在内的静态分析功能和全面的静态分析报告，用户可以选择行业认可的标准，如MISRA C、DERA、AV C++ 等。提供软件质量的度量，以可视化的方法了解代码的复杂性，提供详细的函数调用关系和程序控制流程图，自动生成报告，提供软件质量文档。提供了接口分析、识别潜在的错误、程序变量，对所有过程参数以及函数的全局变量和返回值进行分析。

LDRA软件测试套件TBRun单元测试功能可辅助开发工程师或测试工程师自动生成测试向量、测试驱动及桩模块以实现单元测试与回归测试，通过代码覆盖率分析辅助可实现在进入系统集成和系统测试之前确保完全的代码测试。通过LDRA单元测试功能可在软件生命周期早期揭示代码缺陷，从而减少代码缺陷遗留到开发后期所引起的更多费用。

（2）设备级嵌入式软件接口测试工具ETest

设备级嵌入式软件接口测试工具ETest是国内厂商-凯云联创（北京）科技有限公司开发的，具有自主产权的嵌入式系统实时半实物仿真测试工具，主要用于建立设备级电子信息装备的嵌入式软件测试环境。

ETest具有测试设计、测试运行、测试监控与数据分析等功能，提供了嵌入式设备常见的对外交联环境接口，主要包括RS232、RS422/RS485、以太网、A/D、D/A、DI/DO、继电器接口、CAN总线接口等。设备级嵌入式软件测试平台通过这些接口与待测系统相连，实现对设备级电子信息装备系统运行外部环境的仿真模拟，确保设备级电子信息装备能在各种环境下完成试验与测试任务。

（3）测试管理工具

甲方利用对软件测试质量管理体系的建立过程以及自身申报科研项目的方式，通过使用现有的开源的测试管理工具及免费使用的商用测试管理系统，积累了软件测试管理系统的开发经验，主要包括BestCode（国内厂商研发）、STM（国内厂商研发）等。根据这些测试管理系统，甲方利用自身的研发能力，对软件测试管理系统进行了原型系统的开发，主要完成了测试需求分析与策划、测试用例设计、测试执行管理和测试总结等功能，具备基本的对软件测试的管理

## 差距与建设需求

根据对甲方现有装备试验和软件测评的条件分析，这些现有条件与当前形势任务的需求存在巨大差距，急需根据电子信息装备软件试验和电子信息装备软件测评的需求，从顶层重新规划建设可灵活构建电子信息装备软件试验环境和测评环境、可集成各种软件测试工具、可对软件试验和软件测评全过程进行统筹规划和管理、可支持实现测试大数据的云测试平台，并通过在云测试平台上补充多种语言的软件静态代码分析和单元测试能力，新增电子信息装备非嵌入式软件部件、配置项和系统测试能力，形成体系化、综合性的电子信息装备软件试验与测试能力。

为达到以上效果，按照项目的建设需求，需要开展的建设内容主要包括以下几点：

（1）配齐软件测试工具体系

调研行业软件测试工具的工具体系，配齐成熟的、行业认同度高的主流软件测试工具，主要包括自动化功能测试工具、性能测试工具、软件源代码自动分析工具、软件可靠性测试用例辅助设计工具、代码质量测试工具、数据标准规范检查工具、逻辑覆盖率测试工具、FPGA混合仿真验证工具。

（2）建设和开发软件测试云平台

通过购置服务器、终端机、网络交换机、网络安全防护设备等，建立起信息系统的基础硬件资源；利用软硬件虚拟化、云计算、大数据等技术，构建小型的私有云数据中心；利用云基础设施，设计和开发软件测试云平台；通过对现有软件测试工具和新购软件测试工具的适当改造或集成应用，实现在测试云平台中能使用现有软件测试工具和新购的软件测试工具。

（3）集成和建设软件测试资产库管理系统

通过购置或新开发软件测试资产库管理系统，并进行系统的云化改造，集成到软件测试云平台中，以满足软件测试全过程的规划与管理。

通过上述建设任务，形成用于各种不同自主可控程度的电子信息装备非嵌入式软件与嵌入式软件的静态分析、单元测试、部件测试、配置项测试和系统测试的测试测试能力，形成典型FPGA软件的混合仿真验证能力，满足业主所要求的分布式信息系统软件装备试验、电子信息装备软件测评任务的需要。

# 项目建设的技术要求

## 用途

项目建设主要用于保障电子信息装备软件尤其是信息系统装备软件的软件试验和软件测评，同时兼顾基地试验保障装备软件的软件测评，可用于各种不同自主可控程度的电子信息装备非嵌入式软件的静态分析、单元测试、部件测试、配置项测试和系统测试，同时可用于典型FPGA软件的混合仿真验证。

建成后主要具备以下功能：

(1)软件测试云平台功能。主要通过建设软件测试云平台实现，包括：测试环境的动态生成、测试进程调度控制、测试数据采集和分析处理、测试对象集中管理、被测对象的分析评估、测试大数据应用支持、测试工具集成和数据共享等功能。

(2)软件测试功能。主要依托在测试云平台上集成主流非嵌入式软件测试工具和典型FPGA软件测试工具实现，包括：不同自主可控程度的电子信息装备非嵌入式软件的静态分析、单元测试、部件测试、配置项测试和系统测试等功能，典型FPGA软件的混合仿真验证功能。

(3)软件测试全过程规划管理功能。主要通过在测试云平台上集成软件测试资产库管理系统实现，包括：项目管理、决策过程管理、需求管理、过程与产品质量管理、设计管理、配置管理、开发管理、系统管理、测试管理、证书初始化与管理、源码比对与追踪、软件集成发布、软件问题管理、版本状态管控、软件资产管理、软件资产信息管理、资产库统计和知识库管理等功能。

## 系统组成及功能

### 系统总体架构

主要建设内容包括“四层五系统”。“四层”是指逻辑组成上包括硬件资源层、云基础支撑层、云化工具层、测试应用层等四层。“五系统”是指系统物理组成上包括云基础硬件资源、软件測试云平台、软件测试工具集、软件测试资产库管理系统、云桌面端机系统等5大系统。系统总体组成示意图如图1所示(可供参考)



图1系统总体组成示意图

### 云基础硬件资源

#### 设计原则和要求

1.安全性可靠性

高可靠性是云基础硬件资源的关键,不允许出现服务中断类故障,在云基础硬件资源的总体布局、结构设计、设备选型、日常维护等各个方面均需进行可靠性的设计，在关键核心设备采用硬件备份、冗余等可靠性技术的基础上,采用相关的软件技术提供较强的管理机制、控制手段和事故监控与安全保密等技术措施提高安全性。

2.可扩展性

云基础硬件资源的设计中,要充分考虑测试云平台建设的后续扩展需求预留灵活的扩展能力。不仅在机桓中预留足够的空间,还要综合考虑未来扩展设备的环境保暲需求、网络互连需求,要保证整个基础硬件资源可灵活进行模块化扩展。

3.标准化

在云基础硬件资源结构设计时,要符合国家颁布的有关标准的要求,包括各种建筑、机房的设计标准,电力电气保障标准以及计算机网络标准。

4.可管理性

在云基础硬件资源的设计中,建立一套全面、完善的管理和监控系统。所选用的设备应具有智能化、可管理的功能,同时采用先进的管理监控系统,实时监控和监测整个云基础硬件资源的运行状况,确保迅速定位故障,为云基础硬件资源的安全、可靠的运行提供有力保障。

#### 设备组成

1.服务器及网络接口设备

主要由13台华为高性能服务器和3台交换机等组成，主要提供CPU、内存、硬盘、网络、接口等云基础硬件资源。

2.端机系统

规划建设200台端机，本项目建设20台。每台端机包指端机主机、显示器、键盘和鼠标各1套。

3.机柜

选择符合IEC( International Electro technical Commission)60297-1标准的5台标准服务器机柜，为服务器提供可靠稳定的安装空间，保证服务器的安全运行。机柜尺寸统一，并采用前后风道。机柜提供42U可用空间，静载不小于1500kg

4.配电系统

主要由机架式安装的配电单元及电源分配单元(PDU)组成，结构化设计，可拆卸，便于日常维护，与机柜整体集成，无需现场安装。

5.环境管理系统

环境管理系统由空调室内机、室外机等组成。采用模块化的冷热通道封闭方案，配置不小于30KW的精密列间变频空调。

6.辅助设备

辅助设备包括网络安全防护设备、端机操作台、机房防静电地板等，外接共享打印机1台、共享刻录机1台。

### 软件测试云平台

软件测试云平台主要由云基础分系统、测试运行支撑分系统、测试数据采集与分析分系统组成，如图2所示



图2软件测试云平台组成结构

1.资源虚拟化

对基础硬件资源系统提供的CPU、内存、存储、网络等资源进行虚拟化。主要包括计算资源虚拟化、存储资源虚拟化、网络资源虚拟化等。资源虚拟化子系统对系统资源进行统一登记、标识、管理和状态监视。

2.虚拟机管理

对云平台生成的虚拟机进行可视化管理。主要包括虚拟机的申请、生成管理,已有虚拟机的启动、挂起、关闭等操作,虚拟机运行状态监视等。虚拟机管理子系统使用户可基于基础云平台进行资源的申请与使用。

3.虚拟存储管理

对虚拟存储资源进行可视化管理。主要包括虚拟存储资源状态监视,临时存储资源管理、永久存储资源管理以及存储资源的动态分配管理等。虚拟存储管理子系统确保用户数据的有效、可控、安全。

4.虚拟网络管理

对虚拟网络资源进行规划、构建和管理。主要包括以图形化的方式进行虚拟风络拓扑规划,进行路'设置,为测试项目构建独立的网络环境:实现测试云平台内部虚拟网终与外部实际网络的互连:对构建的虚拟网络环境进行验证确认和状态监视。

5.镜像与备份管理

对虚拟机镜像以及测试环境备份进行管理与维护。主要包括虚拟机镜像的生成、存储、使用、删除等管理,分布式测试环境中虚拟机集群的统一备份、标识、恢复、删除等管理。

6.运行与安全管理

对云平台的运行环境、运行状态进行监视和管理,具有安全防护机制保证云平台的系统安全、数据安全,进行访间权限管理、防病毒管理、安全日志管理等。

7.测试工具云平台适配

完成己有、新集成软件测试工具和开源软件测试工具的云平台适配。提供 License集中登记、管理和监控功能；单机版测试工具的封装和面向分布式环境的功能转换功能；B/S架构、CS架构测试工具的云环境配置功能。

8.测试环境规划设计

对用户构建分布式网络测试环境的需求，提供测试环境规划设计功能，采用可视化手段，主用户以“所见即所得”的方式，规划设计测试需要的网络拓扑图、各节点硬件配置、操作系统、网络地址、被测件、测试工具软件以及相关运行支撑软件环境等。规划设计完成后生成測试环境规划设计方案，可直接提交测试云平台执行虚拟化,生成分布式网终測试环境。同时该子系统还提供相关的维护管理功能,包括测试环境规划设计方案的浏览查询、复用、删改、存储等。

9.测试环境定制与管理

可采用测试环境规划没计或手工定制两种方式,生成测试环境。针对用户构建分布式网络測试环境的需求,采用直接利用测试环境规划设计方案,执行批量虚拟化操作，生成整个测试环境。对单用户的单一测试环境需求，可以采用手工定制测试环境的方式,生成与用户定制需求一致的虚拟机测试坏境,包括硬件配置、操作系统、软件环境、软件测试工具等。该子系统可完成整个测试环境的总体挂起、存储与恢复,可完成测试环境的可视化管理。

10.测试运行监视控制

集中控制复杂分布式测试进程的开始、暂停、恢复、终止等操作。实现分布式功能测试工具和性能测试工具的同步、消息传递和集中控制,实现各种测试工具运行状态的监控,可为测试过程数据的采集提供接口。提供第三方运行监视功能,可根据用户权限,对测试运行过程中各结点端机桌面和用户換作过程进行监视。

11.被测对象管理

对被测对象进行标识、入库、出库、更动控制、版本管理、访问权限设置等,为系统提供独立手測试项目的被测对象存储、管理、维护环境。

12.用户管理

对用户进行注册、注销、登录等管理,维护用户基本信息,赋予用户角色,实现用户对被测对象、測试项目、测试工具、测试数据、测试流程、试计划、测试人员、測试日志等对象的访间权限规划,对用户的测试活动进行日志记录与存储。

13.测试配置数据采集存储

对测试配置数据,主要包括测试环境配置、测试工具配置、测试数据配置、测试人员配置,以及整个测试项目的总体策划规划文案等基本信息进采集、存储和管理,支持对测试配置数的深度分析,对支持测试配置数据的复用提供基本数据支撑。

14.测试过程数据采集存储

对测试过程中动态产生的各种测试数据和中间测试结果数据,例如测试用例以及用例执行情况的数据,进行采集、存储和管理。支持对过程数据的深度分析和测试报告的生成,对测试用例复用提供基本数据支撑。

15.测试结果数据采集存储

对测试结果数据,主要包括各软件測试工具生成的测试报告以及报告中的结论、各测试项的测试结论等数据进行采集、存储和管理。可对测试结果数据等进行深度分析,可支持对被测对象的综合分析。

16.测试数据综合分析

对同一被测对象不同版本、不同測试轮次的各神测试数据,例如测试配置数据、测试过程数据、测试结果数据的综合分析,可支持对被测对象总体评价报告的生成。可实现多个类似被测对象的综合数据分析,可支持优化同类被测对象的测试方案。

17.测试用例辅助设计支持

通过对多种被测对象、多个测试场景下不同测试用例的执行情况进行综合分析,形成典型被测对象在典型测试场景下的测试用例推荐,可辅助用户进行测试用例设计。

18.测试数据提取应用接口

该子系统是軟件测试云平台集中的对外数据发布的子系统,除了提供标准的数据库形式、文档形式的数据提取接口外,还提供外部应用软件可直接读取测试云平台数据的接口函数,用于支持后续测试大数据分析与应用。同时该子系统对提取测试数据的请求进行权限验证。

### 软件测试工具集

在测试云平台上，重点补充建设软件测试工具，包括软件源码分析测试工具、自动化功能测測试工具、逻辑覆盖率测測试工具等9种软件测试工具，详见表1。

表1拟补充建设的软件测试工具列表

|  |  |  |
| --- | --- | --- |
| 序号 | 软件测试工具 | 类别 |
|  | 自动化功能测试工具 | 非嵌入式 |
|  | 性能测试工具 | 非嵌入式 |
|  | 软件源码自动分析工具 | 非嵌入式 |
|  | 单元测试工具LDRA Testbed升级 | 非嵌入式、嵌入式 |
|  | 软件可靠性测试用例辅助设计工具 | 非嵌入式 |
|  | 代码质量测试工具 | 非嵌入式 |
|  | 数据标准规范检查工具 | 非嵌入式 |
|  | 逻辑覆盖率测试工具 | 非嵌入式 |
|  | FPGA混合仿真验证工具 | 嵌入式 |

### 软件测试资产库管理系统

软件测试资产库管理系统由测试软件研制过程管理分系统、軟件状态控制与管理分系统和软件产品管理分系统组成,分别服务于软件测试过程和工具集成开发的软件产品研制、软件测评、软件产品使用。如图3所示。



图3软件測试资产庳管理系统组成

1.测试软件研制过程管理分系统

测试软件研制过程管理分系统具备项目管理、决策过程管理、需求管理、过程与产品质量管理、设计管理、配置管理、开发管理、系统管理和测试管理等功能,管理标准参照《GJB5000A军用软件研制能力成熟度模型》执行。

2.软件状态控制与管理分系统

软件状态控制与管理分系统具备软件信息管理、证书初始化与管理、源码比对与追踪、软件集成发布、软件问题管理、版本状态管控、系统管理和系统通讯录等功能,管理标准参照《GJB2725A-01测试实验室和校准实验室通用要求》执行。

3.软件产品管理分系统

软件产品管理分系统具备测试资源管理、被测软件管理、资产库统计知识库管理和系统配置与管理等功能,管理标准参照《GJB900C-2017质量管理体系要求》执行。

### 云桌面端机系统

云桌面端机系统由浏览器/服务器架构的端机和客户端/服务器的端机等两种端机组成,其主要组成示意图如图4所示。



图4软件测试资产库管理系统组成示意图

浏览器/嚴务器架构的端机主要为用户执行门户网站式軟件测试时提供操作使用平台;客户端/服务器的端机主要为用户执行定制客户机测试环境式软件测试时提供软件测试操作使用环境。

## 典型应用模式

主要描述系统两种典型的应用模式：软件试验中最为复杂的分布式信息系统软件装备的试验模式、电子信息装备软件测评任务模式。软件测评任务中黑盒測试部分（指功能测试、性能测试、接口測试等）任务的应用模式与软件试验的类似。

### 分布式信息系统软件装备试验应用模式

分布式信息、系统软件具有节点多，组网复杂，应用模式灵活等特点，软件测试云平台可提供基于需求的精确计算资源、软件可定义的网络拓扑结构模式可重用的软件測试环境,其试验环境构建如图5所示。



图5分布式信息系统软件装备试验应用模式

通过云计算技术形成计算、存储、网络等虚拟资源池,根据软件装备研制总要求和软件实际运行环境,申请计算、存储等运行资源,作为席位节点部署信息系统软件、云化軟件测试工具等:根据装备软件运行模式和组网需求申请网络资源,依据信息系统软件组网模式软件定义网络拓扑结构;实现仿真试验系统的云化部署,作为试验配试资源,提供传感器、信息系统节点等仿真模拟;对于试验中需要,而又无法实现云化部署的真实装备、模拟工装等,可通过外部网络接入测试云平台内部,形成外部真实装备、云化仿真试验系统、云化測试工具与被测信息系统软件一体集成的综合测試环境。试验过程和结果数据由数据采集与分析分系统集中存储,并基于大数据技术进行测试数据的分析与评估。

### 电子信息装备软件测评任务应用模式

在电子信息装备軟件测评任务应用模式下,软件云测试系统主要实现软件测试工具的云化应用,针对测试工具的可云化程度,主要分为三种应用方式:多用户并行使用方式、多用户分时使用方式、单用户登陆使用方式,如图6所示。



图6电子信息装各软件测评应用模式

多用户并行使用方式：对于可完全云化的测试工具,每个云用户节点均可获取完整的使用权限,多用户可同时进行多任务的测试

多用户分时使用方式：对于使用浮动 License云化的测试工具，同一时间只能有一个用户获取使用权限，实现 License党争管理机制，多用户可分时进行多任务的测试。

单用户登陆使用方式:对于已有的单机软件测试工具，或不能云化的软件测试工具，静态安装于基于云平台的指定虚机上，采用登陆管理的方式实现单用户登陆使用。

## 技术指标和使用要求

### 总体指标

a)可灵活构建软件测測试环境,文持电子信息装各非嵌入式软件(含自主控软件)的软件试验和软件测评。

b)支持靶场保障条件建设装备中各类非嵌入式软件的软件测评,可支持典型FPGA软件的混合仿真验证。

c)能够涵盖的电子信息装备非嵌入式軟件的测试级别包括单元测试、部件测试、配置项测试和系统测试

d)能够涵盖的电子信息装各非嵌入式软件的测试类型包括静态分析、代码审查、功能测试、性能测试、接口测试、标准符合性测试、人机交互界面测试、强度测试、余量測试、边界测试、安全性测试、可靠性測试、恢复性测试、安装性测试、兼容性測试等测试类型

e)能够同时部署开展的软件试验和软件测评任务数量不少于5个;单任务模式下,最大可支持不少于100个节点的指挥信息系统开展軟件试验和软件测评。

f)软件测试工具总体要求:软件测试工兵优先选用成熟的、行业认同度高的主流软件测试工具;单机版测试工具的 License不少于2个,至少可保证License在云测试平台使用,另一个 License在单机上使用;除FPGA混合仿真验证工具外,其他软件测试工具均能够与软件测试云平台集成,测试数据能与平台共享。

### 分系统指标

#### 基础硬件资源系统

构建小型私有云数据中心,为软件测试云平台、软件测试工具运行以及软件测试资产库管理系统提供所需的硬件环境,具体指标如下：

a)基于可扩展的模块化机房方式构建,模块化机房集成服务器、存储设备、网络设备、供电设备、空调设备、环境监测设备等,无需对机房环境进行大规模基础建没。

b)计算资源指标。7台计算节点服务器,基本配置不低子:2颗英特尔至强E5-2650v4处理器:4×32GDDR4 RDMM内存:2×2000GB通用硬盘SAS:支持RAID;4×10GE端口。

e)存储资源指标。6台存储节点服务器,基本配置不低于:2颗英特尔至强E5-2650v4处理器:4×32GDDR4 RDMM内存:10×2000GB通用硬盘SAS;2×800GB固态硬盡-SATA6Gb/s:支持RAID:4×10GE端ロ。

d)12U高性能一体化机絶模块指标。2个12U高性能一体化机箱,基本配置不低于:2块机框管理模块:14个风扇框;6块冗余交流电源:2块交换模块板:4块10GE光模块板;配套线缆及安装滑轨2个。

e)网络资源指标:2台万兆交换机,接口不少于48个;1台千兆交换机接口不少于48个。服务器与交换机以万兆接互连。

f)测试云平台配各UPS电源,外部断电情況下可支持系统连续运15min 。

g)其他指标:测试云平台外接串口不少于50路;具有视频切换与视频输出功能,实现对虚拟机界面的图像拼接和切换输出。

h)布线安装指标:包括机架内布线、操作席位布线、演示环境布线、云桌面端机布线等

i)机.柜主要技术参数包指:外形尺寸(高x宽x深)2000m×600mm×1350mm;机柜提供42U可用空间:静載不小于1500kg:前后风道通风设计

j)配置入侵检测測、漏润扫描、日志审计、防火墙、终端安全管理、网络防病毒等网络安全防护设备。

#### 软件测试云平台

软件测试云平台包括云基础分系统、测试运行支撑分系统和测试数据采集与分析分系统三部分,云基础分系统与测试运行支撑分系统和测试数据采集与分析分系统之间采用松耦合方式构建。云基础分系统统一管理系统资源,对计算资源、存储资源、网终资源等进行虚拟化,提供资源的按需分配与管理,并开放管理接口,满足軟件测试平台对系统资源的需求。

1.总体指标

a)平台可在集成的测试工具支持下完成电子信息装各非嵌入式软件的静态分析、代码扫描、单元测试、配置项测试、系统测试等測试任务。

b)平台具备同时开展多项测试任务的能力,同时支持的测试任务数量不少于5个,可支撑大規模分布式软件系统的集成测试。

c)可统一管理多种硬件资源,实现虚拟机的生成、部署和联网,提供虚拟机终止、挂起、存储和唤醒服务。

d)可对计算机网络进行虚拟化,实现软件定义网络功能。

e)可对虚拟机采用的操作系统、功能软件、硬件配额(CPU核心数、内存大小、硬大小)、网络环境和测试工具进行订单式的选配及安装,形成满足不同测试任务需求的测试环境集(测试计算机网络),并可对测试环境集进行整体备份和恢复。

f)支持不同测试工具间的测试数据的互联与共享,支持测试大数据分析。

g)可实现测试过程数据和结果数据的实时采集、管理和存储,可自动分析处理測试数据,形成测试报告,并可对测试报告进行定制。

2.云基础分系统指标

基础云平台统一管理系统资源,对计算资源、存傜资源、网络资源等进行虚拟化,提供资源的按需分配与管理,并开放管理接口,满足软件测试平台对云平台管理的集成要求。基础云平台应主要包括以下功能:

a)基础云平台应具有硬件资源可扩展、按需分配、弹性伸缩、负载均衡等云平台的基本功能。

b)支持 Docker容器功能,可实现测试工具、測试环境的轻量化部署。

e)具有虚拟机定制生成功能,文持 Linux、 Windows和 Kylin OS等多种操作系统的虛拟机生成,单个虚拟机支持TB级存储卷。

d)具有网络虚拟化功能,支持虚拟机浮动IP,支持跨网段虚拟路由配置能够在虚拟机节点间构建多级交换网络,支持虚拟机节点与外部网络间的互联。

e)具有大数据存储与管理功能。对測试数据进行存储管理,支持后续大数据分析功能升级。

f)具有镜像与各份管理功能。支持虚拟机镜像的生成、存储、使用、删除等管理,支持虚拟机集群管理,对虚拟机集群进行统一各份、标识、恢复、删除等管運。

g)具有安全防护功能,保证云平台的系统安全、数据安全。支持访问权限管理、防病毒管理、安全日志管理等。

h)支持加密狗授权软件的使用。支持在服务器USB接口接入加密狗,可在云平台生成的任意虚拟机上使用该软件授权(不同时使用具有串口虚拟化功能。可在云平台生成的虚拟机上实现串口虚拟化,虚拟机与虚拟机间,虚拟机与外部设备间可通过串回进行通信。

j)具有虚拟机界面监视功能。在不影响虚拟机操作员操作的前提下,可以将虚拟机操作界面切换至监视端显示。

k)具有用户管理功能。支持多用户共享资源,支持多用户按需动态分配资源,支持用户权限控制。

l)具有虚拟机管理功能。支持虚拟机的按需申请、自动生成,实现虚拟机的启动、挂起、关闭等操作,可对虚拟机运行状态进行监视,具有虚拟机配置开机执行脚本等功能。

m)具有可视化管理与运维界面,可对系统资源进行统一登记、标识、管理和状态监视,可单独实现对云平台的管理操作。

n)云基础及硬件管理功能。完成云计算主机群的安装、互联和配置,为上层的云计算提供硬件支撑。完成云主机操作系统的安装和系统计算、内存网终、存储资源的虚拟化功能提供云环境配置功能。

3.测试运行支撑分系统指标

测试运行支撑分系统主要对软件测试过程和实施进行管理,包括测试对象管理、測试环境构建、测试运行控制等,主要功能指标包括:

a)被测对象管理功能:建立被测对象生命周期管理数据库,实现基于用户角色的被测对象入库、出库、更动、版本管理功能,为系统提供独立于测试项目的被测对象存储、管理环境。

b)测试环境定制功能:提供测试环境可视化定制功能,具备操作系统、常用功能软件、测试工具软件的定制、初始化与部署功能。支持基于虚拟机的部署和可视化虚拟网络括扑规划设计,支持测试环境集的整体备份、恢复等管理功能。

c)测试运行管理功能:支持虚拟机的创建、删除、挂起、迁移、扩容等。

a)用户管理功能:实现被测对象、測试项目、测试工具、测试数据、澳测试流程、测试计划、测试人员、测试日志等对象的访间权限规划。对关键试活动进行日志记录与存储。

e)云平台软件工具适配功能:完成已有、新购及开源软件测试工具的云平台配。提供软文件、硬件Key、网终版 License的集中登记、管理,实现虚拟机与 License的适配;单机版测试工具的封装和面向分布式云环境的功能转换:B/S架构、CS架构测试工具的云环境配置功能

4.测试数指采集与分析分系统指标

a)具有数据搜集与存储功能。能够对测试配置数据、测測试过程数据、测试结果数据进行搜集和存储;可实现格式化数据和非格式化数据的存储与应用;具有大数据分析扩展接口,可为后续大数据处理提供支撑。

b)具有测试数据综合分析功能。具有用户可定义的软件评价体系,能够根据评价指标对测试数据进行综合分析,并以可视化方式显示分析结果,同输出评价分析报告。

c)具有测试用例辅墈设计功能。通过对测试需求进行分析,基于历史测试场景、测试用例、测试数据等己有信息,形成典型被测对象在典型测试场景下的测试用例推荐,可辅助用户进行测试用例效计。

5.性能指标

a)測试云平台对硬件资源进行管理,所管理的服务器模不小于13台,可扩展至50台以上;管理存储资源规模不小于200T,最大可扩展至不少于1000T。

b)测试云平台能够虚拟出不少于200个虚拟机(配置为2核CPU、4G内存、100G存储容量),支持自定义虚拟机资源配置。

c)支持管理200个以上的虚拟机镜像:创建50个虚拟机实例用时10min;完成单虚拟机整机各份时间小于5min。

#### 软件测试工具

1.自动化功能测试工具技术指标

a)提供图形化和关键字驱动测測试功能,为非编程技术人员提供图形化的关键字视图,为数据驱动测试提供熱悉易用的数据表。

b)提供测试流程模板(来自于实的流程指导),用户也可以自定义流程。

c)能够和测试管理平台集成,在应用开发生命周期中实现案例的管理和重用。

d)支持基于数据驱动的功能测试,可按用户自定义方式模拟生成功能测试所需的各种测试数据。

e)支持历史測试数据文件方式生成功能测试所需的测试数据,驱动被测软件自动运行。

f)支持典型分布式软件系统(或体系),例如指挥信息系统的自动化功能测试,支持规模不低于旅(团)级(或分布式节点数不小于24个)。

g)对于不断变化的应用,易于创建、执行和维护回归测试。

h)支持文档的自动生成,具有易用性和内置的环境支持支持运行的操作系统: Windows系列操作系统、 Linux操作系统等。

j)浏览器:支持IE、 Firefox等主流浏览器:其中IE浏览器版本不低于6.0SP1, Firefox浏览器版本不低于1.5。

k)支持对以下类型的应用程序进行自动化功能测试:标准 Windows应用程序,包括基于Win32API和MFC的应用程序、 Activex控件、 Visual Basic应用程序、Web页面。

2.性能测试工具技术指标

a)具有非侵入性的实时性能监视功能,可提供被测系统主要性能指标的监视。

b)可支持用户定义方式自动生成压力测试数据,支持对被测软件进行压力测试和稳定性、可靠性分析。

c)可测试分析被测软件运行时对CPU等硬件资源占用率、系统响应时间等关键性能指标

d)支持对以下类型的应用程序进行自动化性能测试:标准 Windows应用程序,包括基于Win32API和MFC的应用程序、 Activex控件、Web页面。

e)支持对典型分布式软件系统(或体系),例如指揮信息系统软件的整体性能测试,支持规模不低于旅(团)级(或分布式节点数不小于24个)。

f)具有测试脚本录制功能。

g)支持文档自动生成,具有易用性和内置的环境支持。

h)支持运行的操作系统: Windows系列操作系统、Linx操作系统等。

3.软件源码自动分析工具技术指标

a)至少可支持CC-+、Java、NET等3种语言的软件源码自动分析:NET语言的源码自动分析至少应支持检査C#、VB.NET、 ASP. NET源码的规范。

b)可以导入和导出一个函数、一个文件和整个工程的测试用例。

c)可以按“自动生成桩函数”、“使用原函数作为桩函数”和“自定义桩函数”三种方式生成桩函数。

d)内置一定数量的代码编程规范,提供图形化的代码编程规范的定制工具,支持用户自定义代码编程规范,提供自动化的代码编程规范检查功能。

e)支持对内置代码编程规范和用户自定义代码编程规的修改。

f)可图形化地实现代码走查,能检査未初始化而引用的变量和引用而未释放的变量,能査找空指针异常、数組越界、除零等锆误。

g)能自动执行白盒测試,确保代码中每一条独立的路径至少执行一次,支持圈复杂性指标的测试。

h）能根据某种质量模型评价代码的质量。

i)能够根据需要自动生成测试报告,并根据用户的需求对报告格式进行定制。

4.单元测试工具 LDRA Testbed技术指标

a)具有编程规则验证、数据流分析、控制流分析、表达式分析、接口分析、软件质量度量分析等功能。

b)可提供多神代码覆盖率分析,主要包括:语句覆盖、分支/判定覆盖、LCSAJ覆盖、过程/函数调用覆盖、分支条件覆盖、分支条件组合覆盖、修正条件判定覆盖、动态数据流覆盖等。

e)支持图形化显示功能,主要包括:柱状堅、流程图、调用图、 Kiviat图等的显示。

d)支持CC++、Java等语言。

e)支持的代码规则主要包括:电子信息装备软件尤其是信息系统装备软件领域代码规则集:以及国际主流代码规则集:并支持自定义新建代码规则集。

5.软件测试用例辅助设计工具技术指标

a)习导用户构建图形化的软件测试剖面不少于5种。

b)支持构建可靠性測试割面,例如客户割面元素不少于10个,用户面元素不少于20个,状态剖面元素不少于50，功能剖面元素不少于100个，操作剖面元素不少于100个。

e)可一次自动生成XML格式的软件测试用例数据最大不少于1万个。

d)生成包含1万个测试用例数据的文件时间不超过1min。

e)支持测试人员开展规范的软件测试剖面构建和测试用例设计工作。

f)支持用户、状态、功能、操作等需求要素的层次关系和概率信息描述,同时支持描述各妻素之间的转移关系和时序约束等信息,可充分反映软件运行过程中的各类场景和外部激励。

g)可依据软件测试剖面及外部接口取值分布特征,结合外部接口环境文件,自动生成海量XML格式的用例数据,用于软件功能测试。

h)支持运行的操作系统: Windows系列操作系统、 Linux操作系统等。

6.代码质量测測试工具技术指标

a)至少支持C/C++、Java、C#等3种语言。

b)支持代码缺陷裣测、安全漏洞检测、度量分析、架构分析、质量趋势分析、架构约東分析等功能。

c)支持 Windows、 Linux、 Kylin OS等操作系统。

d)不需要写测试月例就能找出程序中的关鍵bug,包括程序动态运行时发生的错误(如指针越界、内存泄露、溢出、数组越界、变量未初始化等),并且能够准确定位程序出错的位置。

e)支持主流编码规则。

f)支持 Mccabe复杂度、 Halstead程序度量、代码行数、继承数、循环数等各种基本度量。

g)支持缺陷按级别、类型、存在状态、处理状态的分析统计。

h)支持自动生成缺陷报告、缺陷报表査看和导出、多样化的结果统计(bug分类统计、bug状态统计、函数复东度、多神缺陷过滤标准等)。

7.数据标准规范检测測工具技术指标

a)可支持 Oracle、 SQL Sever、 MYSQL、达梦、金仓等多种关系型数据库

b)支持多种数据标准

c)单检查规则测试时间:在100M局域网环境中,在标准库数据表中数据小于1000个时,单检査规则测试每分钟检登不少于100张数据表。

d)测试报告导出时间:在问题数小于1000个时,测试报告导出时间小于3分钟。

8.逻辑覆盖率测试工具技术指标

a)支持C、C++语言的插桩

b)插桩膨胀率控制在15%以内

c)可以快速定位系统性能的瓶颈所在,即时非抽样监控超过12万个C/C++函数、150万行代码。

d)支持4种覆盖率:块覆盖、语句覆盖、判定覆盖、MC/DC

e)支持网口、串口、CAN、JTAG等测试接口。

f)支持离线查看测试结果。

g)在测试过程中,可以实时更新程序的各种语句、判决、MCDC的盖率、函数的流程及调用关系、内存、A- B Timer。

9.FPGA混合仿真验证工具技术指标

a)支持 Verilog、VHDL、 System Verilog、 Systemc、 Open Vera、混合语言仿真。

b)支持主流FPGA芯片厂商的主要器件的仿真,具有较好的RTL与门级仿真性能。

e)支持性能分析功能,能够分析性能瓶颈。

d)支持仿真时访问VHDL或 Verilog混合设计中的下层模块的信号,便于仿真调试。

e)支持仿真恢复与继续功能。

f)提供设置复杂断点、信号眼踪等调试功能。

g)支持SVA等断言的调试。

h)支持代码覆盖率统计功能。

i)支持在通用的 Linux操作系统下运行。

#### 软件测试资产库管理系统技术指标

1.总体指标

a)能够与软件测试云平台实现功能集成,数据能与测试云平台共享、主要管理功能可通过云平台实现统一调度。

b)具备项目管理、决策过程管理、需求管理、软件产品过程与质量管理、设计管理、配置管理、开发管理、系统管理、测试管理、证书初始化与管理、源码比对与追踪、软件集成发布、软件题管理、版本状态管控、测试资源管理、被测软件管理、资产库统计和知识库管理等功能。

c)对外预留数据和功能的开放接口,支持软件试验和软件测试的大数据分析评估。

2.主要功能指标

(1)测试软件研制过程管理分系统

a)测试软件研制过程管理分系统具备项目管理、决策过程管理、需求管理、过程与产品质量管理、设计管理、配置管理、开发管理、系统管理和测试管理等功能,管理标准参照《GJB5000A军用软件研制能力成熟度模型》执行。

b)项目管模块能满足软件开发组织对软件项目在进度、工作量、质量、资源等进行集成管理的需要。能够基于平台进行项目策划、资源分配、进度成本汇报眼踪、测量与分析、风险管理、质量保证;能够基于平台提供的各种视图,了解项目的状况、问题点和各项综合统计指标。

c)决策过程管理模块支持对项目间题进行决策分析正式评价过程的管理,包括:判断问题是否需要采取正式评价,如需要正式评价,则标识间题各选方案,按照评价准则对备选方案进行正式评价,记录正式评价过程数据远中最终方案并在风险管理中识别相应的风险。

d)过程与产品质量管理模玦支持过程和产品质量保证任务的策划、执行、跟踪。支持从组织资产库中导入常用的裣查单,支持项目中检査项的删减；支持在审核时发现的不符合项的在线处理流程,支持不符合项的汇.总功能;支持生成质量保证的阶段报告、里程碑报告及质量总结报告。

e)配置管理模块应满足相关标淮和规范,以软件配置管理为核心,同时支持需求管理和缺陷追踪的功能,具有软件配置策划管理、变更控制、版本管理、产品依赖关系管理、产品一致性管理、軟件问题追琮管理、软件配置状态审计管理等功能。

f)需求管理模块可管理项目的产品和产品部件的需求,并标识这些需求与项目的计划和工作产品之间的不一致性。週过需求管理系统可管理项目接收的或产生的所有需求,以及组织强加于项目的需戒。需求在项目期间演化时,通过变更管理系统管理需求的更改,同时在需求分析的过程中,维护需求和工作产品的双向可追溯性。即完整的需求管理流程应包括需求的建立需求的分析、需求的变更及需求的双向性追溯等。

g)设计管理模块能够实现对设计阶段的配置项集中管理,实现软件研制设计阶段配置项的出库、入库、版本发布、版本管理。

h)开发管理模块可以有效对软件产品研制开发阶段的源代码实现开发库、受控库和产品库的三库管理模式,支持源代码的出库、入库、版本发布版本管理。

i)测试管理模块符合相关标准中的管理要求,能够满足软件产品测试的业务需求。可为测试人员提供测试用例辅助设计和测试用例管理功能,可辅助测试人员梳理测试需求、制定测试计戈划、设计测试用例,并依据测试结果分析产生出测试报告,支持测试工作的可移动模式,应能够在 Windows、 Linux、麒麟操作系统环境下独立安装;支持向导服务方式引导测试人员按照软件测试各个阶段的要求开展相应的工作;支持测试用例自动推荐,能够对历史测试用例进行分析。在测试用例辅助设计过程中,能够根据当前项目信息、测试功能点和该功能点的测试需求,自动推荐出合理的测试用例。

j)系统管理模块应满足保密“三员”管理要求,即系统管理员、安全保密员(权限角色管理)、安全审计员(审计系统管理和安全管理行为)。满足涉密信息系统的管理,系统管理员负贡系统的日常管理工作,安全审计员负责系统使用日志的管理,安全保密管理员负责系统的安全措施设定。

k)能够与软件测试云平台集成,数据能与平台共享。

(2)软件状态控制与管理分系统

a)软件状态控制与管理分系统具各软件信息管理、证书初始化与管理源码比对与追踪、软件集成发布、软件问题管理、版本状态管控、系统管理和系统通讯录等功能,管理标准参照GJB2725A执行。

b)软件信息管理模块能够依据软件所属的装各谱系、装备、分系统对软件信息进行管理,维护软件版本变化,管理軟件各版本对应的可采集资源。通过树形结构和列表的形式,对装备谱系、装备、分系统、软件以及软件版本做统一管理,确定软件状态、记录软件变更历程、采集软件版本资源并确定其版本状态。

c)源码比对与追踪模块能够对不同软件源代码进行比对分析,形成比对报告和统计报告,计算版本相似度。可自动建立軟件版本变更和变更需求之间的追踪关系;分析不同软件版本间的差异,统计差异文件,形成文件差异清单,建立和维护软件版本和变更需求间的追踪关系。

d)软件间题管理模块能够管理软件版本下存在的间题,实现问题列表的导入、导出。记录软件版本存在的问题,为软件维护提供依据。

e)系统管理模块能够实现组织、人员管理;实现角色设置和权限定义实现系统使用过程中的基础数据和类型定义,包括:问题类型、资料类型等。

f)证书初始化与管理模块能够管理用户,生成和带有用户唯一身份标识的公、私钥证书,作为使用状态管控应用分系统的身份识别。

g)软件集成发布模块能够对应生成软文档的安全文库(单一文件),其逆向过程为解包,由安全文库还原出原始软件文档。

h)版本状态管控模块能够对应生成软件状态文件,以及根据軟件状态对软件实施验证鉴别。

i)系统通讯录模块在用户安装客户端后,可获取联系单位或联系人员的公钥信息,生成系统通讯录,方便安全文库的共享和状态文件会签。

i)能够与软件测试云平台集成,数据能与平台共。

3)软件产品管理分系统

a)软件产品管理分系统具备测试资源管理、被测软件管理、资产库统计知识库管理和系统配置与管理等功能,管理标准参照G.JB9001C执行。

b)测试资源管理和被測软件管理模块能够实现软件资产实体的接收、入库、领用及处置流程管理,并结合软件状态管理工具,实现对软件版本的有效控制,可视化展示软件迭代曲线。

c)测试资源管理模块能够支持软件资产在软件全生命周期的相关信息的存储、管理和维护,对软件测试资产库中的信息发布、废弃、修订,提供增、删、改、査等操作,并根据授权管理实现资产库的浏览和操作权限的控制。

d)资产库统计模块可对软件资产信息进行分类统计,实现软件资产生命周期可视化展示,可视化内容丰富、形式多样。

e)具备知识库管理功能,可支持对各类文档模板的自定义上传,自定义分组管理,支持建立文档模板与标准过程的关联关系。

f)系统配置与管理模块能够实现对标程及过程包含的活动的定义管理,包括标准过程定义、发布、废弃、修订和历史记录;能够实现对软件资产管理制度和质量管理体系等标准、流程、规范的存储与维护更新;可分类建立风险库、问题库、测量库、配置管理库、过程评估与改进库、项目归档库等专项资产信息的统一管理平台:能够实现自定义设置多层次资产目录结构,支持对资产目录的维护:支持自定义资产状态、资产来源、資产类别等选项字典。

g)系统基础管理功能模块具有用户管理、组织管理、权限管理、三员管理、审计管理和安全设置等功能。

h)能够与软件测试云平台集成,数据能与平台共享。

3.性能要求

a)辅助生成测试扱告的时间小于3分钟,且有实时显示报告生成进度。

b)软件资产生命周期的可视化效果生成时间在1秒以内。

c)支持多用户并发使用,同时最大用户数不小于50用户。

### 使用要求

#### 环境适应性要求

a)硬件设备存储环境温度:-10℃~+50℃

b)硬件设各工作环境温度:0℃~35℃(空调开)。

c)硬件设备工作环境相对湿度:30%~-80%(+25℃）。

#### 可靠性要求

任务可靠性要求:单测试任务连续测试工作时间不小于48h。

#### 维修性要求

硬件设备平均故障修复时(MTTR)≤30min,关键件和重要件可修复、可更换。

#### 保障性要求

硬件设备采用市电供电，电压：AC220X(1ェ10%)V、频率:50X(1士5%)Hz。

## 质量控制

### 研制过程控制

合同生效之日起,研制单位在总体设计方案评审等重要节点前,应以书面形式提请组织会议评审。

### 验收

按验收地点的不同一般分厂所测试和靶场测试两个阶段。招标单位组织编写验收大纲。装备研制完成后,研制单位以书面形式提交验收申请。招标单位按照验收大纲开展验收工作。

#### 厂所测试

招标单位依托研制单位进行厂所测试,成立联合测试组。研制单位与招标单位讨论商定,拟制厂所测试实施细则。测试完成后联合测试组编制厂所测试报告。厂所测试所需保障条件由研制单位提供或协调解决。

#### 验收转阶段会

厂所測试报告编制完成后,招标单位按照验收大纲要求组织召开转阶段会议,研制单位配合组织会务工作,形成评审意见。评审通过后,转入靶场测试阶段。

#### 靶场测试

转入靶场测试阶段后，研制单位积极配合招标单位完成靶场测试。

### 移交

设备经验收合格,由招标单位组织设备初步移交。设备初步移交后,试用满一年或参加一次试验训练任务后,性能稳定且无遗留问题,招标单位组织办理正式移交手续。

### 质保期

正式移交后,免费质保2年。

### 其他

研制过程中,已审定的技术方案若确需调整的,须经招标单位认可后实施。

合同生效之日起,研制单位每季度向招标单位提供研制情况和质量管理筒报,如出现重大间题及关键岗位人员变动,研制羊位应10天内通报招标单。

对于大型或重要引进、外协部件,研制单位应严把质量关,招标单位参加合同签订和设备验收。

招标单位可根据需要,定期对设备的研制进展情况进行检查,研制单位应予以积极配合。

分系统研制完成或测试工具研制完成后,须进行测试,由中标方组织,招标方参加。

## 设备数量与研制周期

设备套量：1套。

研制周期：合同签订后30个月完成厂所测试，具备甲方验收条件。

# 对项目建设技术要求满足情况

## 系统组成及功能满足情况

### 系统总体架构满足情况

主要建设内容包括“四层五系统”。“四层”是指逻辑组成上包括硬件资源层、云基础支撑层、云化工具层、测试应用层等四层。“五系统”是指系统物理组成上包括云基础硬件资源、软件测试云平台、软件测试工具集、软件测试资产库管理系统、云桌面端机系统5大系统。系统总体组成示意图如图1所示（可供参考）

**应答：满足。**

**我方提供的云测试平台满足甲方所提出的系统总体架构要求。系统采用“高性能计算资源成熟云基础平台+灵活云测试服务+高质量软件测试工具”的技术架构，遵循云计算与云服务的技术体制，形成分层的技术架构。系统主要由以下五层组成：**

* **基础硬件设施层**
* **云基础设施层（Infrastructure as a Service，基础设施即服务）**
* **测试支撑平台层（Platform as a Service，平台即服务）**
* **测试应用层SaaS（Software as a Service，软件即服务）**
* **云桌面端机层**

**通过五层的架构设计，确保了系统具有较高的架构先进性，符合云平台与云计算的结构规范，确保系统能够满足非嵌入式软件测评需求、指挥信息系统软件全系统测试需求、典型FPGA软件的混合仿真验证需求和软件资产管理需求。**

### 云基础硬件资源满足情况

#### 设计原则和要求满足情况

1.安全性可靠性满足情况

高可靠性是云基础硬件资源的关键,不允许出现服务中断类故障，在云基础硬件资源的总体布局、结构设计、设备选型、日常维护等各个方面均需进行可靠性的设计，在关键核心设备采用硬件备份、冗余等可靠性技术的基础上，采用相关的软件技术提供较强的管理机制、控制手段和事故监控与安全保密等技术措施提高安全性。

**应答：满足。**

2.可扩展性满足情况

云基础硬件资源的设计中,要充分考虑测试云平台建设的后续扩展需求预留灵活的扩展能力。不仅在机柜中预留足够的空间,还要综合考虑未来扩展设备的环境保障需求、网络互连需求，要保证整个基础硬件资源可灵活进行模块化扩展。

**应答：满足。**

3.标准化满足情况

在云基础硬件资源结构设计时，要符合国家颁布的有关标准的要求，包括各种建筑、机房的设计标准，电力电气保障标准以及计算机网络标准。

**应答：满足。**

4.可管理性满足情况

在云基础硬件资源的设计中，建立一套全面、完善的管理和监控系统。所选用的设备应具有智能化、可管理的功能，同时采用先进的管理监控系统，实时监控和监测整个云基础硬件资源的运行状况，确保迅速定位故障，为云基础硬件资源的安全、可靠的运行提供有力保障。

**应答：满足。**

#### 设备组成满足情况

1.服务器及网络接口设备满足情况

主要由13台华为高性能服务器和3台交换机等组成，主要提供CPU、内存、硬盘、网络、接口等云基础硬件资源。

**应答：满足。**

2.端机系统满足情况

规划建设200台端机，本项目建设20台。每台端机包指端机主机、显示器、键盘和鼠标各1套。

**应答：满足。**

3.机柜满足情况

选择符合IEC( International Electro technical Commission)60297-1标准的5台标准服务器机柜，为服务器提供可靠稳定的安装空间，保证服务器的安全运行。机柜尺寸统一，并采用前后风道。机柜提供42U可用空间，静载不小于1500kg。

**应答：满足。**

4.配电系统满足情况

主要由机架式安装的配电单元及电源分配单元(PDU)组成，结构化设计，可拆卸，便于日常维护，与机柜整体集成，无需现场安装。

**应答：满足。**

5.环境管理系统满足情况

环境管理系统由空调室内机、室外机等组成。采用模块化的冷热通道封闭方案，配置不小于30KW的精密列间变频空调。

**应答：满足。**

6.辅助设备满足情况

辅助设备包括网络安全防护设备、端机操作台、机房防静电地板等，外接共享打印机1台、共享刻录机1台。

**应答：满足。**

### 软件测试云平台

软件测试云平台主要由云基础分系统、测试运行支撑分系统、测试数据采集与分析分系统组成，如图2所示。

1.资源虚拟化

对基础硬件资源系统提供的CPU、内存、存储、网络等资源进行虚拟化。主要包括计算资源虚拟化、存储资源虚拟化、网络资源虚拟化等。资源虚拟化子系统对系统资源进行统一登记、标识、管理和状态监视。

2.虚拟机管理

对云平台生成的虚拟机进行可视化管理。主要包括虚拟机的申请、生成管理,已有虚拟机的启动、挂起、关闭等操作,虚拟机运行状态监视等。虚拟机管理子系统使用户可基于基础云平台进行资源的申请与使用。

3.虚拟存储管理

对虚拟存储资源进行可视化管理。主要包括虚拟存储资源状态监视,临时存储资源管理、永久存储资源管理以及存储资源的动态分配管理等。虚拟存储管理子系统确保用户数据的有效、可控、安全。

4.虚拟网络管理

对虚拟网络资源进行规划、构建和管理。主要包括以图形化的方式进行虚拟风络拓扑规划,进行路'设置,为测试项目构建独立的网络环境:实现测试云平台内部虚拟网终与外部实际网络的互连:对构建的虚拟网络环境进行验证确认和状态监视。

5.镜像与备份管理

对虚拟机镜像以及测试环境备份进行管理与维护。主要包括虚拟机镜像的生成、存储、使用、删除等管理,分布式测试环境中虚拟机集群的统一备份、标识、恢复、删除等管理。

6.运行与安全管理

对云平台的运行环境、运行状态进行监视和管理,具有安全防护机制保证云平台的系统安全、数据安全,进行访间权限管理、防病毒管理、安全日志管理等。

7.测试工具云平台适配

完成己有、新集成软件测试工具和开源软件测试工具的云平台适配。提供 License集中登记、管理和监控功能；单机版测试工具的封装和面向分布式环境的功能转换功能；B/S架构、CS架构测试工具的云环境配置功能。

8.测试环境规划设计

对用户构建分布式网络测试环境的需求，提供测试环境规划设计功能，采用可视化手段，主用户以“所见即所得”的方式，规划设计测试需要的网络拓扑图、各节点硬件配置、操作系统、网络地址、被测件、测试工具软件以及相关运行支撑软件环境等。规划设计完成后生成測试环境规划设计方案，可直接提交测试云平台执行虚拟化,生成分布式网终測试环境。同时该子系统还提供相关的维护管理功能,包括测试环境规划设计方案的浏览查询、复用、删改、存储等。

9.测试环境定制与管理

可采用测试环境规划没计或手工定制两种方式,生成测试环境。针对用户构建分布式网络測试环境的需求,采用直接利用测试环境规划设计方案,执行批量虚拟化操作，生成整个测试环境。对单用户的单一测试环境需求，可以采用手工定制测试环境的方式,生成与用户定制需求一致的虚拟机测试坏境,包括硬件配置、操作系统、软件环境、软件测试工具等。该子系统可完成整个测试环境的总体挂起、存储与恢复,可完成测试环境的可视化管理。

10.测试运行监视控制

集中控制复杂分布式测试进程的开始、暂停、恢复、终止等操作。实现分布式功能测试工具和性能测试工具的同步、消息传递和集中控制,实现各种测试工具运行状态的监控,可为测试过程数据的采集提供接口。提供第三方运行监视功能,可根据用户权限,对测试运行过程中各结点端机桌面和用户換作过程进行监视。

11.被测对象管理

对被测对象进行标识、入库、出库、更动控制、版本管理、访问权限设置等,为系统提供独立手測试项目的被测对象存储、管理、维护环境。

12.用户管理

对用户进行注册、注销、登录等管理,维护用户基本信息,赋予用户角色,实现用户对被测对象、測试项目、测试工具、测试数据、测试流程、试计划、测试人员、測试日志等对象的访间权限规划,对用户的测试活动进行日志记录与存储。

13.测试配置数据采集存储

对测试配置数据,主要包括测试环境配置、测试工具配置、测试数据配置、测试人员配置,以及整个测试项目的总体策划规划文案等基本信息进采集、存储和管理,支持对测试配置数的深度分析,对支持测试配置数据的复用提供基本数据支撑。

14.测试过程数据采集存储

对测试过程中动态产生的各种测试数据和中间测试结果数据,例如测试用例以及用例执行情况的数据,进行采集、存储和管理。支持对过程数据的深度分析和测试报告的生成,对测试用例复用提供基本数据支撑。

15.测试结果数据采集存储

对测试结果数据,主要包括各软件測试工具生成的测试报告以及报告中的结论、各测试项的测试结论等数据进行采集、存储和管理。可对测试结果数据等进行深度分析,可支持对被测对象的综合分析。

16.测试数据综合分析

对同一被测对象不同版本、不同測试轮次的各神测试数据,例如测试配置数据、测试过程数据、测试结果数据的综合分析,可支持对被测对象总体评价报告的生成。可实现多个类似被测对象的综合数据分析,可支持优化同类被测对象的测试方案。

17.测试用例辅助设计支持

通过对多种被测对象、多个测试场景下不同测试用例的执行情况进行综合分析,形成典型被测对象在典型测试场景下的测试用例推荐,可辅助用户进行测试用例设计。

18.测试数据提取应用接口

该子系统是軟件测试云平台集中的对外数据发布的子系统,除了提供标准的数据库形式、文档形式的数据提取接口外,还提供外部应用软件可直接读取测试云平台数据的接口函数,用于支持后续测试大数据分析与应用。同时该子系统对提取测试数据的请求进行权限验证。

### 软件测试工具集

在测试云平台上，重点补充建设软件测试工具，包括软件源码分析测试工具、自动化功能测測试工具、逻辑覆盖率测測试工具等9种软件测试工具，详见表1。

表1拟补充建设的软件测试工具列表

|  |  |  |
| --- | --- | --- |
| 序号 | 软件测试工具 | 类别 |
|  | 自动化功能测试工具 | 非嵌入式 |
|  | 性能测试工具 | 非嵌入式 |
|  | 软件源码自动分析工具 | 非嵌入式 |
|  | 单元测试工具LDRA Testbed升级 | 非嵌入式、嵌入式 |
|  | 软件可靠性测试用例辅助设计工具 | 非嵌入式 |
|  | 代码质量测试工具 | 非嵌入式 |
|  | 数据标准规范检查工具 | 非嵌入式 |
|  | 逻辑覆盖率测试工具 | 非嵌入式 |
|  | FPGA混合仿真验证工具 | 嵌入式 |

### 软件测试资产库管理系统

软件测试资产库管理系统由测试软件研制过程管理分系统、軟件状态控制与管理分系统和软件产品管理分系统组成,分别服务于软件测试过程和工具集成开发的软件产品研制、软件测评、软件产品使用。如图3所示。

1.测试软件研制过程管理分系统

测试软件研制过程管理分系统具备项目管理、决策过程管理、需求管理、过程与产品质量管理、设计管理、配置管理、开发管理、系统管理和测试管理等功能,管理标准参照《GJB5000A军用软件研制能力成熟度模型》执行。

2.软件状态控制与管理分系统

软件状态控制与管理分系统具备软件信息管理、证书初始化与管理、源码比对与追踪、软件集成发布、软件问题管理、版本状态管控、系统管理和系统通讯录等功能,管理标准参照《GJB2725A-01测试实验室和校准实验室通用要求》执行。

3.软件产品管理分系统

软件产品管理分系统具备测试资源管理、被测软件管理、资产库统计知识库管理和系统配置与管理等功能,管理标准参照《GJB900C-2017质量管理体系要求》执行。

### 云桌面端机系统

云桌面端机系统由浏览器/服务器架构的端机和客户端/服务器的端机等两种端机组成,其主要组成示意图如图4所示。

浏览器/服务器架构的端机主要为用户执行门户网站式軟件测试时提供操作使用平台;客户端/服务器的端机主要为用户执行定制客户机测试环境式软件测试时提供软件测试操作使用环境。

## 典型应用模式满足情况

主要描述系统两种典型的应用模式：软件试验中最为复杂的分布式信息系统软件装备的试验模式、电子信息装备软件测评任务模式。软件测评任务中黑盒测试部分（指功能测试、性能测试、接口测试等）任务的应用模式与软件试验的类似。

### 分布式信息系统软件装备试验应用模式满足情况

分布式信息、系统软件具有节点多，组网复杂，应用模式灵活等特点，软件测试云平台可提供基于需求的精确计算资源、软件可定义的网络拓扑结构、模式可重用的软件测试环境，其试验环境构建如图5所示。

通过云计算技术形成计算、存储、网络等虚拟资源池，根据软件装备研制总要求和软件实际运行环境，申请计算、存储等运行资源，作为席位节点部署信息系统软件、云化软件测试工具等：根据装备软件运行模式和组网需求申请网络资源，依据信息系统软件组网模式软件定义网络拓扑结构；实现仿真试验系统的云化部署，作为试验配试资源，提供传感器、信息系统节点等仿真模拟；对于试验中需要，而又无法实现云化部署的真实装备、模拟工装等，可通过外部网络接入测试云平台内部，形成外部真实装备、云化仿真试验系统、云化测试工具与被测信息系统软件一体集成的综合测试环境。试验过程和结果数据由数据采集与分析分系统集中存储，并基于大数据技术进行测试数据的分析与评估。

**应答：满足。**

**分布式信息系统软件具有节点多、组网复杂，应用模式灵活等特点，我方提供的软件测试云平台可提供基于需求的精确计算资源、软件可定义的网络拓扑结构、模式可重用的软件测试环境。**

**在分布式信息系统软件装备的试验模式应用中主要的特点有以下几点：**

* **根据信息系统软件装备研制总要求和软件席位运行环境，生成测试运行的环境；**
* **根据信息系统的组网模式定义出网络拓扑结构，可实现不同网络性能的仿真；**
* **实现仿真试验系统的云化部署；**
* **无法云化的仿真系统可以通过外部网络接入到系统；**
* **实装真实设备可以通过外部网络接入到系统；**
* **试验结果数据可进行大数据的分析。**

**我方提供的系统能够通过云计算技术，形成计算、存储、网络等虚拟资源池，根据软件装备研制总要求和软件实际运行环境，申请计算、存储等运行资源，生成测试运行的环境；能够依据信息系统软件组网模式软件定义网络拓扑结构，可实现不同网络性能的仿真；可以实现仿真实验系统的云化部署；能够提供传感器、信息系统节点等仿真模拟；对于无法云化的真实装备、仿真系统，可以通过外部网络接入到系统；可以形成一体集成的综合测试环境。试验过程和结果数据由数据采集与分析分系统集中存储，并基于大数据技术进行测试数据的分析与评估。**

### 电子信息装备软件测评任务应用模式满足情况

在电子信息装备軟件测评任务应用模式下，软件云测试系统主要实现软件测试工具的云化应用，针对测试工具的可云化程度，主要分为三种应用方式：多用户并行使用方式、多用户分时使用方式、单用户登陆使用方式，如图6所示。

多用户并行使用方式：对于可完全云化的测试工具，每个云用户节点均可获取完整的使用权限，多用户可同时进行多任务的测试；

多用户分时使用方式：对于使用浮动 License云化的测试工具，同一时间只能有一个用户获取使用权限，实现 License竞争管理机制，多用户可分时进行多任务的测试；

单用户登陆使用方式：对于已有的单机软件测试工具，或不能云化的软件测试工具，静态安装于基于云平台的指定虚机上，采用登陆管理的方式实现单用户登陆使用。

**应答：满足。**

**在电子信息装备软件测评任务应用模式下，使用软件测试云平台系统主要有以下几个特点：**

* **主要针对软件开展配置项和系统级别的测试；**
* **测试工具和被测件运行环境主要以虚拟测试用机为主要支撑；**
* **测试过程主要按照GJB2725A附加技术能力要求的几个测试阶段进行；**
* **云化的测试工具调度需要考虑多用户并发及多用户使用的情况；**
* **多个软件同步开展测试时，可以构建出多套测试环境，但需要考虑License的资源限制的分时调度；**
* **测试时要按照完整的测试文档模板要求实现测试过程文档的生成。**

**我方提供的系统主要分为三种应用方式：**

* **多用户并行使用方式：对于可完全云化的测试工具，每个云用户节点均可获取完整的使用权限，多用户可同时进行多任务的测试；**
* **多用户分时使用方式：对于使用浮动 License云化的测试工具，同一时间只能有一个用户获取使用权限，实现 License竞争管理机制，多用户可分时进行多任务的测试；**
* **单用户登陆使用方式：对于已有的单机软件测试工具，或不能云化的软件测试工具，静态安装于基于云平台的指定虚机上，采用登陆管理的方式实现单用户登陆使用。**

**综上所述，我烦共提供的系统能够满足电子信息装备软件测评任务应用模式的要求。**

## 技术指标和使用要求满足情况

### 总体指标满足情况

a)可灵活构建软件测试环境，支持电子信息装备非嵌入式软件(含自主控软件)的软件试验和软件测评。

b)支持靶场保障条件建设装备中各类非嵌入式软件的软件测评，可支持典型FPGA软件的混合仿真验证。

c)能够涵盖的电子信息装备非嵌入式軟件的测试级别包括单元测试、部件测试、配置项测试和系统测试

d)能够涵盖的电子信息装各非嵌入式软件的测试类型包括静态分析、代码审查、功能测试、性能测试、接口测试、标准符合性测试、人机交互界面测试、强度测试、余量測试、边界测试、安全性测试、可靠性測试、恢复性测试、安装性测试、兼容性測试等测试类型。

e)能够同时部署开展的软件试验和软件测评任务数量不少于5个；单任务模式下，最大可支持不少于100个节点的指挥信息系统开展軟件试验和软件测评。

f)软件测试工具总体要求：软件测试工兵优先选用成熟的、行业认同度高的主流软件测试工具;单机版测试工具的 License不少于2个,至少可保证License在云测试平台使用,另一个 License在单机上使用；除FPGA混合仿真验证工具外，其他软件测试工具均能够与软件测试云平台集成,测试数据能与平台共享。

### 分系统指标满足情况

#### 基础硬件资源系统

构建小型私有云数据中心,为软件测试云平台、软件测试工具运行以及软件测试资产库管理系统提供所需的硬件环境,具体指标如下：

a)基于可扩展的模块化机房方式构建,模块化机房集成服务器、存储设备、网络设备、供电设备、空调设备、环境监测设备等,无需对机房环境进行大规模基础建没。

b)计算资源指标。7台计算节点服务器,基本配置不低子:2颗英特尔至强E5-2650v4处理器:4×32GDDR4 RDMM内存:2×2000GB通用硬盘SAS:支持RAID;4×10GE端口。

e)存储资源指标。6台存储节点服务器,基本配置不低于:2颗英特尔至强E5-2650v4处理器:4×32GDDR4 RDMM内存:10×2000GB通用硬盘SAS;2×800GB固态硬盡-SATA6Gb/s:支持RAID:4×10GE端ロ。

d)12U高性能一体化机絶模块指标。2个12U高性能一体化机箱,基本配置不低于:2块机框管理模块:14个风扇框;6块冗余交流电源:2块交换模块板:4块10GE光模块板;配套线缆及安装滑轨2个。

e)网络资源指标:2台万兆交换机,接口不少于48个;1台千兆交换机接口不少于48个。服务器与交换机以万兆接互连。

f)测试云平台配各UPS电源,外部断电情況下可支持系统连续运15min 。

g)其他指标:测试云平台外接串口不少于50路;具有视频切换与视频输出功能,实现对虚拟机界面的图像拼接和切换输出。

h)布线安装指标:包括机架内布线、操作席位布线、演示环境布线、云桌面端机布线等

i)机.柜主要技术参数包指:外形尺寸(高x宽x深)2000m×600mm×1350mm;机柜提供42U可用空间:静載不小于1500kg:前后风道通风设计

j)配置入侵检测測、漏润扫描、日志审计、防火墙、终端安全管理、网络防病毒等网络安全防护设备。

#### 软件测试云平台

软件测试云平台包括云基础分系统、测试运行支撑分系统和测试数据采集与分析分系统三部分,云基础分系统与测试运行支撑分系统和测试数据采集与分析分系统之间采用松耦合方式构建。云基础分系统统一管理系统资源,对计算资源、存储资源、网终资源等进行虚拟化,提供资源的按需分配与管理,并开放管理接口,满足軟件测试平台对系统资源的需求。

1.总体指标

a)平台可在集成的测试工具支持下完成电子信息装各非嵌入式软件的静态分析、代码扫描、单元测试、配置项测试、系统测试等測试任务。

b)平台具备同时开展多项测试任务的能力,同时支持的测试任务数量不少于5个,可支撑大規模分布式软件系统的集成测试。

c)可统一管理多种硬件资源,实现虚拟机的生成、部署和联网,提供虚拟机终止、挂起、存储和唤醒服务。

d)可对计算机网络进行虚拟化,实现软件定义网络功能。

e)可对虚拟机采用的操作系统、功能软件、硬件配额(CPU核心数、内存大小、硬大小)、网络环境和测试工具进行订单式的选配及安装,形成满足不同测试任务需求的测试环境集(测试计算机网络),并可对测试环境集进行整体备份和恢复。

f)支持不同测试工具间的测试数据的互联与共享,支持测试大数据分析。

g)可实现测试过程数据和结果数据的实时采集、管理和存储,可自动分析处理測试数据,形成测试报告,并可对测试报告进行定制。

2.云基础分系统指标

基础云平台统一管理系统资源,对计算资源、存傜资源、网络资源等进行虚拟化,提供资源的按需分配与管理,并开放管理接口,满足软件测试平台对云平台管理的集成要求。基础云平台应主要包括以下功能:

a)基础云平台应具有硬件资源可扩展、按需分配、弹性伸缩、负载均衡等云平台的基本功能。

b)支持 Docker容器功能,可实现测试工具、測试环境的轻量化部署。

e)具有虚拟机定制生成功能,文持 Linux、 Windows和 Kylin OS等多种操作系统的虛拟机生成,单个虚拟机支持TB级存储卷。

d)具有网络虚拟化功能,支持虚拟机浮动IP,支持跨网段虚拟路由配置能够在虚拟机节点间构建多级交换网络,支持虚拟机节点与外部网络间的互联。

e)具有大数据存储与管理功能。对測试数据进行存储管理,支持后续大数据分析功能升级。

f)具有镜像与各份管理功能。支持虚拟机镜像的生成、存储、使用、删除等管理,支持虚拟机集群管理,对虚拟机集群进行统一各份、标识、恢复、删除等管運。

g)具有安全防护功能,保证云平台的系统安全、数据安全。支持访问权限管理、防病毒管理、安全日志管理等。

h)支持加密狗授权软件的使用。支持在服务器USB接口接入加密狗,可在云平台生成的任意虚拟机上使用该软件授权(不同时使用具有串口虚拟化功能。可在云平台生成的虚拟机上实现串口虚拟化,虚拟机与虚拟机间,虚拟机与外部设备间可通过串回进行通信。

j)具有虚拟机界面监视功能。在不影响虚拟机操作员操作的前提下,可以将虚拟机操作界面切换至监视端显示。

k)具有用户管理功能。支持多用户共享资源,支持多用户按需动态分配资源,支持用户权限控制。

l)具有虚拟机管理功能。支持虚拟机的按需申请、自动生成,实现虚拟机的启动、挂起、关闭等操作,可对虚拟机运行状态进行监视,具有虚拟机配置开机执行脚本等功能。

m)具有可视化管理与运维界面,可对系统资源进行统一登记、标识、管理和状态监视,可单独实现对云平台的管理操作。

n)云基础及硬件管理功能。完成云计算主机群的安装、互联和配置,为上层的云计算提供硬件支撑。完成云主机操作系统的安装和系统计算、内存网终、存储资源的虚拟化功能提供云环境配置功能。

3.测试运行支撑分系统指标

测试运行支撑分系统主要对软件测试过程和实施进行管理,包括测试对象管理、測试环境构建、测试运行控制等,主要功能指标包括:

a)被测对象管理功能:建立被测对象生命周期管理数据库,实现基于用户角色的被测对象入库、出库、更动、版本管理功能,为系统提供独立于测试项目的被测对象存储、管理环境。

b)测试环境定制功能:提供测试环境可视化定制功能,具备操作系统、常用功能软件、测试工具软件的定制、初始化与部署功能。支持基于虚拟机的部署和可视化虚拟网络括扑规划设计,支持测试环境集的整体备份、恢复等管理功能。

c)测试运行管理功能:支持虚拟机的创建、删除、挂起、迁移、扩容等。

a)用户管理功能:实现被测对象、測试项目、测试工具、测试数据、澳测试流程、测试计划、测试人员、测试日志等对象的访间权限规划。对关键试活动进行日志记录与存储。

e)云平台软件工具适配功能:完成已有、新购及开源软件测试工具的云平台配。提供软文件、硬件Key、网终版 License的集中登记、管理,实现虚拟机与 License的适配;单机版测试工具的封装和面向分布式云环境的功能转换:B/S架构、CS架构测试工具的云环境配置功能

4.测试数指采集与分析分系统指标

a)具有数据搜集与存储功能。能够对测试配置数据、测測试过程数据、测试结果数据进行搜集和存储;可实现格式化数据和非格式化数据的存储与应用;具有大数据分析扩展接口,可为后续大数据处理提供支撑。

b)具有测试数据综合分析功能。具有用户可定义的软件评价体系,能够根据评价指标对测试数据进行综合分析,并以可视化方式显示分析结果,同输出评价分析报告。

c)具有测试用例辅墈设计功能。通过对测试需求进行分析,基于历史测试场景、测试用例、测试数据等己有信息,形成典型被测对象在典型测试场景下的测试用例推荐,可辅助用户进行测试用例效计。

5.性能指标

a)測试云平台对硬件资源进行管理,所管理的服务器模不小于13台,可扩展至50台以上;管理存储资源规模不小于200T,最大可扩展至不少于1000T。

b)测试云平台能够虚拟出不少于200个虚拟机(配置为2核CPU、4G内存、100G存储容量),支持自定义虚拟机资源配置。

c)支持管理200个以上的虚拟机镜像:创建50个虚拟机实例用时10min;完成单虚拟机整机各份时间小于5min。

#### 软件测试工具

1.自动化功能测试工具技术指标

a)提供图形化和关键字驱动测測试功能,为非编程技术人员提供图形化的关键字视图,为数据驱动测试提供熱悉易用的数据表。

b)提供测试流程模板(来自于实的流程指导),用户也可以自定义流程。

c)能够和测试管理平台集成,在应用开发生命周期中实现案例的管理和重用。

d)支持基于数据驱动的功能测试,可按用户自定义方式模拟生成功能测试所需的各种测试数据。

e)支持历史測试数据文件方式生成功能测试所需的测试数据,驱动被测软件自动运行。

f)支持典型分布式软件系统(或体系),例如指挥信息系统的自动化功能测试,支持规模不低于旅(团)级(或分布式节点数不小于24个)。

g)对于不断变化的应用,易于创建、执行和维护回归测试。

h)支持文档的自动生成,具有易用性和内置的环境支持支持运行的操作系统: Windows系列操作系统、 Linux操作系统等。

j)浏览器:支持IE、 Firefox等主流浏览器:其中IE浏览器版本不低于6.0SP1, Firefox浏览器版本不低于1.5。

k)支持对以下类型的应用程序进行自动化功能测试:标准 Windows应用程序,包括基于Win32API和MFC的应用程序、 Activex控件、 Visual Basic应用程序、Web页面。

2.性能测试工具技术指标

a)具有非侵入性的实时性能监视功能,可提供被测系统主要性能指标的监视。

b)可支持用户定义方式自动生成压力测试数据,支持对被测软件进行压力测试和稳定性、可靠性分析。

c)可测试分析被测软件运行时对CPU等硬件资源占用率、系统响应时间等关键性能指标

d)支持对以下类型的应用程序进行自动化性能测试:标准 Windows应用程序,包括基于Win32API和MFC的应用程序、 Activex控件、Web页面。

e)支持对典型分布式软件系统(或体系),例如指揮信息系统软件的整体性能测试,支持规模不低于旅(团)级(或分布式节点数不小于24个)。

f)具有测试脚本录制功能。

g)支持文档自动生成,具有易用性和内置的环境支持。

h)支持运行的操作系统: Windows系列操作系统、Linx操作系统等。

3.软件源码自动分析工具技术指标

a)至少可支持CC-+、Java、NET等3种语言的软件源码自动分析:NET语言的源码自动分析至少应支持检査C#、VB.NET、 ASP. NET源码的规范。

b)可以导入和导出一个函数、一个文件和整个工程的测试用例。

c)可以按“自动生成桩函数”、“使用原函数作为桩函数”和“自定义桩函数”三种方式生成桩函数。

d)内置一定数量的代码编程规范,提供图形化的代码编程规范的定制工具,支持用户自定义代码编程规范,提供自动化的代码编程规范检查功能。

e)支持对内置代码编程规范和用户自定义代码编程规的修改。

f)可图形化地实现代码走查,能检査未初始化而引用的变量和引用而未释放的变量,能査找空指针异常、数組越界、除零等锆误。

g)能自动执行白盒测試,确保代码中每一条独立的路径至少执行一次,支持圈复杂性指标的测试。

h）能根据某种质量模型评价代码的质量。

i)能够根据需要自动生成测试报告,并根据用户的需求对报告格式进行定制。

4.单元测试工具 LDRA Testbed技术指标

a)具有编程规则验证、数据流分析、控制流分析、表达式分析、接口分析、软件质量度量分析等功能。

b)可提供多神代码覆盖率分析,主要包括:语句覆盖、分支/判定覆盖、LCSAJ覆盖、过程/函数调用覆盖、分支条件覆盖、分支条件组合覆盖、修正条件判定覆盖、动态数据流覆盖等。

e)支持图形化显示功能,主要包括:柱状堅、流程图、调用图、 Kiviat图等的显示。

d)支持CC++、Java等语言。

e)支持的代码规则主要包括:电子信息装备软件尤其是信息系统装备软件领域代码规则集:以及国际主流代码规则集:并支持自定义新建代码规则集。

5.软件测试用例辅助设计工具技术指标

a)习导用户构建图形化的软件测试剖面不少于5种。

b)支持构建可靠性測试割面,例如客户割面元素不少于10个,用户面元素不少于20个,状态剖面元素不少于50，功能剖面元素不少于100个，操作剖面元素不少于100个。

e)可一次自动生成XML格式的软件测试用例数据最大不少于1万个。

d)生成包含1万个测试用例数据的文件时间不超过1min。

e)支持测试人员开展规范的软件测试剖面构建和测试用例设计工作。

f)支持用户、状态、功能、操作等需求要素的层次关系和概率信息描述,同时支持描述各妻素之间的转移关系和时序约束等信息,可充分反映软件运行过程中的各类场景和外部激励。

g)可依据软件测试剖面及外部接口取值分布特征,结合外部接口环境文件,自动生成海量XML格式的用例数据,用于软件功能测试。

h)支持运行的操作系统: Windows系列操作系统、 Linux操作系统等。

6.代码质量测測试工具技术指标

a)至少支持C/C++、Java、C#等3种语言。

b)支持代码缺陷裣测、安全漏洞检测、度量分析、架构分析、质量趋势分析、架构约東分析等功能。

c)支持 Windows、 Linux、 Kylin OS等操作系统。

d)不需要写测试月例就能找出程序中的关鍵bug,包括程序动态运行时发生的错误(如指针越界、内存泄露、溢出、数组越界、变量未初始化等),并且能够准确定位程序出错的位置。

e)支持主流编码规则。

f)支持 Mccabe复杂度、 Halstead程序度量、代码行数、继承数、循环数等各种基本度量。

g)支持缺陷按级别、类型、存在状态、处理状态的分析统计。

h)支持自动生成缺陷报告、缺陷报表査看和导出、多样化的结果统计(bug分类统计、bug状态统计、函数复东度、多神缺陷过滤标准等)。

7.数据标准规范检测測工具技术指标

a)可支持 Oracle、 SQL Sever、 MYSQL、达梦、金仓等多种关系型数据库

b)支持多种数据标准

c)单检查规则测试时间:在100M局域网环境中,在标准库数据表中数据小于1000个时,单检査规则测试每分钟检登不少于100张数据表。

d)测试报告导出时间:在问题数小于1000个时,测试报告导出时间小于3分钟。

8.逻辑覆盖率测试工具技术指标

a)支持C、C++语言的插桩

b)插桩膨胀率控制在15%以内

c)可以快速定位系统性能的瓶颈所在,即时非抽样监控超过12万个C/C++函数、150万行代码。

d)支持4种覆盖率:块覆盖、语句覆盖、判定覆盖、MC/DC

e)支持网口、串口、CAN、JTAG等测试接口。

f)支持离线查看测试结果。

g)在测试过程中,可以实时更新程序的各种语句、判决、MCDC的盖率、函数的流程及调用关系、内存、A- B Timer。

9.FPGA混合仿真验证工具技术指标

a)支持 Verilog、VHDL、 System Verilog、 Systemc、 Open Vera、混合语言仿真。

b)支持主流FPGA芯片厂商的主要器件的仿真,具有较好的RTL与门级仿真性能。

e)支持性能分析功能,能够分析性能瓶颈。

d)支持仿真时访问VHDL或 Verilog混合设计中的下层模块的信号,便于仿真调试。

e)支持仿真恢复与继续功能。

f)提供设置复杂断点、信号眼踪等调试功能。

g)支持SVA等断言的调试。

h)支持代码覆盖率统计功能。

i)支持在通用的 Linux操作系统下运行。

#### 软件测试资产库管理系统技术指标

1.总体指标

a)能够与软件测试云平台实现功能集成,数据能与测试云平台共享、主要管理功能可通过云平台实现统一调度。

b)具备项目管理、决策过程管理、需求管理、软件产品过程与质量管理、设计管理、配置管理、开发管理、系统管理、测试管理、证书初始化与管理、源码比对与追踪、软件集成发布、软件题管理、版本状态管控、测试资源管理、被测软件管理、资产库统计和知识库管理等功能。

c)对外预留数据和功能的开放接口,支持软件试验和软件测试的大数据分析评估。

2.主要功能指标

(1)测试软件研制过程管理分系统

a)测试软件研制过程管理分系统具备项目管理、决策过程管理、需求管理、过程与产品质量管理、设计管理、配置管理、开发管理、系统管理和测试管理等功能,管理标准参照《GJB5000A军用软件研制能力成熟度模型》执行。

b)项目管模块能满足软件开发组织对软件项目在进度、工作量、质量、资源等进行集成管理的需要。能够基于平台进行项目策划、资源分配、进度成本汇报眼踪、测量与分析、风险管理、质量保证;能够基于平台提供的各种视图,了解项目的状况、问题点和各项综合统计指标。

c)决策过程管理模块支持对项目间题进行决策分析正式评价过程的管理,包括:判断问题是否需要采取正式评价,如需要正式评价,则标识间题各选方案,按照评价准则对备选方案进行正式评价,记录正式评价过程数据远中最终方案并在风险管理中识别相应的风险。

d)过程与产品质量管理模玦支持过程和产品质量保证任务的策划、执行、跟踪。支持从组织资产库中导入常用的裣查单,支持项目中检査项的删减；支持在审核时发现的不符合项的在线处理流程,支持不符合项的汇.总功能;支持生成质量保证的阶段报告、里程碑报告及质量总结报告。

e)配置管理模块应满足相关标淮和规范,以软件配置管理为核心,同时支持需求管理和缺陷追踪的功能,具有软件配置策划管理、变更控制、版本管理、产品依赖关系管理、产品一致性管理、軟件问题追琮管理、软件配置状态审计管理等功能。

f)需求管理模块可管理项目的产品和产品部件的需求,并标识这些需求与项目的计划和工作产品之间的不一致性。週过需求管理系统可管理项目接收的或产生的所有需求,以及组织强加于项目的需戒。需求在项目期间演化时,通过变更管理系统管理需求的更改,同时在需求分析的过程中,维护需求和工作产品的双向可追溯性。即完整的需求管理流程应包括需求的建立需求的分析、需求的变更及需求的双向性追溯等。

g)设计管理模块能够实现对设计阶段的配置项集中管理,实现软件研制设计阶段配置项的出库、入库、版本发布、版本管理。

h)开发管理模块可以有效对软件产品研制开发阶段的源代码实现开发库、受控库和产品库的三库管理模式,支持源代码的出库、入库、版本发布版本管理。

i)测试管理模块符合相关标准中的管理要求,能够满足软件产品测试的业务需求。可为测试人员提供测试用例辅助设计和测试用例管理功能,可辅助测试人员梳理测试需求、制定测试计戈划、设计测试用例,并依据测试结果分析产生出测试报告,支持测试工作的可移动模式,应能够在 Windows、 Linux、麒麟操作系统环境下独立安装;支持向导服务方式引导测试人员按照软件测试各个阶段的要求开展相应的工作;支持测试用例自动推荐,能够对历史测试用例进行分析。在测试用例辅助设计过程中,能够根据当前项目信息、测试功能点和该功能点的测试需求,自动推荐出合理的测试用例。

j)系统管理模块应满足保密“三员”管理要求,即系统管理员、安全保密员(权限角色管理)、安全审计员(审计系统管理和安全管理行为)。满足涉密信息系统的管理,系统管理员负贡系统的日常管理工作,安全审计员负责系统使用日志的管理,安全保密管理员负责系统的安全措施设定。

k)能够与软件测试云平台集成,数据能与平台共享。

(2)软件状态控制与管理分系统

a)软件状态控制与管理分系统具各软件信息管理、证书初始化与管理源码比对与追踪、软件集成发布、软件问题管理、版本状态管控、系统管理和系统通讯录等功能,管理标准参照GJB2725A执行。

b)软件信息管理模块能够依据软件所属的装各谱系、装备、分系统对软件信息进行管理,维护软件版本变化,管理軟件各版本对应的可采集资源。通过树形结构和列表的形式,对装备谱系、装备、分系统、软件以及软件版本做统一管理,确定软件状态、记录软件变更历程、采集软件版本资源并确定其版本状态。

c)源码比对与追踪模块能够对不同软件源代码进行比对分析,形成比对报告和统计报告,计算版本相似度。可自动建立軟件版本变更和变更需求之间的追踪关系;分析不同软件版本间的差异,统计差异文件,形成文件差异清单,建立和维护软件版本和变更需求间的追踪关系。

d)软件间题管理模块能够管理软件版本下存在的间题,实现问题列表的导入、导出。记录软件版本存在的问题,为软件维护提供依据。

e)系统管理模块能够实现组织、人员管理;实现角色设置和权限定义实现系统使用过程中的基础数据和类型定义,包括:问题类型、资料类型等。

f)证书初始化与管理模块能够管理用户,生成和带有用户唯一身份标识的公、私钥证书,作为使用状态管控应用分系统的身份识别。

g)软件集成发布模块能够对应生成软文档的安全文库(单一文件),其逆向过程为解包,由安全文库还原出原始软件文档。

h)版本状态管控模块能够对应生成软件状态文件,以及根据軟件状态对软件实施验证鉴别。

i)系统通讯录模块在用户安装客户端后,可获取联系单位或联系人员的公钥信息,生成系统通讯录,方便安全文库的共享和状态文件会签。

i)能够与软件测试云平台集成,数据能与平台共。

3)软件产品管理分系统

a)软件产品管理分系统具备测试资源管理、被测软件管理、资产库统计知识库管理和系统配置与管理等功能,管理标准参照G.JB9001C执行。

b)测试资源管理和被測软件管理模块能够实现软件资产实体的接收、入库、领用及处置流程管理,并结合软件状态管理工具,实现对软件版本的有效控制,可视化展示软件迭代曲线。

c)测试资源管理模块能够支持软件资产在软件全生命周期的相关信息的存储、管理和维护,对软件测试资产库中的信息发布、废弃、修订,提供增、删、改、査等操作,并根据授权管理实现资产库的浏览和操作权限的控制。

d)资产库统计模块可对软件资产信息进行分类统计,实现软件资产生命周期可视化展示,可视化内容丰富、形式多样。

e)具备知识库管理功能,可支持对各类文档模板的自定义上传,自定义分组管理,支持建立文档模板与标准过程的关联关系。

f)系统配置与管理模块能够实现对标程及过程包含的活动的定义管理,包括标准过程定义、发布、废弃、修订和历史记录;能够实现对软件资产管理制度和质量管理体系等标准、流程、规范的存储与维护更新;可分类建立风险库、问题库、测量库、配置管理库、过程评估与改进库、项目归档库等专项资产信息的统一管理平台:能够实现自定义设置多层次资产目录结构,支持对资产目录的维护:支持自定义资产状态、资产来源、資产类别等选项字典。

g)系统基础管理功能模块具有用户管理、组织管理、权限管理、三员管理、审计管理和安全设置等功能。

h)能够与软件测试云平台集成,数据能与平台共享。

3.性能要求

a)辅助生成测试扱告的时间小于3分钟,且有实时显示报告生成进度。

b)软件资产生命周期的可视化效果生成时间在1秒以内。

c)支持多用户并发使用,同时最大用户数不小于50用户。

### 使用要求满足情况

#### 环境适应性要求

a)硬件设备存储环境温度:-10℃~+50℃

b)硬件设备工作环境温度:0℃~35℃(空调开)。

c)硬件设备工作环境相对湿度:30%~-80%(+25℃）。

应答：满足。

我方提供的硬件设备存储环境温度

#### 可靠性要求

任务可靠性要求:单测试任务连续测试工作时间不小于48h。

#### 维修性要求

硬件设备平均故障修复时(MTTR)≤30min，关键件和重要件可修复、可更换。

#### 保障性要求

硬件设备采用市电供电，电压：AC220X(1ェ10%)V、频率:50X(1士5%)Hz。

## 质量控制满足情况

### 研制过程控制

合同生效之日起，研制单位在总体设计方案评审等重要节点前，应以书面形式提请组织会议评审。

**应答：满足。**

**我方将自合同生效之日起，在总体设计方案评审等重要节点前，以书面形式提请组织会议评审。**

### 验收

按验收地点的不同一般分厂所测试和靶场测试两个阶段。招标单位组织编写验收大纲。装备研制完成后,研制单位以书面形式提交验收申请。招标单位按照验收大纲开展验收工作。

**应答：满足。**

**我方将在装备研制完成后，以书面形式提交验收申请。**

#### 厂所测试

招标单位依托研制单位进行厂所测试，成立联合测试组。研制单位与招标单位讨论商定，拟制厂所测试实施细则。测试完成后联合测试组编制厂所测试报告。厂所测试所需保障条件由研制单位提供或协调解决。

**应答：满足。**

**我方将配合招标单位进行厂所测试，成立联合测试组。我方将与招标单位讨论商定，拟制厂所测试实施细则。测试完成后联合测试组编制厂所测试报告。我方将提供或协调解决厂所测试所需的保障条件。**

#### 验收转阶段会

厂所測试报告编制完成后，招标单位按照验收大纲要求组织召开转阶段会议，研制单位配合组织会务工作，形成评审意见。评审通过后，转入靶场测试阶段。

**应答：满足。**

**我方将在厂所測试报告编制完成后，召开转阶段会议阶段，配合招标单位进行组织会务工作，形成评审意见。评审通过后，转入靶场测试阶段。**

#### 靶场测试

转入靶场测试阶段后，研制单位积极配合招标单位完成靶场测试。

**应答：满足。**

**我方将在转入靶场测试阶段后，积极配合招标单位完成靶场测试。**

### 移交

设备经验收合格,由招标单位组织设备初步移交。设备初步移交后,试用满一年或参加一次试验训练任务后,性能稳定且无遗留问题,招标单位组织办理正式移交手续。

**应答：满足。**

**我方将在设备验收合格后，组织初步移交。设备初步移交后，试用满一年或参加一次试验训练任务后,性能稳定且无遗留问题，我方将配合招标单位组织办理正式移交手续。**

### 质保期

正式移交后,免费质保2年。

**应答：满足。**

**我方将提供免费质保2年。**

### 其他

研制过程中，已审定的技术方案若确需调整的，须经招标单位认可后实施。

合同生效之日起，研制单位每季度向招标单位提供研制情况和质量管理通报，如出现重大间题及关键岗位人员变动，研制单位应10天内通报招标单位。

对于大型或重要引进、外协部件，研制单位应严把质量关，招标单位参加合同签订和设备验收。

招标单位可根据需要，定期对设备的研制进展情况进行检查，研制单位应予以积极配合。

分系统研制完成或测试工具研制完成后，须进行测试，由中标方组织，招标方参加。

**应答：满足。**

**在研制过程中，如已审定的技术方案若确需调整的，我方将提交经招标单位认可后实施。**

**我方将自合同生效之日起，每季度向招标单位提供研制情况和质量管理通报，如出现重大问题及关键岗位人员变动，我方将在10天内通报招标单位。**

**对于大型或重要引进、外协部件，我方将严把质量关，招标单位参加合同签订和设备验收。**

**招标单位可根据需要，定期对设备的研制进展情况进行检查，我方将予以积极配合。**

**分系统研制完成或测试工具研制完成后，我方将组织进行测试，并要求招标方参加。**

## 设备数量与研制周期满足情况

设备套量:1套。

研制周期:合同签订后30个月完成厂所测试，具备甲方验收条件。

**应答：满足。**

**我方交付的系统设备套量为1套。**

**我方系统研制周期定为30个月完成厂所测试，具备甲方验收条件。**

# 项目建设总体方案

## 项目的目标、指导思想与原则

### 项目的目标

利用先进成熟的云计算、大数据技术建成软件测试基础云平台，集成主流的非嵌入式测试工具和典型FPGA软件测试工具，实现各测试工具的云化集成和数据共享，实现对软件试验和软件测试全过程的统筹规划和集中管理，满足不同自主可控程度的非嵌入式软件的代码分析、单元测试、部件测试、配置项测试和系统测试需求，满足典型FPGA软件的混合仿真验证需求，形成电子信息装备非嵌入式软件和典型FPGA软件的软件试验和软件测评能力。

系统建成后将为电子信息装备软件试验和测试提供统一的非嵌入式软件、嵌入式软件和FPGA软件测试平台和测试服务，使得测试基础硬件资源和测试工具得到充分有效的利用和调度，使得构建复杂的试验和测试环境成为可能，为保证电子信息装备软件的质量和系统的应用提供有效的验证试验环境。

### 指导思想

建设的总体指导思想是“立足现实、瞄准前沿、采用成熟技术、充分利用现有资源、框架标准先行”，保障建设成果持续可用和先进可靠。

通过本项目建设，建成软件云测试平台、主流的非嵌入式软件测试工具、软件测试全过程规划管理系统，实现主要软件测试工具的数据集成。

（1）先进的总体架构

采用“高性能计算资源+成熟云基础平台+灵活云测试服务+高质量软件测试工具”的总体技术架构，确保平台在当前和今后相当长时间里都能满足业主单位的装备试验与软件测试任务需求。

（2）成熟的技术应用

在云基础平台、软件测试工具的选用方面，优先考虑成熟的、行业认同度高的云基础平台以及主流软件测试工具，确保项目建设完成后系统的成熟、可靠。

（3）充分的资源利用

在平台设计时充分考虑对当前已有硬件资源、软件工具的利用，将其进行云化改造和适配；同时，对于新购置的软件测试工具均要求其能够与软件测试云平台进行集成；现有工具改造以及新工具的适配，确保项目中所使用的软件测试工具都能在测试云平台的调度下统一使用，使起发挥出更大的利用效能。

（4）标准、框架先行，持续集成

按照电子信息装备软件试验和软件测评需要，在分析各类测试工具特点的基础上，提出测试平台的有关标准，建设云平台框架，再分析测试需求，认真选购所需测试工具，持续集成所需的各类测试工具。

### 设计原则

为保证云测试平台建设的满足电子信息装备软件试验和软件测评需要，使得系统建设的指导思想得到落实，系统建设遵循标准与规范性原则、先进性和成熟性原则、稳定性与可靠性原则、易用性与方便性原则、安全性和保密性原则。

（1）标准化与规范化原则

云平台建设遵循国家和军队信息系统建设的有关标准和规范，遵循XML、JDBC、HTTP、SSL等业界主流的信息系统开发标准，同时建立测试云平台集成标准和平台框架，以便于相应的软件测试工具集成和新研测试工具的规范化。

（2）先进性与成熟性原则

在系统设计时，充分利用先进和成熟的技术来满足建设的需求，同时把科学的管理理念和先进的技术手段紧密结合起来，提出先进合理的电子信息装备试验和软件测试业务流程。系统将使用成熟的技术手段和标准化的产品，使得系统具有较高性能，符合当今技术发展方向，确保系统具有较强的生命力，有长期的使用价值。

（3）稳定性与可靠性原则

系统采用稳定可靠的硬件系统、采用成熟的基础云软件系统，在各个环节都考虑到故障诊断、恢复和容错能力，并在安全体系建设、复杂环节解决方案和系统切换等各方面考虑周到、切实可行，建成的系统确保安全可靠、稳定性强，把各种可能的风险降到最低。

（4）易用性与方便性原则

云测试平台主要面向测试人员使用，因此软件必须灵活易用，既能支撑简单的测试应用，也能应对复杂的测试应用，并且在使用过程中尽量做到简单易用，以支持测试环境、测试设计和测试执行的快速实现。在使用时，系统提供统一的界面风格，为每个用户群以及每个用户提供一个一致的、个性化定制的和易于使用的操作界面。

（5）安全性和保密性原则

在系统设计时，要将安全性放在首位，配置入侵检测、漏洞扫描、防火墙网络安全防护设备，确保基础设置的安全性；在系统的各个层次对访问进行控制，设置严格的操作权限，并充分利用日志系统、健全的备份和恢复策略增强整个云测试系统的安全性。

## 系统的总体架构设计

系统采用“高性能计算资源成熟云基础平台+灵活云测试服务+高质量软件测试工具”的技术架构，遵循云计算与云服务的技术体制，形成分层的技术架构，如下图3-1所示。系统主要由以下五层组成：

* 基础硬件设施层
* 云基础设施层（Infrastructure as a Service，基础设施即服务）
* 测试支撑平台层（Platform as a Service，平台即服务）
* 测试应用层SaaS（Software as a Service，软件即服务）
* 云桌面端机层

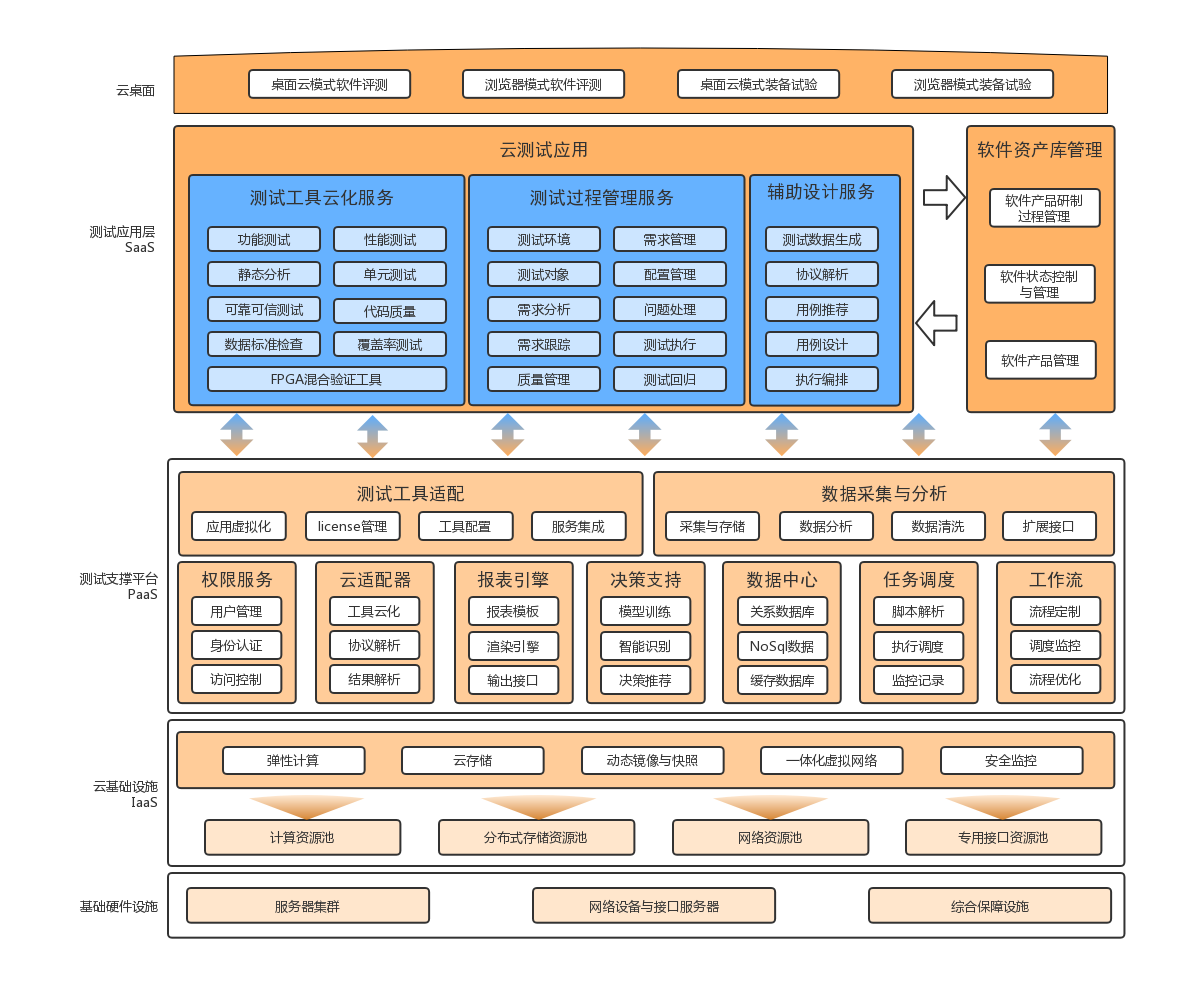


图 4.1‑1云测试平台系统架构

通过五层的架构设计，确保了系统具有较高的架构先进性，符合云平台与云计算的结构规范，确保系统能够满足非嵌入式软件测评需求、指挥信息系统软件全系统测试需求、典型FPGA软件的混合仿真验证需求和软件资产管理需求。

### 基础硬件设施层

基础硬件设施层为整个测试云计算平台提供了强大的计算能力，超大的存储空间，高速的网络带宽为用户部署运行各种应用服务提供硬件支撑，在该层中用户获取的不是信息而是计算和存储资源能力。

基础硬件设施层主要由高性能服务器、网络交换机、用户终端机、精密空调、网络防护设备、共享办公设备等组成，分别形成云服务器集群、网络与接口分系统、保障分系统。

### 云基础设施（IaaS）

IaaS层是测试云平台中硬件部分和软件部分的集成通道，为测试云平台提供虚拟化基础资源管理服务，主要实现资源的弹性计算、负载均衡、动态迁移和按需供给等功能。

IaaS层采用成熟的虚拟化技术，将硬件设施虚拟化为计算资源池、存储资源池、网络资源池和专用接口（Usb、串口）资源池；并为上层系统提供各类资源池的使用接口服务，主要包括以下服务：

1. 弹性计算服务

弹性计算服务主要实现对虚拟机的管理。通过配置镜像类型、CPU资源、内存资源、存储资源、网络资源等参数后可快速按需创建虚拟机；虚拟机运行过程中，各类资源由IaaS平台动态分配调度，当虚拟机所在的物理服务器出现故障时，虚拟机会动态迁移至满足资源需求的其它物理服务器上。

1. 云存储服务

云存储服务采用分布式存储架构，通过将物理服务器中的本机存储虚拟化为统一的存储资源池，为云平台提供块存储和文件存储两类服务。通过采用RADOS（A reliable, autonomous, distributed object storage）等技术，实现存储的无限扩展和冗余备份。

1. 镜像和快照服务

镜像服务为测试云平台所需的各类操作系统镜像和测试工具镜像提供统一的管理入口，通过使用系统分层技术，可动态组合操作系统镜像和测试工具镜像，灵活构建各类测试试验环境。

快照服务可在虚拟机运行的同时快速创建多个快照，并可动态恢复至任一指定快照，非常适合测试环境复现和环境配置的增量备份。

1. 虚拟网络服务

虚拟网络服务通过软件对虚拟机使用的虚拟网络和物理服务器连结的物理交换机进行统一配置管理，通过Genericsswitch插件动态配置物理交换机中的VLan，从而形成软硬件一体的L2虚拟网络。同时采用虚拟路由和交换技术，为测试云平台提供L3虚拟网络运行环境。

1. 安全和监控服务

安全服务由Authentication（认证）、 Token（令牌）、Catalog（目录授权）和 Policy（安全策略）四个方面的核心服务组成，为测试云平台提供统一的身份认证和访问控制实现机制。

监控服务将测试云平台运行过程中的各类事件和时序数据进行统一记录和处理，并可根据设定的条件进行报警。

### 测试支撑平台（PaaS）

PaaS层由支撑平台和测试运行平台两部分组成，其中支撑平台为云平台应用提供通用的基础功能服务，测试运行平台则提供测试工具适配和测试数据处理功能。

（1）支撑平台

* 权限服务

为上层测试应用提供基于角色的访问控制（RBAC）服务。

* 云适配器

云适配器为测试云平台提供统一的调度服务接口，并提供测试过程中输入输出数据和协议的解析引擎。

* 报表引擎

使用数据中心内的数据和用户自定义模板，动态渲染输出各类报表。

* 决策支持

使用经机器学习训练而来的自然语言处理模型，智能识别被测对象需求文档，并通过人工智能算法自动推荐测试用例。

* 数据中心

数据中心负责测试云平台运行过程中的数据集中存储，由关系数据库（mySql）、NoSql数据库（mogodb）和缓存数据库（redis）三类数据库构成。

* 任务调度

提供对自动化测试任务的脚本解析和测试任务执行调度，并对执行过程中的状态参数进行监控和记录。

* 工作流引擎

为上层管理应用系统提供流程定制编排和流程执行服务，并可对历史执行记录进行统计分析和优化推荐。

* 测试工具适配

针对不同使用模式的测试工具，使用相应的虚拟化技术，实现测试工具在云端的集成使用和license共享。

* 测试数据采集与分析

提供对测试的配置数据、过程数据和结果数据采集存储服务，并将数据转换为符合上层应用和下层引擎使用要求的格式。

### 测试应用层（SaaS）

测试应用层由云测试应用和软件资产库管理系统两部分构成。其中云测试应用由云化测试工具服务、测试过程管理服务、测试辅助设计服务组成；软件资产库管理系统提供软件研发全生命周期的管理功能，通过与测试云平台的集成实现测试数据、软件研制过程、软件状态管理和软件产品管理的全流程过程管理。

云测试应用是测试云平台的核心业务功能，提供测试项目全生命周期的云端应用服务，主要由以下三类应用组成：

1. 测试工具云服务

提供自动化功能测试、性能测试、数据标准规范检查、源码自动分析、代码质量测试、单元测试、逻辑覆盖率测试机FPGA混合仿真验证等云端服务。

1. 测试过程管理服务

提供测试项目实施过程中的需求管理、测试环境开设、测试对象管理、配置管理、需求分析、问题归并、需求跟踪、测试执行、质量管理、测试回归等全生命周期管理功能。

1. 测试辅助设计服务

提供测试数据生成、协议解析、用例设计、用例智能推荐、执行编排等辅助设计等服务功能。

### 云桌面端机系统

云桌面端机是测试云平台的用户使用入口，业务人员通过客户端机中的浏览器或虚拟桌面使用测试云平台提供的测试应用服务，开展软件评测和电子信息装备软件的试验业务。云桌面端机系统组成主要包括已有的计算机和新购置的云桌面端机。

## 系统组成及功能

按照3.2节系统的总体架构设计，系统组成分为硬件系统、软件系统两大部分，见图22所示。

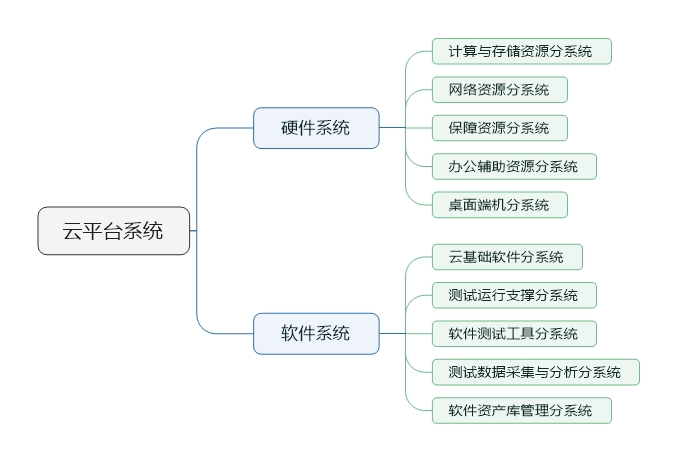
**

图4-2 系统的组成

硬件系统为五大硬件分系统：

* 计算与存储资源分系统
* 网络资源分系统
* 保障资源分系统
* 办公辅助资源分系统
* 桌面端机分系统

软件部分也分为五大软件分系统，分别为：

* 云基础软件分系统
* 测试运行支撑分系统
* 软件测试工具分系统
* 测试数据采集与分析分系统
* 软件资产库管理分系统

### 计算与存储资源分系统

计算资源与存储资源分系统由13台华为高性能服务器组成，其中计算资源7台服务器构成计算节点，存储资源由6台服务器构成存储节点。

计算资源主要为全系统提供CPU运算的功能，可以满足200个以上的虚拟机的计算需求，每个虚拟机的最小配置如下：

* CPU 核心数2核
* 内存 4G
* 存储容量 100G

存储资源主要为全系统提供云数据存储的功能，可管理的存储空间200T以上。

### 网络资源分系统

。。。请往下写其余三个分系统的功能

。。。。

## 系统的部署方案

测试云平台的系统部署方式，如图4.2-1所示。

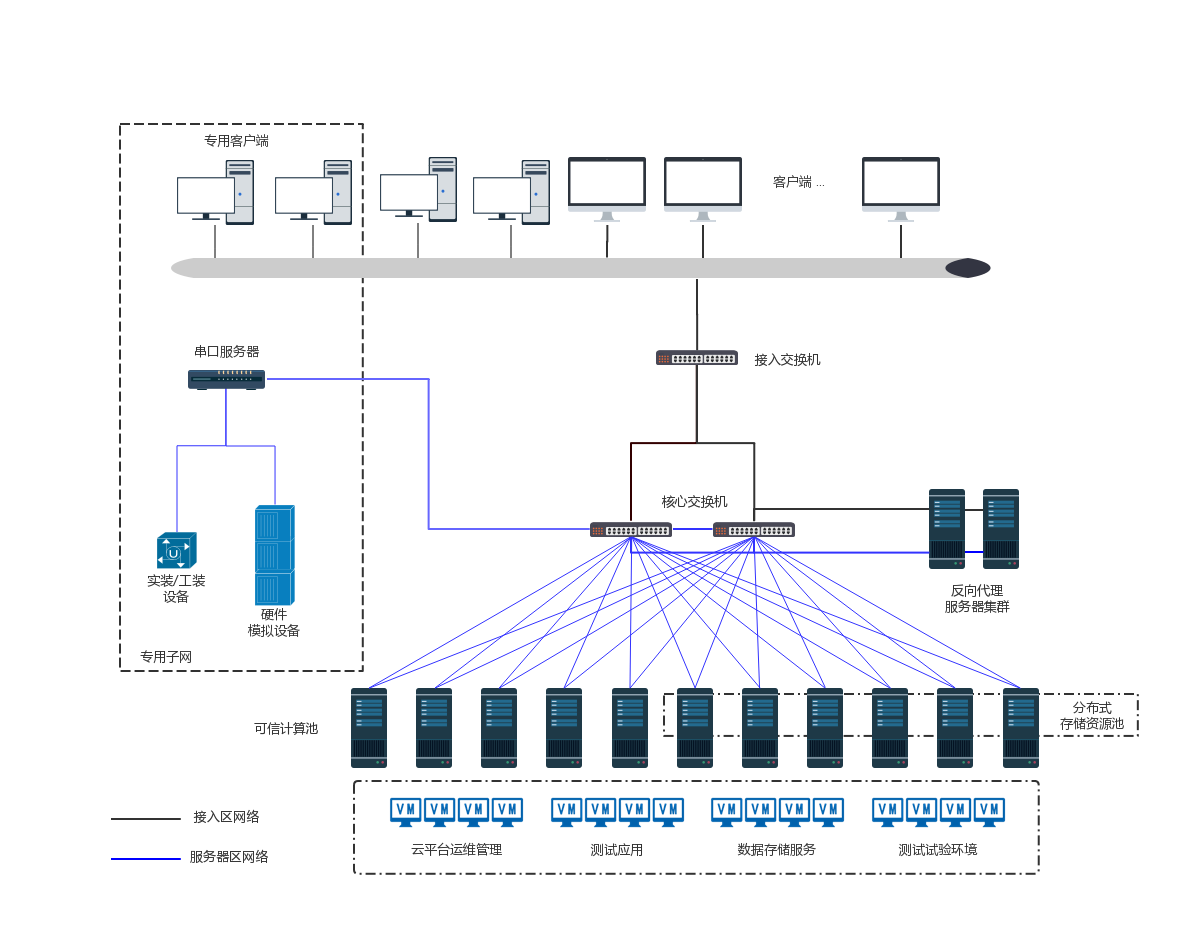


图 4.2-1云测试平台系统部署

测试云平台的物理部署方式分为接入区网络和服务器区网络两部分，接入区网络由1台接入交换机连接各个客户端和核心交换机；服务器区网络由2台可堆叠核心交换机和11台物理服务器连接而成；为保证云平台安全性，在接入区网络和服务器区网络边界部署2台互为热备的反向代理服务器作为测试云平台的访问网关。

为满足部分不可云化资源的接入需求，在服务器区部署1台串口服务器，用于连接外部设备；并通过软件定义的专用子网，将外部真实装备与云平台进行一体化集成，从而构建出专用设备的综合测试环境。

服务器区的13台物理服务器组成了测试云平台的核心服务器集群，其中7台物理服务器组成可信计算池，提供云平台运维管理和测试应用部署所需的服务器资源；另外6台服务器部署分布式存储系统，为测试云平台提供数据存储服务和测试环境管理服务。

## 系统使用的业务流程

系统基本使用的业务流程分为接收被测系统、测试需求分析、测试策划、测试设计与实现、测试执行、测试回归、测试总结七个阶段，软件测试流程示意图如图4.3-1。



图 4.3‑1软件测试流程示意图



图 4.3‑2系统使用流程图

系统基本使用流程分为接收被测系统、测试需求分析、测试策划、测试设计与实现、测试执行、测试回归、测试总结七个阶段，软件测试流程示意图如图4.3-1，系统按阶段使用流程如图4.3-2，详细描述如下：

### 接收被测系统

接收的被测系统包括被测系统源代码、被测系统可执行程序、被测系统文档（包括软件研制总要求、软件需求规格说明、软件设计说明、软件用户手册、软件接口协议等），接收到的被测系统被存储到软件产品库，同时测试过程管理系统对被测系统出库入库做配置管理。

### 测试需求分析阶段

a.测试内容需求分析

测试过程管理系统进入到测试需求分析阶段，测试人员选择此次的测试级别后，辅助设计服务模块从产品库中调取软件需求规格说明文档，提取测试需求后生成测试项存入测试过程管理系统，并自动适配对应的测试类型，测试人员可对提取的测试类型、测试项进行修改调整或确认后生成正式测试需求存储到数据库。

b.接口数据需求分析

辅助设计服务模块从软件接口协议说明文档中提取接口交互数据需求，包括接口类型、内容、格式、源地址、目的地地址等信息，生成接口数据需求，测试人员可修改调整或确认后生成正式测试环境需求存储到数据库。

c.系统运行业务流程需求分析

辅助设计服务模块从研制总要求和需求规格说明中，提取系统运行典型场景流程，生成运行流程需求，测试人员可进行修改调整或确认后生成正式系统运行流程需求，并作为测试项存入数据库。

d.测试环境需求分析

辅助设计服务模块从需求规格说明文档中运行环境部分提取被测件运行环境所需要的操作系统及版本、硬件配置及台位数量、支撑软件、外部交互接口类型等信息，生成测试环境需求，测试人员可修改调整或确认后生成正式测试环境需求存储到数据库。

e.测试工具需求分析

辅助设计服务模块从需求规格说明文档中提取被测系统测试类型要求、操作系统、开发语言、开发平台、测试环境需求等信息，自动适配需要的白盒测试工具和黑盒自动化测试工具，生成测试工具需求，测试人员可修改调整或确认后生成正式测试工具需求存储到数据库。

### 测试策划阶段

a.编写测评大纲及进行评审

测试项目负责人根据提取到的测试需求内容，对测评大纲测试内容与方法、测试环境、测试进度、测试结束条件、配置管理、质量保证及风险分析等进行进一步完善后，生成测评大纲。

之后通过网上评审的方式，组织专家对测评大纲内容进行评审，专家给出评审修改意见后，按意见进行修改完善，并根据修改意见对测试需求进行调整。

b.生成测试环境及部署被测系统

辅助设计服务根据测试环境需求和测试工具需求为测试云平台所需的各类操作系统镜像和测试工具镜像提供统一的管理入口，通过使用系统分层技术，动态组合操作系统镜像和测试工具镜像，构建系统需要的测试试验环境并部署被测系统，并将测试环境配置信息存储到数据库。

云适配器根据接口数据需求，调取测试工具Etest，根据接口数据需求，生成系统所需要的输入数据，并将数据环境配置信息存储到数据库。

### 测试设计阶段

辅助设计服务模块调取被测对象需求规格说明文档、设计说明文档、用户手册文档，通过决策支持经验库，采用智能算法自动推荐测试用例，测试人员对推荐的测试用例进行修改完善后求存储到测试过程管理系统，再生成测试说明文档。

辅助设计服务模块调取测试工具需求分析结果，自动解析后通过支撑层测试工具适配，调用功能测试、性能测试、接口测试等对应的黑盒自动化测试工具，根据已设计好对应的测试用例，录制自动化执行脚本，由测试人员进行调整完善后，根据系统运行流程需求，由生成对应的系统运行流程场景。自动化执行脚本和运行流程场景配置存入数据库。

### 测试执行阶段

a.白盒测试部分（静态分析、代码审查、逻辑测试等）

测试人员根据测试工具需求分析结果，调用对应的测试工具（TestBed、C++ Test、PolySpace、Klocwork等）后，通过云适配器对相应的白盒测试用例数据进行解析，对被测系统源代码进行数据标准规范检查、源码自动分析、代码质量测试等，检查结果自动存储数据库。

b.黑盒测试部分（功能测试、性能测试、接口测试、人机界面测试、边界测试等）

测试工具云化服务模块根据系统业务流程场景，将对应的执行脚本分别部署于对应被测系统终端，按被测系统的功能和流程自动执行功能、性能、接口等测试类型测试。测试执行结果自动存储到数据库。

c.测试数据采集与分析系统自动采集测试结果，结合白盒测试测试结果，对测试结果进行分析后分析结果存入数据库，并由测试过程管理系统生成对应的测试记录和测试问题报告单。

### 测试回归阶段

测试项目负责人将测试问题报告单与研制方进行确认后，正式提交给研制方，由研制方对测试问题进行修改。

研制方对测试问题修改完成后，测试方对被测系统进行第一轮回归测试，如问题未全部关闭，继续由研发方进行修改，如问题全部关闭，则进入到测试总结阶段。

### 测试总结阶段

研制方测试数据采集与分析系统采集的测试结果，对测试数据进行分析后，结合测试需求分析结果，生成测评报告，给出测评结论。

之后通过网上评审的方式，组织专家对测评报告内容进行评审，给出评审修改意见后，根据修改意见对测评报告进行修改。

## 系统的工作流程

使用测试云平台开展试验和测试工作时，根据对软件测试业务的分析，系统的工作流程见图XX所示。



图4-1 系统工作流程图

1、利用软件测试资产库管理分系统的软件状态控制与管理系统进行被测件接收时的状态检查；

2、利用测试运行支撑分系统的被测对象管理功能，将符合技术状态要求的被测对象入库存储；

3、利用软件测试资产库管理分系统的软件产品研制过程管理中的测试管理功能，对存储的被测件对象进行测试需求分析与测试设计，分析与测试设计时利用测试数据采集与分析分系统开展测试用例设计的辅助设计支持；

4、利用测试运行支撑分系统的测试环境定制功能定制需要的测试环境，为测试用虚拟机、网络环境建立各种配置信息；

5、利用测试运行支撑分系统创建测试虚拟机和网络环境；

6、利用软件测试资产库管理分系统的软件产品研制过程管理中的测试管理功能，进行测试用例的执行过程；

7、测试运行支撑分系统中的测试工具云平台适配模块根据测试用例所需要使用的测试工具，调用软件测试工具分系统中相应的软件测试工具，开展对被测软件的测试工作；

8、测试数据采集与分析分系统在测试过程中，对测试配置数据、测试过程数据、测试结果数据进行测试数据采集；

9、测试执行结束后，利用测试数据采集与分析分系统对测试数据进行综合分析，以可视化的方式显示出分析结果；

10、测试项目结束后，将测试的各类数据资源，包括被测对象、测试需求、测试用例、测试执行结果等，利用软件测试资产库管理分系统将这些资源作为组织资产，当需要时可对形成的组织资产进行统计、查询以及各种分析工作。

## 系统的应用模式

系统主要有两种典型的应用模式，分别为：分布式信息系统软件装备的试验模式和电子信息装备软件测评任务模式。

分布式信息系统软件试验应用模式与软件测评任务不同，其重点是按照软件研制总要求的规定，校验被试软件的功能性能与研制总要求规定的战技指标的复合度，在软件试验中，被试软件运行环境和外部信息交互关系的构建要求更加贴近被试软件的实际运行环境。

软件测评应用模式主要按照《GJB2725A军用软件测评实验室通用要求》的相关要求和软件测评任务书的具体规定，进行以下15个类型的软件测评工作：

* 文档审查
* 静态分析
* 代码审查
* 功能测试
* 性能测试
* 接口测试
* 人机交互界面测试
* 强度测试
* 余量测试
* 边界测试
* 安全性测试
* 可靠性测试
* 恢复性测试
* 安装性测试
* 兼容性测试

### 分布式信息系统软件装备的试验模式

分布式信息系统软件具有节点多、组网复杂，应用模式灵活等特点，软件测试云平台可提供基于需求的精确计算资源、软件可定义的网络拓扑结构模式可重用的软件测试环境。

在分布式信息系统软件装备的试验模式应用中主要的特点有以下几点：

* 根据信息系统软件装备研制总要求和软件席位运行环境，生成测试运行的环境；
* 根据信息系统的组网模式定义出网络拓扑结构，可实现不同网络性能的仿真；
* 实现仿真试验系统的云化部署；
* 无法云化的仿真系统可以通过外部网络接入到系统；
* 实装真实设备可以通过外部网络接入到系统；
* 试验结果数据可进行大数据的分析。

### 电子信息装备软件测评任务应用模式

在电子信息装备软件测评任务应用模式下，软件云测试系统主要实现软件测试工具的云化应用，针对测试工具的可云化程度，主要分为三种应用方式

* 多用户并行使用方式
* 多用户分时使用方式
* 单用户登陆使用方式

在电子信息装备软件测评任务应用模式下，使用软件测试云平台系统主要有以下几个特点：

* 主要针对软件开展配置项和系统级别的测试；
* 测试工具和被测件运行环境主要以虚拟测试用机为主要支撑；
* 测试过程主要按照GJB2725A附加技术能力要求的几个测试阶段进行；
* 云化的测试工具调度需要考虑多用户并发及多用户使用的情况；
* 多个软件同步开展测试时，可以构建出多套测试环境，但需要考虑License的资源限制时的分时调度；
* 测试时要按照完整的测试文档模板要求实现测试过程文档的生成。

# 关键技术解决与验证

在软件测试工具研制项目中，涉及到的主要内容为测试云支撑分系统的开发、测试数据采集与分析系统的开发、测试大数据的积累与组织资产利用等相对较先进的技术，在这些先进的技术开发中，需要解决的关键技术问题主要有5个方面，分别是：

* 用于测试过程管理的工作流技术
* 用于测试大数据积累与组织资产利用的资产检索技术
* 用于测试支撑分系统的并行任务生成技术
* 用于测试支撑分系统的测试任务智能调度技术
* 用于测试数据采集与分析分系统的测试用例智能推荐技术

上述5个方面的关键技术，我司均进行了相应的攻关与开发工作。

## 工作流技术

在本项目中，大量的地方需要用到工作流技术，如软件研制管理过程、软件测试过程、软件测试环境的开设过程、软件资产管理的审批过程导尿管。工作流是一系列相互衔接、自动进行的业务活动或任务。

一个工作流包括一组任务（或活动）及它们的相互顺序关系，还包括流程及任务（或活动）的启动和终止条件，以及对每个任务（或活动）的描述。

工作流包含两种类型的工作流：顺序工作流和状态机工作流。

顺序工作流提供了一系列有组织的步骤，一般情况下，步骤是逐一执行的。可能有的步骤需要等待某些事件的发生才可以继续执行，但通常情况下顺序工作流一般用于无需人工干预的操作。

状态机工作流提供了一系列的状态。工作流从初始状态开始，到终止状态结束。两个状态之间定义行为进行过渡。通常情况下，状态机工作流对事件作出反应，事件的发生将会使状态发生改变。

### 工作流的基本原理

在工作流引擎中，首先需要对业务流程、过程定义、过程、活动、工作流管理系统、过程活动实例几个基本概念进行定义。

下图6-1-1是这几个概念的关系图。



图 6-1-1 概念关系图

（1）业务流程(Business Process)

是在功能确定的组织结构中，能够实现业务目标和策略的相互连接的过程和活动集。例如：被测件处理过程、项目开发过程等。

（2）过程(Process)

是业务流程的规范视图，由一系列为共同目标连接在一起的协同的过程活动组成。

（3）活动(Activity)

指的是工作流中的一个逻辑步骤或环节，是过程执行中可被工作机调度的最小工作单元。它既可以是计算机自动执行，也可以是人来完成。

（4）工作流管理系统(Workflow Management System)

是一种能定义、创建和管理工作流执行的系统。它可通过单个或多个工作机运行，并能存储和解释过程定义。它包含的信息有：开始和结束条件，可参与到此环节中的拥护，完成此活动所需的应用程序或数据，以及关于此活动应如何完成的一些限制条件。

（5）过程活动实例(Process Activity Instances)

指的是实际运行中的一个过程活动。每个实例代表一个能独立控制执行、具有内部状态的线程，可被外界通过标识进行存取。过程实例有工作流管理系统创建、管理、终止、它与过程定义相对应。每个活动实例代表一个活动的一次调用，只和一个过程实例相关，并使用过程实例的数据。一个过程实例可能有若干个活动实例组成，但一个活动实例不能同时属于几个过程实例。

工作流在大多数的实际应用中的情况可以这样来简单地描述：在网络、服务器和多台计算机客户端的硬件平台上，业务过程按照预先设定的规则并借助应用程序和人对相关数据的处理而完成。例如，在日常办公中，当撰写好某份报告之后，可能需要将其提交给领导进行审阅或批示；审批意见可能需要汇集并提交给另外一个人，以便对报告进行进一步的修改。这样，可能会形成同一篇文档在多个人之间的顺序或同时传递。对于这样的情况，我们可以使用工作流技术来控制和管理文档在各个计算机之间自动传递，而非手工传递。

类似的关于文档的自动化处理只是工作流技术的一种简单应用。事实上，工作流技术在现实生活中能够完成更多更复杂的任务。如企业(或机构)内部的各种数据或信息的自动处理，多种业务流程的整合，企业(或机构)之间的数据交换，借助Internet技术实现跨地域的数据传输和处理等等。

### 自定义工作流

工作流可以分为两类：

一类是固定工作流：

流程的基本设定和用途己经确定，用户可自定义的内容很少，基本上是直接应用。各种企业信息系统内集成的处理常见工作事务的模块都可以算做固定工作流。

另一类是自定义工作流：

系统只提供工作流设计和运行的环境以及基本元素，将流程定义独立出来，不和具体业务绑定，由系统管理员根据企业自身需求，用这些元素组合出各种各样的流程。

自定义工作流户定义与任务执行完全分离。整个工作流的表单和流程步骤都是由系统管理员预先设计好的，办理时每个步骤的操作人员在限定的区域操作，完毕后选择下一分支与经办人进行转交，如此一步步流转直至结束流程。用户可定义部分包括如下部分：

* 表单设计

作为工作流数据的载体，表单力求易编辑、易操作和美观。管理者可以根据需要来设计满足需要的表单。

* 控件使用

用户可操作的就是表单上的控件，因此表单上可以使用的控件类型直接决定了工作流中可以存放哪些形式的数据，基本空间类型包括标签、文本输入、下拉菜单、单选列表，复选列表，日期等。系统中的工作流还提供诸如公式计算，SQL查询语句之类的高级控件，能够设计出功能更强大的表单。在我们的工作流中还包括如自动统计，手写审批等高级控件。

* 图形化流程设计

我们图形化的流程表述取代了传统的列表方式，这给设计者和使用者提供了很大方便。并能提供更多直接显示的内容，如转向条件等。

* 流程转向控制

工作流转交时，往往同时有多个方向可以选择，流程转向控制即是根据已有条件自动判断出该往那个方向进行，或者允许往哪个方向进行。这一控制减少了使用者误操作的概率，提高了工作流的自动化程度。使用者可以直接设定转向条件，根据条件，流程会实现自动的转向。

* 统一步骤多人办理

转交时设定一个主办人，其他人只能在限定的区域书写自己的看法。这就是确定主办权的模式。

同时交多人，谁先接谁办理或谁先接谁主办。用于有多人处理相同的事物，不必指定由谁办理的场合。这其实是自动分配主权的模式。

多人会签，所有人都办理完或已开始办理才能结束本步。此功能用于一件事物需要多人审批的场合，避免了逐一传递工作的繁琐步骤。这可以算是没有主办权的模式。我们的工作流应该同时支持上面三种模式，以适应现实工作中的不同需要。

### 工作流引擎的对外接口

工作流的参考功能结构如下图6-1-2所示。



图 6-1-2 工作流功能结构图

（1）工作流服务与工作流建模工具之间的接口

工作流服务与工作流建模工具之间的接口，包括工作流模型的解释和读写操作。此接口实现了建模环境和运行环境的分离，使用某种建模工具创建的模型可以运行在不同的工作流产品上。此外，它还是得多种工作流产品可以协同工作，构成一个统一的工作流执行服务。

（2）工作流执行服务与客户应用之间的接口

工作流执行服务与客户应用之间的接口，用来约定所有工作流客户端应用与工作流执行服务之间的操作方式。

这些操作包括参与运行的各系统间的通信管理、工作流实例的创建与激起或挂起、过程状态的管理、用户任务列表项的处理、工作流实例的状态监控与信息查询等等。

为了完成以上诸操作，我们提供一套标准的API集合用来使用户在不考虑实际工作流产品实现平台的情况下以一种一致的方式通过工作流应用客户端访问工作流引擎和任务列表。

任务列表工作流执行期间必须与用户交互的地方，工作流引擎在其中放置项目以使工作流参与者与工作流实现交互。

（3）工作流相关数据设计接口

工作流引擎直接调用的应用程序接口。在异构产品环境中，为了实现不同的工作流产品间相互调用，系统定义了工作流引擎直接调用的应用程序接口。此类接口主要包括以下几个方面的功能：

* 通信建立

建立和断开与应用程序的通信

* 活动管理

启动、挂起、重新激活、结束活动，查询活动状态以及产生及响应各种事件;

* 数据处理

提供工作流相关数据或数据访问地址。在较复杂的情况（如异构环境下的工作流机间的协作）下，该接口可能还需要完成在不同工作楼引擎之间传递激活应用程序所需的数据、传递运行或过程建模完成时的工作流模型数据等其他功能。

（4）工作流执行服务间的互操作接口

不同的工作流引擎之间进行协作的接口规范。

（5）工作流管理与监控设计

为了多种不同的管理应用于工作流引擎相互交互，需要在工作流引擎中设定一个公共的接口完成其与不同应用间的逻辑转换。这种逻辑转换最终也将通过一系列的API来实现。主要包括以下几个方面的内容：

* 用户管理操作

创建、删除、暂停、修改用户或用户组的权限；

* 角色管理操作

定义、删除、修改角色以及设计定角色属性；

* 审核管理操作

查询、打印、删除、开始新的审核跟踪日志；

* 资源控制操作

设置、取消、修改过程或活动实例的并发层次；

* 过程监控管理

初始化过程实例、改变过程实例或活动状态、终止过程实例；

* 过程状态管理

查询过程或活动实例的运行状态。

### 工作流系统的框架设计

系统的工作流开发平台的主要部件分为可视化建模工具、工作流引擎系统、客户端、监控与管理工具、资源管理系统（后台数据库服务器）。

平台通过建模工具，依托工作流引擎、后台数据库服务器实现流程流转，采用基于Web的客户端和监控工具完成对流程的监控。如下图6-1-3所示：



图 6-1-3 工作流系统总体框架

系统的工作流程大致如下：设计者在了解业务流程要求后，使用系统设计流程设计器把业务流程转换成一个有向流程图，这是自定义工作流设计器使用的重要一步。有向流程图各节点及其关系通过XML的方式存储于数据库中，由定义态引擎进行解析。用户填写相应的表单，然后申请，生成过程实例。过程实例由运行态引擎导航，逐步实现活动的实例化，其中的审批意见均在流程设计器中定义，比如：审批通过、不通过、返回上一步、返回发起人等。

管理者应用监控和管理工具可以实时的控制业务流程的进展，挂起，返回，终止等，除此之外还可以控制设计者定义的流程是否可用。

### 工作流引擎的主要组件

本系统中的工作流组件主要包括：流程解析器、 流程管理器、 任务分配器、 任务执行器、流程动态调整器、 任务预警器和调度管理器等。

如下图6-1-4所示：



图 6-1-4 工作流组件

（1）流程解析器

流程解析器的主要作用是将用户提交的形式化的流程定义解析到数据库中， 包括流程定义文件中的库所、 变迁、 有向弧、 任务、 条件和事件等元素。

（2）流程管理器

当工作流引擎完成对流程定义文件的解释与持久化工作后，用户即可根据流程定义来创建流程实例，从而启动对现实业务的一次执行。在工作流引擎中，流程管理器主要是对工作流管理系统中的流程实例进行管理，包括创建、终止、挂起相关的流程实例等。

当流程管理器根据调度管理器的指令创建并启动了一个流程实例后，该实例就会根据数据库中的流程定义数据开始流转。

当流程管理器得到了调度管理器的暂停指令时，将会根据此时目标流程实例中令牌所在的库所位置来确定目前正在执行的任务，并暂停其执行，这样令牌就会停留在当前库所中而不再继续向下流转， 从而实现对流程实例的控制。

（3）任务执行器

任务执行器是工作流引擎的核心组成部分，主要负责管理流程实例中相关任务的执行与管理，并为任务的执行调度外部资源，而变迁节点中任务的执行也推动了工作流实例的不断流转。

在工作流引擎中，变迁是改变流程实例执行状态的驱动元素，任务的执行也是发生在变迁节点中，但是任务、工作项、事件和活动分别具有不同的含义。

任务是泛指一个工作的逻辑单元， 并不是某个流程实例的一次具体执行，可能在多个流程实例中都需要执行某个相同的任务。

在每个任务中可以包含多个事件，每个事件都是原子性的，即事件不可分割且必须被完整的执行， 如果在执行过程中出现了异常， 则必须进行“回滚” 操作， 以返回到该事件执行前的状态。

工作项是任务中的事件与流程实例的结合体，只要工作流实例中包含某个任务的变迁已处于就绪状态，那么工作流引擎就可以根据该任务中的事件来创建工作项，然后任务分配器将会动态地为该工作项分配外部资源。

另外，事件可以分为两种类型， 一种是计算机可以自动执行的， 例如自动发送邮件、统计在线人数；另一种是需要人员进行交互的， 例如在线填写审批信息等。

在流程实例的运行过程中，任务管理器还需要为任务中的事件创建工作项和活动信息。其中，任务、事件、工作项、活动之间的对应关系如下图6-1-5所示。



图 6-1-5 任务事件工作项对应关系

（4）任务分配器

任务分配器的主要作用是对流程实例中需要人工执行的事件合理地分配给具有相关处理权限的人员， 并实现不同人员之间的负载均衡，这就需要任务分配器依据一定的任务分配策略来对流程实例执行过程中的任务进行动态的分配。而且，任务分配的合理性会直接影响到企业的业务处理效率，以及相关数据的安全性。

（5）流程调整器

流程调整器的主要作用是在工作流模型被修改的情况下，保证受到修改影响的工作流实例能够继续正常执行。一般情况下，在修改流程定义后创建的工作流实例，将会按照新的工作流模型进行执行；处于执行过程中的工作流实例有三种调整策略：

* 全部按照新模型重新开始执行
* 按照旧模型继续执行直到结束
* 比较新工作流模型和旧模型之间的区别，找出流程定义中发生变更区域的起始节点，如果工作流实例当前的执行节点在起始节点的前面，那么流程实例可以直接迁移到新的模型下开始执行；如果工作流实例已经执行到发生调整区域起始节点的后面，那么工作流实例需要回退到起始节点，并迁移到新的工作模型中开始重新向后流转与执行。

迁移策略的选择与工作流实例的具体迁移操作都是流程动态调整器的主要任务。

（6）任务预警器

任务预警器可以根据任务处理的历史数据来预测未来任务数量的变化趋势，从而实现对人力资源进行动态的配置，达到提高人力资源利用率、 节约办公和生产成本的目标。

### 基于BPMN的业务流程建模

BPMN是由BPMI（The Business Process Management Initiative）开发了一套标准叫业务流程建模符号（BPMN - Business Process Modeling Notation）。BPMN定义了[业务流程图](https://baike.so.com/doc/5399955-5637517.html)，其基于流程图技术，同时为创建业务流程操作的图形化模型进行了裁减。业务流程的模型就是图形化对象的网图，包括活动（也可以说工作）和定义操作顺序的流控制。BPMN2核心分层结构如下图6-1-6所示。



图 6-1-6 BPMN2核心分层结构

业务流程图由一系列的图形化元素组成。这些元素简化了模型的开发，且业务分析者看上去非常熟悉。这些元素每个都有各自的特性，且与大多数的建模器类似。比如，活动是矩形，条件是菱形。应该强调的是：开发BPMN的动力就是为了在创建业务流程模型时提供一个简单的机制，同时又能够处理来自业务流程的复杂性。要处理这两个矛盾的需求的方法就是将标记的图形化方面组织分类为特定的类别。

在BPMN中，具有以下几种基本的类型：

* 流对象(Flow)

流对象包括事件、活动、条件三个核心元素，一个事件用圆圈来描述，表示一个业务流程期间发生的东西。

事件影响流程的流动，一般有一个原因（触发器）或一个影响（结果）。基于它们对流程的影响，有三种事件：开始，中间以及终止事件。

* 连接对象(Connection)

连接对象将流对象连接起来形成一个业务流程的基本结构。提供此功能的三个连接对象是：

* 顺序流

顺序流用一个带实心箭头的实心线表示，用于指定活动执行的顺序。

* 消息流

消息流用一条带有开箭头的虚线表示，用于描述两个独立的业务参与者（[业务实体](https://baike.so.com/doc/7005452-7228334.html)或业务角色）之间发送和接受的消息流动。在BPMN中，用两个独立的池代表两个参与者。

* 关联

用一根带有线箭头的点线表示关联，用于将相关的数据、文本和其他人工信息与流对象联系起来。关联用于展示活动的输入和输出。

* 泳道(Swimlane)

BPMN支持2种主要的泳道构件。

* 池

池描述流程中的一个参与者。可以看做是将一系列活动区别于其他池的一个图形容器，一般用于B2B的上下文中。

* 道

道就是在池里面再细分，可以是垂直的也可以是水平的。道也是用于组织和分类活动。

* 人工信息(Artifact)

人工信息添加到建模的业务流程上下文中作为信息备注，便于人员理解，当前BPMN规范的版本预定义了3种人工信息。

* [数据对象](https://baike.so.com/doc/9696829-10043128.html)

数据对象是一个显示活动是如何需要或产生数据的。它们通过关联与[活动连接](https://baike.so.com/doc/6035696-6248701.html)起来。

* 组

组用一个虚线的圆角矩形表示，用于记录或分析的目的，但不影响顺序流。

* 注释

注释是建模者为BPMN图的读者提供附加文本信息的一个机制。

下图6-1-7为BPMN图元的集合。



图 6-1-7 BPMN图元集合

下图6-18示例为通过BPMN所建立的人力资源与商务部门之间交互的工作流模型。



图 6-1-8 工作流模型示例

BPMN 是一种图形化的抽象模型表示图元集，是一个开放的业务流程标准，通过这个标准，将可以实现在各种不同平台环境中的兼容。

使用 BPMN 标准的流程建模工具将使最终用户受益颇丰，因为使用 BPMN 建模工具将无需考虑建模工具间的兼容性问题，这样极大地方便了数据通讯、协作和标准化。 目前已经有许多支持 BPMN 建模工具，但是还没有能够直接执行 BPMN 的引擎系统， 这是由于以下原因：

1） OMG组织在制订和发展 BPMN 规范的时候根本就没有考虑执行 BPMN 这样的需求， 所以BPMN规范没有能够制订明确的执行语义；

2） BPMN 规范本身的目的是为了提供对业务分析人员进行大粒度地业务流程分析的支持，而直接对BPMN 的解析执行将违背BPMN的工作流建模系统设计规范产生的初衷和宗旨。

为了能使BPMN模型具有可执行的能力，需要将BPMN模型转换为业务流程执行语言BPEL（Business Process Execution Language for WebServices），BPEL基于XML和Web服务技术， 它融合了早期的IBM的Web Services Flow Language（WSFL） 及微软的XLANG规范的很多特点。

随后许多主要供货商如SAP等公司陆续加入规范的制定， 并催生了多项修改和改进，并于2003年3月发布了1.1版。2003年4月，BPEL被提交结构化信息标准促进组织（OASIS）以实现标准化，并组建了 Web服务业务流程执行语言技术委员会（WSBPEL TC）， 该努力使BPEL在业界获得更为广泛的认可。

BPEL的模型如下图6-1-9所示，流程（process） 由一系列活动（Activity） 组成；流程通过伙伴链接（Partner Link）来定义与流程交互的其他服务；服务中可以定义一些变量（variable）；流程可以是有状态的长时间运行过程，BPEL引擎可以通过关联集合（Correlation Set）将一条消息关联到特定的流程实例。



图 6-1-9 BPEL模型

由BPML到BPEL映射的过程如下图6-1-10所示。其基本的思路是定义一个适合用于转换的具有条件限制的BPMN 的子集。对BPMN进行仔细的研究，提取其核心元素。这些核心元素将几乎可以描述绝大部分的BPMN图元能够描述的抽象流程。然后将定义好的子集进行进一步分析，分解成为只有一个入口和出口的适合于映射的块结构组件，对这些块结构组件进行归纳总结出若干个基本的模板映射结构，最后在此基础上，使用URIM（Unit Replacement Iteration Mapping，单元替换迭代映射）映射算法实现到从BPMN 抽象模型到 BPEL 具体模型的映射。



图 6-1-10 BPML到BPEL映射过程

URIM单元替换迭代映射算法模型如下：

URIM（单元替换迭代映射： Unit Replacement Iteration Mapping） 算法

算法说明： 根据输入的 BPMN 模型获取对应的 BPEL 模型

输入参数： input.bpmn 文件

输出参数： output.bpel 文件

01 **Begin**

02 Var E←input.bpmn

03 //将输入的 bpmn 模型文件赋值给迭代执行映射的bpmn模型文件E

04 Var output.bpel //新建输出文件 output.bpel

05 Var num←0 //临时 bpel 文件个数标号

06 Var T[0..MAXSIZE] //临时 bpel 文件数组

07 if (isFinalStruct(E)) //如果 E 是“最终结构”

08 doFinalStructMapping(E, & output.bpel) //执行“最终结构” 的映射

09 return output.bpel

13 while（isNotFinalStruct(E)）

14 if（haveRepeatWhileStruct(E)） //如果 E 中有 repeat+while 模板映射结构

15 doRepeatWhileMapping(E, &T[num++])

16 if（haveSequnceStruct(E)） //如果 E 中有 sequnce 模板映射结构

17 doSequnceMapping(E, &T[num++])

18 if（ haveFlowStruct(E) ） //如果 E 中有 flow 模板映射结构

19 doFlowMapping(E, &T[num++])

20 if（ haveWhileStruct(E) ） //如果 E 中有 while 模板映射结构

22 doWhileMapping(E, &T[num++])

23 if（haveSwitchStruct(E)） //如果 E 中有 switch 模板映射结构

24 doSwitchMapping(E, &T[num++])

25 if （haveRepeatStruct(E)） //如果 E 中有 repeat 模板映射结构

26 doRepeatMapping(E, &T[num++])

27 if（havePickStruct(E) ） //如果 E 中有 pick 模板映射结构

28 doPickMapping(E, &T[num++])

29 endwhile

30 puoduceOutputBPEL(T, output.bpel)

31 //根据每一次迭代产生的临时 bpel 模型文件， 产生最终的 bpel 模型文件

32 **return output.bpmn**

33 **End**

## 资产发现与检索技术

平台中的软件测试工具、仿真系统、被测对象、测试设计都会作为知识资产或软件资产存在，要完成一个测试对象的测试时，如何在云平台中自动发现和检索所需要的知识资产和软件资产就显得非常重要，平台必须提供有效、方便的资产检索机制。

信息检索诞生于本世纪50年代，主要运用于图书馆等部门用来存储和管理文档。到了年代，信息检索领域已经在文档、索引模型、匹配策略等方面得到很好的发展，有关键词检索、布尔逻辑检索、限制检索、嵌套检索和邻接检索等。随着数据规模的扩大以及资产的复杂性增加，目前的检索算法已经无法来胜任整个资产库的检索任务。

### 传统信息检索的局限性

传统信息检索方法都具有较成熟的理论基础，但这些都借助于目录、索引和关键词等方法来实现，在大多数实际应用的情况下，上述方法仅考虑检索条件在计算机中的编码表示的匹配，却没有考虑到用户的意图或相关的领域信息，具有普遍的局限性：

（1）用户的需求难以表达准确

在大多数情况下用户很难通过简单的几个关键词来忠实的表达其检索目的，由于表达困难也就导致检索质量难近人意。而且用户通常不可能知道可能存在的资产的确切描述,加上用户描述的不规范，用户往往无法确切表示自己的需求许多和需求相关但表达不一样的概念无法被检索到，造成了在资产检索时查全率的降低。

（2）缺乏语义理解，很难进行知识检索

检索过程中仅仅借助于目录、索引和关键词等方法来实现基于内容的匹配而没有考虑语义，往往只有查询词出现在文档中才可能检索到，而且计算机无法通过一两个关键词来理解用户的检索需求，而且会导致查询结果中返回过多的无关信息,即返回许多不符合用户需求的资产，造成了在资产检索时查准率的降低。

（3）颏面分类表达能力有限，很难进行知识检索

传统的资产检索方法采用刻面分类检索机制，使用检索对象的描述树与资产库中资产的刻面表示树进行匹配，这对于分类非常明确的资产库是非常有效的，但是对于分类比较模糊的资产，将无法满足检索需求。且刻面树和术语空间的表达能力有限，刻面及术语方案往往只能满足软件高层描述的需要，无法体现特定领域的描述内容。

资产检索工作是使用资产的开始，它将直接影响着后续的资产复用、集成及组装等步骤。资产检索的核心问题就是如何通过分析用户查询资产的信息，来查询资产库中的资产描述信息，找到与其相匹配的资产或资产集合。资产的检索方法依赖于资产的标识和分类，用户借助资产库管理系统检索所需资产时，必须依照提出的一组术语来组成被查找资产的形式化描述，通过这些术语来查找所需的或与所需功能相近的资产。

### 资产的描述规约

对于资产的描述规约，需要更形式化的表示以支持工具化处理，为此我们采用剖面描述文件来规约资产。剖面描述文件用于验证资产实体描述文件的正确性，描述了特定类型资产描述文件的语义约束。剖面描述文件是一种基于XML Schema的文档（即.xsd文件）。

XML Schema定义了文件的所有标记以及标记之间的逻辑关系，它可以帮助分析程序校验文件标一记的合法性，涉及同一领域的文档可以使用统一的Schema。项目使用XML Schema伴随着资产实体描述，作为资产实体描述的类型定义文件，定义了资产的组织结构、内容、限制、可共享的词汇表以及使用这些词汇表的文档结构以及提供它们之间的联系手段。

设计一个完备的、可扩展的资产描述体系，是构建整个资产库管理系统的关键。一个良好的资产描述方法应具有一定的形式化描述的逻辑基础，它不仅要有丰富的语义，而且应该具备语法简单、结构清晰易读、语义的扩展能力强等主要的特点。

XML语言本身的树结构特征,良好的规范性，以及独有的扩展性、结构性和有效性提供了我们开发这样一种描述方法的可能，使得XML语言成为资产描述的上佳选择。资产实体描述文件是描述可复用资产的文档，它是对某特定剖面验证有效的文档,一般命名为或。

资产实体描述文件对于资产而言就相当于身份证，标示了资产的剖面类型、包含的制品、使用规则、各种分类信息和相关资产信息等。而每个实体描述文档可通过引用与剖面相关的文档来对其进行验证。下图6-2-1为资产描述的组成。



图6-2-1 资产的描述组成

### 资产的本体建模

本体最早是一个哲学的范畴，是对世界任何领域内的真实存在所做出的客观描述。20世纪90年代以来，人们将本体的概念引入计算机领域，本体被赋予了新的定义。关于本体，目前最完善且被公认的定义是：“共享概念模型的明确的形式化规范说明”，本体模型包含以下几个方面的含义：

（1）概念模型

本体是客观世界中一些对象的特性或本质揭示出来而得到的抽象模型。

（2）明确

所使用的概念，概念与概念间的关系，以及使用这些概念的约束都有明确的定义。

（3）形式化

对本体的定义具有精确的数学描述，使计算机可理解。

（4）共享

本体中表达的知识是其使用者共同认可的，是某个特定领域的公共知识，反映的是相关领域公认的概念集。

构造本体模型的过程如下：

（1）确定本体表示的对象的目的及范围，定义领域内的概念、属性及其定义域；

（2）分析各个领域本体之间、本体属性、本体类之间相互关系，定义概念之间的层次关系；

（3）添加属性的实例；

（4）使用本体建模语言表示本体；

（5）对构造的本体的检测和评价。

下图6-2-2为构造本体的流程图。



图 6-2-2 构造本体流程图

本体建设的方式一般有两种类型一种是在领域专家和知识工程师的参与下，从抽象功能概念入手构建领域本体框架；另外一种是利用己经存在的自由文本、半结构化数据、数据库等网络资源，从这些资源中抽取语义信息来构建本体。

构造资产本体的目的是为了实现资产的语义检索。资产本体的作用在于给出了某个领域内的特定词汇表、概念以及概念之间的关系，从而清晰地表示了该领域资产的层次结构与语义关联关系。

将软件资产的描述模型描述为一个六元组 A=<AID、AA、AT、AD、AO>。

A：表示资产本体

AID：表示资产的标识

AA：表示资产的属性集合

AT：表示资产的子类描述树

AD：表示资产文件

AO：表示资产本体树

资产的属性集合AA是描述资产的基本信息的集合。

AA={Attribute| Attribute=<name，version，author，date，short\_description>}

分别为资产名称、资产版本、资产作者、时间、简要描述。

资产子类描述树AT由四个主要部分组成：分类，解决方案，用法以及相关资产。

AT={SubTree| SubTree =<Classfication，Solution，Usage，Profile，Ralated-Assets>}

Classfication={ Classfication| Classfication=<context，descriptor-group，descriptor>}

Solution={ Solution| Solution=<requirement，design，test，implementation>}

Usage={usasge| usasge=<areifact-activities>}

Profile={profile| profile =<name，id\_history>}

Related-Assets={ related-Assets| Related-Assets=<name，type>}

本体构建的关键在于定义各种类、类之间关系、属性、属性间关系、属性约束等。类可以进行层次划分,即一个类可以有子类,子类则表达了更为明确、详细的概念。

资产的本体描述采用OWL语言进行，本体包含类、属性、个体及本体元数据等信息,各种构造子和公理用于描述类、属性和个体,以及它们之间的逻辑关系。主要描述了类层次关系、传递属性以及蕴含规则三方面的内容。

在本体建设过程中存在很多大量重复性的工作，因此从零开始建立本体并不是最适宜的方法。最切实际、经济的方法是利用己经存在的自由文本、半结构化数据、数据库等网络资源,从现有信息资源中抽取语义信息来构建本体。

### XML中提取本体模型

在资产的描述中，通常是基于XML文档格式的，其中的标签选取体现了对可复用资产的描述。然而，XML本身包含了语义信息却无法表达出来，使用本体可以显示描述形式化的语义信息。

* **定义1：Simpletype元素**

在树模型中,如果一个元素在树结构模型中是一个叶子节点，即不包含子元素也不包含属性，称这个元素为Simpletype元素。

* **定义2：Complextype元素**

在树模型中，含有子元素或属性的元素称之为Complextype元素。

项目采用XMO算法上述由资产实体描述文件到资产本体的转换问题，实现半结构化数据的语义自动抽取，完成资产本体构建。

算法的基本思想是：借助每个文档中所包含的元数据和数据实例，元数据模型包含的语义信息用来转换成本体，而数据实例信息用来产生一个实例。算法首先抽取树结构模型中同一层次的不同元数据来映射成一个本体，然后根据产生的本体以及数据实例部分相应产生实例。

**算法：XMO**

输入：资产描述文件（.xml）

输出：基于OWL的资产本体实例（.rdf）

步骤：

1. 生成的OWL本体的根元素是一个rdf：RDF元素，它指定说明了相应的一系列命名空间；
2. 从根元素开始，深度遍历XML文档树的所有元素；
3. 如果遍历到的元素是Complextype型元素，使用该元素的标签名作为类名产生一个owl:class定义的类，转（4）、（5）；否则转（6）；
4. 如果这个元素不是根元素，则产生一个使用owl:Object型的对象属性，该属性使用“has\_”前缀加上元素标签名作为属性名，表示该元素与父元素之间存在着子类继承关系；
5. 产生一个owl:DatatypeProperty型的数据类型属性，来限制这个元素的取值类型。这个属性使用元素标签名加上“\_Value”后缀作为该属性名；
6. 如果遍历到的元素是Simpletype型元素，则使用其产生一个owl:DatatypeProperty属性，该属性名使用与元素标签名相同的名字；
7. 继续深度遍历该树模型的所有元素，直到元素为空，遍历的元素如果是相应本体OWL中的一个类，则声明一个新的资源，这个资源由该元素相应的子元素和属性通过引用本体中的术语构成的三元组集；
8. 继续遍历直到遍历所有元素为空，产生相应的基于语法的代码，即资产本体实例文件。

上述算法完成资产实体描述文件到资产本体文件的转换。

### 基于本体的语义检索

资产检索工作是使用资产的开始，它将直接影响着后续的资产复用、集成及组装等步骤。资产检索的核心问题就是如何通过分析用户查询资产的信息，来查询资产库中的资产描述信息，找到与其相匹配的资产或资产集合。

资产的检索方法依赖于资产的标识和分类，用户借助资产库管理系统检索所需资产时，必须依照提出的一组术语来组成被查找资产的形式化描述，通过这些术语来查找所需的或与所需功能相近的资产。

本体由于具有良好的概念层次结构和对逻辑推理的支持，进一步还可在知识层面上辅助检索，通过主题词典、上下位词典、相关同级词典，形成一个知识体系或概念网络，给用户智能知识提示，最终帮助用户获得最佳的检索效果。

基于本体的语义检索方法，通过充分挖掘数据源内部隐含的语义关系，使检索过程由原来的关键词组匹配转化为内容匹配，比如用户检索“计算机”与“电脑”相关的信息也能检索出来，以帮助用户建立准确高效的查询请求，克服上述仅有表达形式上的关键字匹配检索中出现的查全率、查准率不高的问题，进而更高效地完成软件资产管理中资产的检索过程。

当前随着信息的爆炸式增长，信息查询出现了难以克服的问题信息过载和词不匹配。信息过载影响了信息检索的查准率，而词不匹配影响了查全率。对于关系数据库中的数据查询,由于存在一词多义、多词一义的现象以及内容的上下位关系，大多数用户无法明确表达自己的查询要求，从而导致检索的查全率、查准率大大降低。

为了解决这种语义上的局限，通过以下方法：

1. 首先把查询重写为定义在视图上的查询，通过构建本体模型，使用视图来联系本体和关系数据库中的基本表；
2. 利用全局本体对查询语句中概念间的语义约束进行推理；
3. 集成语义信息对查询进行重写，即在原查询词的基础上加入与用户用词相关的词或词组，组成新的、更准确的查询关键词序列，实现对数据的内容而不是关键字的查询。
4. 新的查询与原查询的查询结果在用户意图上等价，以达到统一数据及领域知识的作用，使用户查询意图更完备。

**定义1：**基于创建视图的语法规则，通过Where子句限制语义约束，结合基本表和本体之间的映射关系进行连接查询，形成的视图称为语义视图。

如下伪SQL语句所示：

CREATE VIEW MyView(Column1，…ColumnN) AS

SELECT head1，…headN

FROM BaseTable AS T，Ontology AS O

Where constructor

AND P1 AND … AND Pk

AND m1 AND … AND mk

其中，约束有三种形式，分别为构造器（constructor）、约束（constraint）、映射（mapping）。

（1）构造器

关系表中的记录实例化本体类型。

（2）约束

本体构造的实例与关系表的约束。

（3）映射

本体属性和关系表属性的映射关系。

**定义2：**R是谓词定义的递归关系的集合，P（G）是本体树中谓词节点，pred（n）是谓词名即谓词关联的对象属性。对于节点n，，则称节点n为递归节点。

**定义3：**由基本表（b），本体（r），视图（v）三类元素组成的三元组称为视图三元组，记为（b，r，v）。基本表与视图的映射元组称为关系型视图元组（relational view triples），视图与本体的映射元组称为虚拟列元组（virtual column tripes）。基本表与本体的映射元组称为本体元组。

**定义4：**对于节点，存在视图元祖（b，r，v），且r=pred(n)，且b非空，则称节点n为基础节点。

对于用户的初始查询建立视图,将己建立好的本体和基本数据表作为查询对象。

通过初始查询构建视图三元组，对视图三元组的每一个原子元组作重写判断。

若是基本表与视图的映射，则用户的查询直接关联到基本表，将基于视图的查询改写成基于基本关系表的查询否则，进入本体库，查询本体树，做树型遍历。直到搜索到基础节点，将本体信息关联到基本表，再进行查询重写，最终返回重写后的查询表达式查询。

上述查询重写算法如下：

**算法1：REWRITE（Q，T，R，V）**

输入：初始查询的原则谓词集合Q、本体树T、递归关系的集合R、视图V

输出：重写后的谓词表达式Q’

步骤：

1. 对于Q={A1，A2，A3，….}（其中A代表初始查询的原子谓词），其中的每一个Ai，均可以表示为Ai=（vcol，op，value）的形式；
2. 调用方法getViewTripe（V，vcol）获取视图三元组（b，r，vcol）；
3. 若r为空，则vcol不是虚拟列，存在基本表和视图间的映射，此时将（vcol，op，value）重写为（b，op，value），赋值给Q’；
4. 否则，vcol是虚拟列，则到本体树中进行搜索，调用findRuleNode（T，R，（r，op，value））寻找节点a；
5. 若找到此节点a，调用扩展算法EXPAND（a，T，R，V），进行重写，转（7）；
6. 否则，不做任何重写，转（7）；
7. 返回重写后的谓词表达式Q’，结束。

算法的流程图如下6-2-3所示：



图6-2-3 查询重写算法流程图

**算法2：EXPAND（a，T，R，V）**

**输入：**本体树中要扩展的谓词节点a，本体树T，递归关系的集合R，视图V

**输出：**扩展后的谓词表达式e

步骤:

1. 若a是基础节点，调用getViewTriple(V, pred(a))获取视图三元组((b, r, vcol);
2. 将(b, op(a), val(a))赋给e，此时若a是递归节点，调用ISSUBSUMED(val(a),pred(a), b)(核对传递属性的子节点是否在基本表中)，若是则赋值给e;
3. 若a是递归节点，则调用subsumedAtoms(R, a)，找到每一个传递属性的传
4. 递树上的节点i，嵌套调用EXPAND(i, G, R, V)，转1);
5. 当a既不是基础节点，又不是递归节点，则搜索节点在本体树中的关联节点i，使用方法dependentExp(a,T)，嵌套调用EXPAND(i, G, R, V)，转1);结束，返回重写后的谓词表达式e.



图6-2-4 查询重写算法扩展流程图

在上述算法中，获取传递属性的子节点、传递属性的传递树上的节点、节点在本体树中的关联节点等，都使用Jena提供的API对本体模型进行操作，本文中在后续4.5.2节中将对Jena提供的API进行详述。利用领域本体的概念网络实现概念扩展，扩展的概念在很大程度上依赖于领域知识的描述，由于特定领域的资产本体是对资产的规范化定义，具备概念化、明确、形式化和可共享的特点，因此，借助其进行的语义重写是合理的，能够更准确反映资产相互关联的关系。

下面给出关于资产重查的算法实例描述。

**算法应用实例**

上述语义查询重写算法形象的实例描述如下。假设初始查询“A=v 1”是针对视图的查询表达式，在基本表中不存在映射，则节点A需要扩展到本体中进行扩展查询。图4-3显示扩展查询的本体树结构，A节点为递归节点(RecursiveNode)，调用REWRTE算法，遍历树中与A节点语义相关的节点，直到找到基础节点(Ground Node) B(可关联到基本表)，同时调用EXPAND算法，扩展得到其他基础节点D，则最后由初始查询“A=v 1”扩展后得到的查询表达式为：" B=v2”和“D==v4"。此时，找到所有的基础节点，查询表达式进行重写，完成整个语义查询重写过程。



图 6-2-5 语义查询本体树结构

**算法优化与改进**

由于对本体树的检索采用遍历的方式，遍历过程的优化，是有待解决和改进的问题。可使用以下方法进行相应优化:

1. 标记本体树中既不是递归节点也不是基础节点的树节点，称之为DeadNode，这样的节点永远不需要遍历。遍历之前可以对本体树进行剪枝，舍弃那些不可能存在映射的分支，扩展算法略过此分支下的节点，减少遍历过程;
2. 使用备忘方法(memorization techniques)，记录节点之间的关系以及树节点的匹配信息，利用这些信息可以简化遍历过程，避免节点重复遍历，同时可以将语义处理部分进行离线完成，因此不会影响服务器的实时响应功能;
3. 在调用REWRITE算法之前，查看系统日志表(system catalog tables预算重写过程是否发生，以降低查询重写的时间复杂度。

通过上述优化方法，可以有效地加速语义扩展的速度，快速地返回扩展结果。

### 语义相似度的计算

概念之间的相似度是指相似的程度,它可以是取值[0，1]之间的一个实数。

如果在一个确定的概念化体系中，两个概念之间蕴涵的语义是相同的，那么，这两个概念就是相似的,相等的概念之间的相似度为1；如果两个概念语义之间完全没有相似的地方，那么这两个概念之间的相似度为0。

语义相似度计算时，我们利用关键词实例之间语义相似度值来度量概念间语义的匹配程度。

设t1、t2是本体中的两个概念。Sim（t1，t2）为表示概念之间的相似程度，则有：

其中，n是术语t1，t2在本体中kind-of关系的层次中所具有的最大深度，是

权重(可简单地取)

取值定义如下:

通过公式返回[[0, 1]间的实数来表示两个概念间的语义相似度，值越大，则相似程度越高。

例如，资产本体中的概念层次片段，如图6-2-6所示。Windows UNIX Solaris都属于Operating System中的一种，而Operating System又属于Infrastructure Enviromnent，根据上述公式，则可以在树结构中计算概念间的相似度，如下：



图 6-2-6 资产本体概念层次示例图

Sim (Run-time host, Infrastructure Environment) =1/3+1/3=2/3

Sim (Windows, Infrastructure Environment) =1/4+1/4=1/2

由Sim (Run-time host, Infrastructure Environment)=2/3> Sim (Windows,Infrastructure Environment) = 1/2，可见，结点“Run-time host”与结点“Infrastructure Envirenment”的语义相似程度相比“Windows”与结点“Infrastructure Envirenment”的语义相似程度更大，从XML文档树结构来看，事实正是如此。

设M={ml, m2 ... Mx)，N=f n1, n2 ...ny}用SimOfset(M，N)表示两个概念集合之间的相似度，同理，两个概念集合之间的语义相似度计算方法如下：

由上述可知，语义相似度越大，表明两个概念间的语义关系越强，反之越弱。只有那些相似度值大于系统设定的闽值的元组才需要被选择出来，返回给用户进行二次查询，从而生成语义元组集。阂值是通过实验确定的，是一个经验值，并且与关系数据库相关，不同的关系数据库数据可能需要设置不同的语义相似度闭值。

通过语义相似度计算返回得到结果集为：{Q' **|**Sim(Q,O) > }。

其中：

Q为查询关键词

O为语义相关集

为设定的阈值

为了更好地获得语义扩展的词汇，相似度计算过程可以进行多次反复。

根据相似度计算公式，每次计算都可以除去相似度值过小的词汇，再将语义相似度符合要求的词汇作为待查询的Q，形成新的检索式，再次进行语义相似度计算，如此反复，得到更优的语义相似概念。

上述方法得到概念间的语义相似度，并据此来筛选扩展的词汇，可统一实现基于语义的细化、泛化和相关联想，为后续基于本体的资产检索任务提供更准确完善的语义检索词汇。

### 资产的语义检索模型

语义检索将检索技术与自然语言处理、人工智能等技术相结合，它从语义理解的角度分析信息对象与检索请求，探索基于语义的方法来对用户查询和检索数据集中的数据进行一定程度的语义分析和处理，使得检索过程和检索结果更关注检索问题的域，是一种基于概念及其相关关系的检索匹配机制。

在基于本体的资产语义检索方法中，用户使用自然语言指定目标资产，系统首先根据用户的需求生成初始的查询请求，在关系数据库中搜索资产元信息，继而根据搜索到的资产存储路径在文件系统中寻找相应资产匹配资产。

如果未找到满足需求的匹配记录，则可以把用户的查询请求映射到领域本体上（关系数据库中的XML列存储的资产本体文件）；

接着，采用Jena2提供的ontology API操作本体，以及支持关系数据库的XQuery，SQL/XML解析本体，进行树型搜索查询(树结构匹配和标签语义匹配)，并利用领域信息对查询请求进行语义扩展，在领域本体中找出最适合的术语描述(语义查询重写)，用户根据返回的术语集进一步明确自己的需求(语义相似度计算)，通过设置闭值将符合要求推理出的结果返回给用户。

最后，根据返回的结果进行二次查询并提交给查询处理模块，查询处理模块搜索关系数据库中相关资产的元信息，来获得资产实体的实际存储位置(关键词匹配)，查找文件系统中的资产实体，以完成检索。

基于本体的资产检索模型主要由两个相对独立的部分组成:语义信息处理部分和查询处理部分。语义信息处理部分主要完成在领域专家的指导下进行领域分析，结合上下位相容性细化、同义词相容性推理及外延概念扩展思想，利用Protege构建资产本体，利用相关算法法进行语义信息自动抽取，利用OWL标准语言进行语义编码;查询处理部分主要完成借助资产本体库，利用Jena进行语义推理，完成由关键词到领域内共享概念的转换，进行该概念与其它概念的相似度分析，将经过处理的查询条件与关系数据库中的资产信息进行匹配。

### 本体查询方法

领域本体使用OWL来描述。项目使用Jena提供的API对本体模型进行操作，本体解析调用API如下表6-2-1所示。

表 6-2-1 本体解析调用的API

|  |  |  |
| --- | --- | --- |
| 类 | 操作 | 描述 |
| OntModel | Read(FiledInputStream,String) | 读入OWL文件到内存 |
| listClasses() | 返回所有Class |
| OntClass | listSubClasses() | 返回Class的所有subClass |
| listDeclaredProperties | 返回Class的所有属性 |
| getURI() | 返回Class的绝对URI |
| OntProperty | isDatatypeProperty() | 判断属性是否为数据属性 |
| isObjectProperty() | 判断属性是否为对象属性 |
| getRange() | 返回与该属性相关的数据类型或概念 |

通过Jena解析本体的基本思想：以owl：Thing类（所有类的超类）作为类树的根节点，通过listSubClasseso方法的调用，获取owl:Thing的所有直接子类并将它们作为根节点的子节点添加到类树中，然后通过递归调用的方法，对owl:Thing的每个子类采用相同的处理过程，构建类树。根据关键词所关联的本体，通过getDomain()和listInstances()方法的调用，确定本体中属性及个体所关联的类，显示类的相关信息。

1. 读取本体文件

用Jena读取OWL本体，首先连接数据库，并为用户本体创建一个可以永久存储的model，从指定文件把模型读到内存里。

使用ModelFactory从数据库中读出指定用户本体的信息并生成OntModel对象。例如:

OntModel m=ModelFactorycreateOntologyModel();

File myFile="D:\Program Files\Protege 3.1\ myAsset.owl";

m.read(new FilelnputStream(myFile),”，，);//读取RDF/XML文件

Model.write(new PrintWriter(System.out)); //标准输出

1. 实现类查询

根据用户选择的类，调用Jena API中的getSubClass()或getSuperClass()方法，得到该类的子类或超类的集合，将结果存入动态数组中;然后处理类的属性查询，根据用户选择的属性，调用getDomain()方法，得到该属性所关联的类，将结果存入另一动态数组中;最后将两个动态数组数据集合并，存入结果集。

1. 实现属性查询

给出一个资源的URI，使用Model.getResource(String uri)方法从图中得到该资源所对应的对象，根据用户输入的属性值域、定义域、属性的类型、全局基约束以及属性的逻辑特征，分别通过

getDomain(), getRange(), isDatatypeProperty(),isObj ectProperty()，isFunctionalProperty()，isInverseFunctionalProperty()isSymmetricProperty(), isTransitiveProperty()

等方法来查询本体中满足条件的属性，将查询结果存入结果集。

1. 个体事实查询

根据用户输入的类、属性以及属性的取值，首先采用isObjectProperty()和isDatatypeProperty()方法判断个体属性的类型，然后根据个体属性的类型执行下一步的查询处理。

如果是对象属性，则根据用户输入的类及属性名，调用listInstances()方法，得到该类的所有个体后再用循环处理的方法，依次判断每个个体对象属性的取值是否包含用户输入的个体，将查询结果存入结果集;如果是数据类型属性，则根据用户输入的类及属性名，调用listInstances()方法，得到该类的所有个体后再用循环处理方法，依次判断每个个体的数据类型属性取值是否与用户输入的数值相等，将满足条件的个体存入结果集。

### 语义匹配策略

检索过程通过关键词匹配在关系数据库中的资产元信息中检索匹配的资产，若此时无法满足需求，则通过树结构匹配、标签语义匹配及Jena相关方法查询资产本体树，再通过语义查询重写并引入相似度的概念优化语义检索过程。

1. 关键词匹配

根据用户选择的检索周境及资产描述的描述子和描述子组，输入关键词，检索资产库中的相应资产描述字段进行模糊匹配。

当检索语境为“context”下，检索关键词匹配的资产，代码片断如下:

NodeList contexts=doc.getElementsByTagName ("context");

for (int i=O;i<contexts.getLengthQ;i++)

{selectType=((Attr) contexts.item(i).getAttributes().item(1)).getValue();

if (searchType.equals(selectType))

check= true;

}

若关键词匹配能检索到用户需要的资产，则完成检索，否则，需要借助资产本体的语义查询完成检索。

1. 树结构匹配

树是一个无环连通图，可以表示为T=（V, E, root(T)）。

其中，V表示树中所有的节点的集合，E表示树中所有的边的集合，root(T)表示树的根节点。

如果(u,v)EE，则表示u为v的父节点，v为u的子节点；

如果((u, v) E E+ ( E+为E的传递闭包)，则u为v的祖先节点。

如果树中的每一个节点对应一个标签，则该树称为标签树。

节点v上的标签记为label(v) 。

设T1=(V1, E1, root(T1)), T2=(V2, E2, root(T2))为两棵无序标签树，如果存在一个V1到V2的映射f，使得节点之间按照某一约束条件满足祖先后代关系，则称为T1与T2在f的作用下是树结构匹配的。

对于一棵元数据标准结构树，要使查询成功，用户的查询输入就要求在结构上必须与元数据标准树的结构按照一定的映射约束条件进行成功地匹配。

不同的结构映射约束条件可形成不同的匹配类型，从而可提供查全率和查准率的调节，满足不同用户的查询需求。

1. 语义匹配

设T1=(V1, E1, root(T1)), T2=(V2, E2, root(T2))为两棵无序标签树，存在一个V1到V2的映射f,使得vEVl, f(v)EV2，若v, f(v)的标签值属于同一个同义词库，则称v与f(v)是标签语义匹配的，简称为语义匹配，记为label(v)U label(f(v))。

在保证树结构匹配的基础上，还必须保证查询中同义词能进行语义匹配，具体表现为查询树与数据树的标签名或标签值近似相同，从而促进查全率的提高。

### 基于本体的语义检索算法

算法思想：当在关系数据库中使用传统的关键词模糊匹配检索，不能快速高效的获取用户所需的检索结果时，就进入高级查询阶段，即本文提出的语义检索功能。

语义检索强调的是基于知识的、语义上的匹配，目的就是使资产检索结果能达到更好的效果。

此时，对于用户的初始查询建立视图，将已建立好的本体和基本数据表作为查询对象，与领域本体中的概念进行树结构匹配或语义匹配，使得语义相关的概念被检索出来，同时通过计算语义相似度，使语义相似度满足一定阈值的概念被作为相关的概念，然后根据相似度进行排序，用户可根据自身的需求筛选出满足一阈值的节点来扩充查询表达式，查询进行重写，然后显示给用户，用来扩充查询。用户则进行二次查询，得到满意的查询结果。

1. 检索算法伪代码

输入：检索关键词集合Q={ql,q2... qn}

输出：匹配资产结果集A={al,a2...am}

算法伪代码:

For all q; E {ql,q2,..q‑)

搜索资产的元信息，执行关键词匹配方法;

Let I={根据q;关键词检索的初始资产检索集}

if(用户对检索集I满意)Then {return I}

Else{

For all qi E {qi,q2...q‑}

{

在本体库的树结构中，搜索到qi的树结构模型;

MOdelRDB类创建一个model实例；

使用Reasoner类创建一个推理机实例并且与提取出的本体模型

进行捆绑；

ModelFactory类的createInflVIodel方法进行推理；

推理结果集保存在InfModel类的实例中；

调用REWRITE算法进行语义查询重写，返回扩展后的查询关键

词集合Q’={qii'，qi2'. .. qin'};

输入值￡;

Let qi'为扩展后选取的查询关键词集;

For all qij' E{qi j’，qj}…qin}

{

计算qi与qi’的语义相关度sim (qi, qij' );

If(Sim(qi，qij’)>“) then

查询关键词扩展为qi’一qi, U qijl

根据查询关键词qij，用RDQL (RDF Query Language)查询

关系数据库，获取匹配的资产值;

Let ai=id;

Return ai; }

1. 检索过程分析

上述检索算法包含以下检索步骤：

步骤1：用户查询接口获取用户提出的查询请求信息并传递给查询构建模块；

步骤2：查询构建模块判断用户是否有更高级的查询需求，否则，直接进行关键词模糊匹配检索；

步骤3：若检索结果未能满足用户需求，查询构建模块将检索关键词交给本体推理引擎，则其根据本体中定义的概念及概念之间的关系给出推理结果；

步骤4：查询处理模块根据推理结果对查询请求进行改写，根据语义匹配方法，得到相关关键词的本体实例集；

步骤5：给定阈值，计算语义相似度，将阈值符合语义相似度范围内的查询关键词返回;

步骤6：返回改写过的查询语句，使用搜索代理，通过资产描述文件及数据库中的相关资产的元信息，在资产库中进行搜索；

步骤7：查询处理模块将搜索结果返回给用户。

通过语义查询重写算法可以检索到语义上与查询条件相符的结果，其中，语义关系包括同义关系、近义关系、可以进一步扩展为上下位关系。

## 基于测试场景的并行任务生成技术

在大型分布式信息系统软件试验中，测试场景设计是非常关键的一环，测试场景对应这想定，通过利用测试场景，建立测试场景流图，自动生成并行的测试路径，将测试路径分为相对独立的测试任务，从而实现并行调度执行。

### 测试场景的定义及其描述

软件产品存在很多不同的场景，为了直观的描述它们之间的关系，这里借助测试场景流图等概念来进行描述。

在交互式程序中，程序中间态及终结态仅由初始状态、执行过程中的输入操作及其发生的时间顺序决定，其中初始状态可能包括环境配置、数据初值和启动参数等。

* 测试场景（Test Scene，TS）

在交互式程序中，可独立运行的、外界能够对其实施输入操作，包括输入数据、发出一条指令等操作的测试活动称为测试场景。

例如指挥控制系统中的一次编成编组操作和文电收发都可以分别定义为一个测试场景。

* 测试场景流图（Test Scene Flow Graph，TSFG）

统一明确的测试环境中，用来描述不同测试场景间执行关系的流图。TSFG 中边是带有方向性的，图中的节点与测试场景一一对应，边代表TS间的执行先后关系。其中添加了两个特殊节点，用来标记执行的起点与终点，代表开始场景，代表结束场景，开始场景及结束场景是不进行真实测试活动的虚场景。

一个测试场景流图如下图6-3-1所示，这个简单的测试场景流图拥有测试开始场景、测试结束场景以及六个具有实际意义的测试场景。

* 可并行测试路径（Test Parallel Test Path，TPTP）

对测试场景流图进行深度优先搜索，探寻到的所有从开始节点到结束节点的测试路径。对于下图中的测试场景流图进行遍历，可以得到以下并行测试路径：

（1）①->②->③->⑤->⑥

（2）①->②->④->⑤->⑥

（3）①->②->⑥



图 6-3-1 并行测试路径

* 等价测试路径（Equivalence Test Path，ETP)

共享同一条测试路径，但具有不同实际意义即执行不同测试用例的测试路径之间互称为ETP。根据实际测试用例的强度ETP可以进行不同强度的测试。

一般来说，测试场景都具有多条测试用例，可以根据不同的测试需求对用例进行分类。测试人员依据实际的需求来选择所用的测试用例，按照测试强度来选择不同强度的测试用例，这里也体现了ETP的高效性。

一条比较完善测试场景用例至少应具有以下九项描述：

（1）Scene ID：用例所属的场景编号；

（2）Casel D：用例编号；

（3）Step：用例的输入步骤描述；

（4）Result：用例的预期结果或所需完成的效果；

（5）Priority：用来表示测试场景用例的优先级，一般场景中的用例可分为不同等级，标志用例的重要程度；

（6）Next Scene ID：下一个被执行的测试场景，能够根据此信息生成可并行测试路径；

（7）Function：该场景用例的执行方法；

（8）Time Out：用例超时时间，以毫秒为单位，超时之后，则代表当前测试场景在这条测试用例上执行失败；

（9）Is Change Env：指明执行该用例是否需要改变测试环境，布尔值，值为true表示改变，值为false表示不改变。

* 测试环境状态(Test Environment State，TES)

用来描述当前测试所需要的系统条件，主要包含测试的原始数据、设备配置、系统状态信息等。表示该测试在何种系统状态下进行，以便于为测试路径调度提供依据以及收集测试结果时为测试报告提供详细的环境信息。

TES是为测试准备的基本环境，用户设置TES要遵循以下原则：

（1）抽取出所有必须的环境状态因素，以键值对的形式表示，如登录测试场景中，用户名字段表示为username，当前测试使用用户名“ZYP”，则可以表示为username：“ZYP”；

（2）环境因素之间用分号隔开，如：username：“ZYP”；password：“123”；

（3）指定测试过程中，Is Change Env字段分别为false和true时所使用的测试环境。

### 测试场景与测试用例结构化文档描述

测试场景与测试用例结构化文档要包含：测试环境以及场景测试用例的语义描述文档。系统中采用XML语言进行描述，测试文档的生成遵循以下五项规则：

（1）对每一个要素采用key-value对的形式描述；

（2）描述测试环境时，每个测试环境状态均有明显分隔；

（3）测试环境文档保存在Test Env.xml中；

（4）描述测试用例时，以测试场景为单位进行描述；

（5）测试用例文档保存在Test Case.xml中。

如下是对登录场景进行测试环境描述的示例，状态描述来源于测试者的输入：

<Test Env>

<!--环境编号-->

<Env id=l>

<!--用户名、密码、是否改变环境-->

<usemame>usemame:"zyp"</usemame>

<password>password:"psw567"</password>

<Is Change Env> Is C hange Env =false</Is C hange Env>

</Env>

</Test Env>

如下是对一个简单的测试用例进行描述的示例：

<Test Case>

<Scene id=s\_id>

<Case id=c\_id>

<!--测试输入、变量及其初始值-->

<Step>Inputs, data and variables</Step>

<!--被测程序运行后的输出-->

<Result>What the program should output</Result>

<!--所关联的下一个测试场景-->

<Next Scene ID>the next scene ID</Next Scene ID>

<!--所执行的方法-->

<Function>Which Function to be run</Function>

<Time O ut>Time O ut=xx</Time O ut><!--超时时间-->

<!--是否改变环境-->

<Is Change Env> Is C hange Env ={false|true}</Is C hange Env>

</Case>

</Scene>

</Test Case>

测试用例文档Test Case.xml自动生成的流程如下：



图 6-3-2 测试用例自动生成流程图

### 测试场景可并行测试路径

按照测试场景描述文档生成测试场景流图以直观描述场景之间的相互依赖关系，从起始节点出发，结束时回到终止节点，根据每个场景中的用例数量生成对应分支。根据测试用例描述文档的Next Scene ID指向下一个场景，若Next Scene ID指向存在，则指向对应的场景；若Next Scene指向不存在，则指向NULL scene节点，NULL scene用来代表没有后续操作的场景。

为了提高测试效率，需要对测试任务进行并行化处理，采用邻接矩阵存储测试场景间的执行先后关系，通过深度遍历该邻接矩阵来构造可并行测试路径。其构造思想如下所示：

（1）根据测试场景的数量n生成一个n×n的邻接矩阵，初始值均为0；

（2）当两个测试场景存在执行的先后关系的时候将对应的邻接矩阵值设为1；

（3）该邻接矩阵已经记录了测试场景间的执行先后关系，对其进行深度优先搜索，就可以得到所有的可并行测试路径。

可并行测试路径生成的伪代码如下：

Begin（算法开始）

初始邻接矩阵 = [([0] \*场景数) for i in range(场景数)]

所有场景用例 = ElementTree.parse(TestCase.xml)

所有场景集合 = 场景用例.findall(‘/Scene’)

For 场景 in 所有场景集合：

For 场景用例 in 场景.getchildren():

if 场景用例.tag == ‘NextScene’:

if case.text !=NULL:

初始邻接矩阵[in(所有测试场景集合.get(‘id’))]

[int(场景用例.text)]=1

场景栈pathstack.append([0])

While len(pathstack)!=0:

Path =pathstack.pop();

For i in range(len(path))

Iter(request,场景数,path)

Return allpath

End(算法结束)

通过可并行路径的计算，生成了多条测试路径，每条可并行的测试路径形成测试任务。

## 测试任务智能调度技术

云测试是一种新的云服务，其数据、服务以及基础架构都是部署在云中的，对其资源、任务的管理调度要采用符合其特性的策略。为测试任务调度虚拟机时，必须考虑云测试的特性，如果按照普通云服务的方式来进行调度，不能充分发挥云测试的作用。云测试中，虚拟机调度策略的优劣将直接决定其测试效率，也决定了其测试任务的执行质量以及虚拟机的负载均衡，因此云测试资源调度算法对云测试来说是至关重要的。

云测试作为云服务的一种，在资源调度过程也具有云服务的诸多特点：

（1）为使用者提供服务，使用者可以在云测试平台申请各种测试资源或直接使用其中的测试服务；

（2）提供个性定制服务，使用者可以选择由自己申请测试资源并通过虚拟技术自主测试环境，以满足其实际需求；也可以在云测试平台中选择满足自己要求的测试服务，由平台自动完成资源的分配以及环境部署等工作，从而可以大大降低使用者的接入成本。

（3）由使用者配置初始环境，例如建立数据库、录入原始数据、配置注册表等；

（4）测试任务智能调度，使用者选择或定制了某项测试服务并开始执行后，由云测试平台来调度虚拟机资源，使测试任务能够高效执行。一般看来，为使用者的测试任务分配固定虚拟机的策略无法满足云测试的效率要求，并且可能导致平台中的虚拟机负载严重不均衡。

因此，云测试平台必须提供满足云测试特性的虚拟机调度策略，以使虚拟机负载均衡，达到测试效率高且使用成本低的目的。

测试资源智能调度是云测试服务的核心机制，下图6-4-1为测试资源智能调度的机制。



图 6-4-1 测试资源智能调度机

### 测试任务的定义

测试任务是用户用来管理测试的基本单位，包括要执行的相关测试用例、测试脚本以及测试任务要满足的约束条件等。测试用例可以对应一个功能模块，也可以对应多个功能模块。在白盒测试中，测试任务可能是要对一段代码进行测试；在黑盒测试中，测试任务可能是对一些业务功能或性能进行测试。但最终都要归结到测试用例的执行上，所以本方案中所提到的执行测试脚本和测试用例就代表执行测试任务。

### 测试任务的依赖关系

测试任务具有不同于一般任务的特殊性，主要是由于测试任务的重要级别不同以及测试任务之间存在着依赖关系决定的。有的任务紧迫性比较强，所以它的重要级别就相对较高，在执行任务时就应考虑先执行它；而紧迫性不那么强的，重要级别就不那么高，自然也不会被优先考虑执行。另外，测试任务的重要级别还体现在任务本身的价值，有的测试模块是整个系统要实现的主要功能或关键环节，那么它的重要级别就相对较高。

依赖关系是测试任务区别于普通任务的关键，因为有些任务的输出结果将会作为其他任务的输入，或者会影响其他任务的执行。所以想要对测试任务进行调度，就必须考虑测试任务之间的依赖关系。测试任务中常见的依赖关系有下面几种：

（1）一对一依赖关系

一对一的依赖关系也称为单依赖关系，下图所示taski和taskj之间存在着单依赖关系，每个测试任务（除了第一个与最后一个）均存在前驱和后继，对其他测试任务有着直接影响。如图所示，测试任务taski的输出结果作为taskj的输入，taskj依赖于taski的实现。



图 6-4-2 一对一依赖关系

（2）一对多依赖关系

对多依赖关系是一种多依赖关系，下图所示taski和taskj（j=i+k，0<k≤n，）之间就存在着多依赖关系，taskj必须在taski执行之后才能执行。



图 6-4-3 一对多依赖关系

（3）多对一依赖关系

测试任务taski（i=1,2,…,n）是测试任务taskn+1的前驱，taskn+1在taski执行完毕后才开始执行，如下图所示，taski的输出结果作为taskn+1的输入，taskn+1依赖于taskj的实现。



图 6-4-4 多对一依赖关系

（4）多对多依赖关系

测试任务taski（i=1,2,…,m）是taskj（j=1,2,…,n）的前驱，taskj是taski全部执行完毕后才开始执行，taskj依赖于taski的实现，taski的长度制约着taskj的开始执行时间。



图 6-4-5 多对多依赖关系

通过以上测试任务依赖关系描述可知，测试任务间的依赖关系直接影响着整个测试工作的效率。在为测试任务调度虚拟机的过程中，如果考虑测试任务间的依赖关系将会避免虚拟机资源的重复调度，减少了资源的浪费。

### 测试任务调度器的结构模型

用户提交测试任务之后，根据任务的数量以及对测试项目的分析，预测出需要的虚拟机数量是不可缺少的，这样可以减少因请求过多虚拟机而造成的浪费，也可以避免测试执行过程中虚拟机太少重新向云资源管理器申请带来的时间浪费。下图6-4-6为测试任务调度子系统的基本结构。



图 6-4-6 测试任务调度子系统

在基于云平台的测试系统中，根据对资源提出请求角色的不同，将对资源的请求分为两类：

一是由测试执行人员提出对处理机、网络设备、内存和数据库等虚拟资源的请求；

二是系统根据用户提交的测试项目和需求制定好测试计划，测试服务开始执行后，由云监控模块根据执行情况向虚拟资源池提出资源请求。

云控制模块对不同测试项目规模中每类虚拟资源的请求设置一个最大使用量值，该值的设定根据预测公式得到，可以满足一般情况下对应规模的项目测试服务需求。

当测试执行人员对虚拟资源的请求数量在最大使用量值范围内，则分配对应的虚拟机给测试执行人员。

当请求数量超过最大使用量值时，向云管理模块发送消息，请求增加底层虚拟基础设施，云管理模块接收并处理消息，调度部署需求的虚拟资源来保证测试服务能够顺利执行。通过这种方法，可以降低因虚拟基础设施不足向管理模块发送消息带来的消耗，同时避免了虚拟资源分配过多造成的资源控制浪费。

对于测试服务执行时系统对资源池中虚拟资源的请求，把具有简单依赖关系的测试任务动态分配给适当的虚拟机。为平衡虚拟机和服务器的负载，调度系统包括四部分：

（1）预测规划模块

测试服务开始执行前，根据测试需求和测试项目的规模预测项目测试需要的各类虚拟机的数量，如果当前的虚拟机数量不够，发送消息给云资源管理系统，调度新的虚拟机。

需要的虚拟机预测公式如下：

其中：

n----当前的测试任务总数

m---云平台中适用这个任务环境的可用的虚拟资源数量

n0---完成的测试任务数量

d----测试任务的截止时间

r----任务的释放时间

c----上一个任务的完成时间

（2）测试任务调度模块

进行任务之间的任务级调度，选出一个测试任务。

（3）测试用例调度模块

测试任务内部进行用例级调度，调整测试用例执行顺序，保证缩短测试任务执行时间。

（4）虚拟机分配模块

任务级调度和虚拟机分配都是以测试任务为基本单位的，且在虚拟机分配完后需要启动任务执行进程，然后返回任务级调度进行下一个任务的调度，用例级调度在测试任务内部进行，它以测试用例为基本单位。

### 测试任务调度的算法模型

假设云中有n个测试任务Tasks={task1, task2,…,taskn}，m个虚拟机资源Vm={vm1, vm2,…,vmm}，测试任务调度的目标就是为测试任务taski（0<i≤n）选择虚拟机资源vmj（0<j≤m），建立测试任务与虚拟机的映射对应关系。

测试任务之间的依赖关系，可以表示为元素为0或1的方阵TDD：

即：

测试任务的执行状态用数组Alloted表示：

Allted={alloted0, alloted 1,…., alloted n}

allotedi取值如下：

开始时，需要计算测试任务在不同机器上执行所需的时间，EETij表示vmj独立执行taski所需的时间。

为测试任务分配合适的虚拟机以形成最佳的调度序列是测试任务调度的目标。

在调度算法中，给测试任务赋予一个优先权值，优先权值高的任务先执行。

优先权通常分为静态优先权和动态优先权。

静态优先权在测试任务提交时就确定，一直到执行结束保持不变。

动态优先权在任务的执行过程中不断地发生变化，通常运行时间长的任务优先权高，优先权值也会随着任务的等待时间的增加而变高。

基于优先权的任务调度算法的框架如下：

**算法：基于优先级的任务调度**

输入：EETnm—任务执行时间矩阵，任务数n，处理机数m

输出：taski到虚拟机vmj的映射

Step1:计算每个任务的优先级

Step2:计算每个任务的优先级

Step3:将任务按照优先级降序排序

Step4:对任务进行分组

Step5:对各组任务分别执行调度算法

下面给出任务调度算法的详细流程，如下图6-4-7所示：



图 6-4-7 任务调度算法流程

（1）在调度过程中，假设任务数和虚拟机数是有限的。设有n个独立的任务T={T1,T2,…, Tn}，由m个虚拟机M={M1,M2,…, Mn}执行。现约定，任何任务可以在任何一台虚拟机上执行，但未完工之前不允许中断处理。任何任务不能拆分成更小的子任务。第一次调度之后的每次调度过程中，等待执行的任务数赋值给n，空闲虚拟机数赋值给m。

定义两个队列：测试任务等待队列WaitList；空闲虚拟机队列FreeVMList。

（2）用户通过系统接口提交测试任务，并按提示给各任务赋权重值，加入测试任务等待队列WaitList。

用户在提交任务时，决定每个任务的权重，根据测试任务的等待时间，处理时间以及权重得到任务的优先权。计算公式如下：

πwj

式中，

qj---任务j的等待时间

dj---任务j处理时间；

wj---任务j的权重；

pj----任务j的优先级；

π---引进的参数，调节与wj，使两者处于同一数量级。

（3）根据对要测试的项目需求和用户提交的测试任务数量的分析，预测虚拟机的数量、配置等，向云资源管理器发送消息，申请虚拟机，然后根据环境配置模板自动部署操作系统、软件系统以及测试系统等完成测试环境的搭建。这样可以在测试执行过程中减少对新资源的请求。

（4）将队列WaitList中任务数赋给n，队列中空闲虚拟机数赋给m。

（5）当n≤m时，释放多余的处理机，将虚拟机Mi分配给任务Tj即可，并在测试任务执行完之后立即释放对应的处理机。

（6）当n>m时，根据优先权的计算公式，计算出所有任务的优先权。按优先权值降序排序，取WaitList中前m个任务分配给m个虚拟机。将任务T1到Tm从队列中WaitList删除，将虚拟机M1到Mm到从队列中删除。

（7）将用户添加的新任务和执行过程中出错的任务添加到任务等待队列WaitList中，等到有空闲虚拟机时，将空闲虚拟机加到FreeVMList队列中。

（8）重复执行步骤（4）直到所有任务都得到正确执行。

上述算法具有以下优点：

（1）当n≤m时，不需要考虑每个任务的优先权，总执行时间为运行时间最长的那个任务的运行时间，即需要时间O(1)。当n>m时，排序耗时O(nlogn)，循环耗时O(nlogm)，因此算法所需的时间是O(nlogn+nlogm)=O(nlogn)。

（2）每个任务执行完后立即释放虚拟机，云平台中的资源使用率得到提高。

（3）等待时间长，执行时间短，权重大的任务优先执行。

## 测试用例智能推荐技术

测试用例（Test Case）是为某个特殊目标而编制的一组测试输入、 执行条件以及预期结果， 以便测试某个程序路径或核实是否满足某个特定需求。RUP对测试用例的定义为：对一项特定的软件产品进行测试任务的描述，体现测试方案、方法、技术和策略。内容包括测试目标、测试环境、输入数据、测试步骤、预期结果、测试脚本等，并形成文档。

测试用例的优劣直接影响软件测试的质量，它对测试工作的控制和指导作用相当于设计文档对编码的指导作用，尤其在大系统中表现出对系统测试的权威性。

测试用例的复用（Test Case Reuse）属于软件复用中的测试信息的复用，是一个软件的测试用例在新的软件测试中使用。

测试云平台作为测试运行的基础平台，在进行了大量的被测对象测试后，会形成测试案例库，利用测试案例库进行测试用例智能推荐是本项目的特点之一，测试用例智能推荐技术主要采用基于案例的推理CBR技术。

### CBR预处理流程

在云平台中，测试知识库中的知识来源主要有三种：

（1）被测程序的软件规格说明书

测试工程师从软件规格说明书中分析、提取出相应的被测软件规则，存入知识库中。

（2）测试工程师的经验

测试工程师根据自己的以往经验，自己构造相应的规则或者测试权值，并存入知识库中。

（3）根据结构提取规则

利用已有的代码分析工具，分析代码，然后根据结构提取知识规则，这个一般用于单元测试用例的生成。

基于案例的推理CBR是人工智能领域新近崛起的一种重要推理模式，它基于相似和类比推理的一个独立分类，在知识难以表达，具有丰富经验知识的领域得到了广泛的应用。CBR通过修改重用以往解决问题的方案来解决当前问题的方法。它是一个包括案例检索、 案例重用、案例修改、案例存储的循环过程。

基于案例的推理是一个问题求解的方法，它把人的思维求解过程中的“提出问题一寻求方案一确定方案” 的过程在计算机中的循环实现。它意在结合以往的经验，获得新问题的求解。而这恰恰适合测试用例复用的要求。从另一方面讲，虽然软件测试领域存在大量可以复用的测试用例，但是，软件测试会因软件操作系统、运行环境、硬件系统、网络状态、 用户特点及使用习惯、系统版本等等而异。

软件测试用例设计者需要综合考虑各个方面的因素，才能设计出高效和高覆盖率的测试用例。这就注定了测试用例的复杂性，从而给测试用例的复用提出了巨大的挑战。这种复杂性决定了复用测试用例要综合各个方面，在已有复用经验和用例的基础上，给出最优的解决方案。

一个完整的测试用例内容包括测试目标、测试环境、 输入数据、测试步骤、预期结果、 测试脚本和测试文档。在CBR推理前，需要对测试用例进行改造（此过程可在设计测试用例阶段为方便复用而有计划的进行），保证推理的有效性。

测试用例复用的预处理流程如下图6-5-1所示：



图6-5-1 测试用例复用的预处理流程

（1）根据测试文档和测试目标，提取关键信息，作为测试目标关键字，作为条件匹配用例。提取的规则是用尽量简短丰富的文字描述测试的功能，即粒度小，数量多，涵盖精确全面。系统给出一个参考文件和一些示例，以帮助用户确定关键字。

（2）测试用例的设计者会根据测试目标，确定测试输入数据需求，而穷举测试不可能实现，因而他们会对测试数据有所取舍，以兼顾测试效率和覆盖率，这样数据本身不能全面反映需求。同时，对测试数据字段名称的定义也千差万别，如用户名可以定义为UseName，UserID等等。我们通过测试文档和测试数据，确定所有测试数据的类型和长度，仅用这两个方面作为条件匹配测试数据。这样消除以上因素影响的同时，提高了匹配的准确度和有效性。

（3）将测试环境、测试步骤和各步骤间的依赖关系，以及脚本语言、测试工具等等测试文档中对测试的明确要求提取出来，作为匹配条件。给所有匹配条件分配参考权重。 其中，关键字权重最高，其次是测试数据和脚本语言。其他条件可以根据经验或者系统给出的参考权重来设置。

（4）将以上获得的目标关键字、数据需求、匹配条件等作为标签，权重作为属性， 生成XML文档，作为用作检索时的模式文档。

（5）确定每个测试用例所包含文件的存储位置，建立文件索引，以便检索和修改测试数据、测试脚本等文件。

（6）用例存储模式可以用如下二元组表示：C=<用例检索模式文档， 文件索引＞， 这是CBR的核心，用XML文件来描述。

整个过程本质上是对测试用例进行处理，提取其关键信息并建立索引，为它建立一定的结构，便于存储和推理。

### CBR推理的结构框架

整个CBR系统以用例检索模式文档作为输入，经过系统实现的用例产生机，输出符合要求的结果测试用例。结构如下图6-5-2所示。



图6-5-2 CBR检索机制

系统的核心是用例产生机，其原理如下图所示。当用户输入新的用例检索文档后， 系统按如下图6-5-3机制处理：



图 6-5-3 CBR搜索处理机制

（1）搜索本地用例数据库，如果有完全相同的记录，则匹配成功，可直接获得结果用例。

（2）如果第一步匹配失败，则启动CBR，搜索相似用例。根据用例检索模式文档中的属性值和权重，进行推理。这里使用的相似性算法是最近邻近K-NN算法，系统会选择最大相似度用例，作为进行差异比较的对象。

考虑到对测试用例的复杂性带来的修改的复杂性，为了减小用例修改的成本，设置一个最小匹配度；如果推理匹配度小于此值，则系统失败退出；否则给出最匹配用例和新需求用例的差异文档并给出差异提示。

（3）用户根据差异提示，决定是否启用动态配置。

差异提示是差异文档在用户界面的显示；差异文档是新需求用例和CBR推理获得的最匹配用例存储模式XML文件比对的差异结果，其结构同用例存储模式文档。

（4）在启用动态配置进行自动修改结束后，系统会给出修改结果，包括测试数据和期望结果、测试流程、测试脚本等，提示用户进行确认，如果结果满足测试要求，则确认，否则对结果进行人工修改进行修正。这一过程主要是满足测试的准确性要求。

（5）人工修改过程相对复杂，需要用户根据测试需求调整最匹配用例的测试数据、 步骤和测试脚本，并根据这三者确定正确的期望结果。

在这几个过程中需要注意以下问题：

* 最小匹配度的设置问题

如果太小，会导致对用例的修改代价太高；而如果太大，则导致复用度太低，所以在设置时需要慎重，系统根据经验对不同的用例给出一个参考最小匹配度。

* 动态配置的启动问题

动态配置的启动问题实际是自动修改和人工修改的界限划分问题。启动的标准是： 如果仅对测试数据和测试步骤的配置，对测试脚本的改动较小，则启动动态配置；否则， 需要人工修改用例。这一过程充分体现了测试经验对于软件测试的重要性， 丰富的用例设计和测试配置经验会使得这一决策过程游刃有余。比如预期结果总体分为两类：一类为是/否（真/假、对/错）等判定性数据；另一类是测试数据经过软件作用后的真实结果。后者在多数情况下需要人工修改。

### 用例的检索模式文档的设计

为了更好地匹配已有的用例，项目用XML文件定义了一个搜索文模式作为匹配条件， 并为每一个条件确定了比重。其中，测试数据作为输入；而最重要的是测试目的的描述及其权重的定义，鉴于不同测试用例设计者对测试目的描述的差异性，引入了关键字，来更加精确地描述测试目的，同时，降低测试目的描述的权重，从而提高了关键字的权重。给出示例描述如下：

<?xml version=“1.0”encoding=“gb2312”?>search-condingtions xmlns:xsi=[http://www.w3.org/2001/XML Schema-instanee](http://www.w3.org/2001/XML%20Schema-instanee) xsi:noNamespaeeSchernal.ocation-“condition-schema.xsd”>

<case>

<purposes>

<purposes weight=“0.1”>登录验证</purpose>

</purposes>

<keys>

key weight=“1.0”登录</key>

</keys)

<os weight=“0.1”>Linux</os>

<env> weight=“0.5”>Apache 2.0</env>

<db weight=“0.5”>My SQL 8.0</db>

<arch weight=“0.3”>BS</arch>

<Inputs>

<Input weight=“0.5”>

<type> int</type>

<length>)4</length>

</input>

</inputs>

<netspeed weight=“0.5”) 1M</netspeed>

<maxuser weight=“0.5”)10</maxuser>

<coneurrents weight=“0.5”>10</concurrents>

<step weight=“0.5”>4</step>

<browser weight=“0.5” Firebox 2.0</browser>

<scriptlan weight=“0.5”>python</scriptlan>

<tool weight=“1.0”TeatMaker</tool>

</case)

</search\_condition>

### CBR的推理算法

假设用例库中用例X＝{x1，x2， …，xn｝和一个新的需求用例F={F1，F2，…，Fn}，Xi、Fi是搜索模式文档的n个属性的特征值，定义Xi和Fi的相似度如下面的公式：

对应的用例检索模式文档的相似度用下面的公式进行计算：

其中wi是该属性的权重，特殊情况下可以选择不为特征属性设定权值，这样，计算时将默认所有有效wi的值为1。因此，若两条用例特征完全相同，则用例的总相似度为1。属性值一致的属性个数越多，用例相似度也越高，同时伴随不一致属性个数的增多，其案例相似度也会降低。

在CBR推理算法中，采用KNN（K-Nearest Neghbor Approach）最近邻算法实现分类，其基本思想是：

在训练集中数据和标签已知的情况下，输入测试数据，将测试数据的特征与训练集中对应的特征进行相互比较，找到训练集中与之最为相似的前K个数据，则该测试数据对应的类别就是K个数据中出现次数最多的那个分类。

KNN算法的基本过程如下：

1. 计算测试对象到训练集中每个对象的距离，这里的距离就采用相似度计算公式所确定的公式；
2. 按照距离的远近排序；
3. 选取与当前测试对象最近的k的训练对象，作为该测试对象的邻居；
4. 统计这k个邻居的类别频率；
5. k个邻居里频率最高的类别，即为测试对象的类别。

简要描述KNN算法的过程就是找邻居，然后投票决定类别。

有了算法的思想，首先需要做的是将采集到的样本集做归一化处理，归一化处理0均值标准化处理，如下公式所示：

其中u为样本X的均值，为样本X的方差。

该方法将原始数据归一化为均值为0、方差为1的数据集。

下面给出KNN算法的Python语言实现。

|  |  |
| --- | --- |
|  | import csv, numpy as np |
|  | import matplotlib.pyplot as plt |
|  | from mpl\_toolkits.mplot3d import Axes3D |
|  | class KNearestNeighbor(object): |
|  | def \_\_init\_\_(self): |
|  | self.Run(split=0.75, k=10) |
|  | # 样本集数据归一化 |
|  | def toOneDaDataset(self): |
|  | temp = np.array(np.array(self.dataset)[:, 0:-1], dtype=np.float) |
|  | # temp = ((temp - temp.mean()) / temp.std()).tolist() # Z-score标准化 |
|  | temp = ((temp - temp.min()) / (temp.max() - temp.min())).tolist() # min-max标准化 |
|  | [line.append(self.dataset[idx][-1]) for idx, line in enumerate(temp)] |
|  | return temp |
|  | # 导入解析数据 |
|  | def loadDataset(self, filename, split): |
|  | with open(filename, 'r') as csvfile: |
|  | self.dataset = list(csv.reader(csvfile)) |
|  | np.random.shuffle(self.dataset) |
|  | self.dataset = self.toOneDaDataset() |
|  | datasetSize = len(self.dataset) |
|  | trainSet = self.dataset[0:int(split \* datasetSize)] |
|  | testSet = self.dataset[int(split \* datasetSize):] |
|  | return trainSet, testSet |
|  | # 计算两个向量的欧式距离 |
|  | def calculateDistance(self, testInstance, trainInstance): |
|  | test, train = np.array(testInstance[0:-1], dtype=np.float), np.array(trainInstance[0:-1], dtype=np.float) |
|  | return np.sqrt(np.sum((test - train)\*\*2)) |
|  | # 根据距离向量组，找到k个最近的邻居 |
|  | def getNeighbors(self, trainSet, testInstance, k): |
|  | distances = [] |
|  | # 对训练集的每一个向量计算其到测试集的欧式距离 |
|  | for x in range(len(trainSet)): |
|  | dist = self.calculateDistance(testInstance, trainSet[x]) |
|  | distances.append((trainSet[x], dist)) |
|  | distances = sorted(distances, key=lambda d: d[1]) # 距离排序 |
|  | neighbors = [] |
|  | [neighbors.append(distances[x][0]) for x in range(k)] # 选取k个邻居 |
|  | return neighbors |
|  | # k个邻居推断测试样本的类别 |
|  | def getLabel(self, neighbors): |
|  | classVotes = {} |
|  | for x in range(len(neighbors)): |
|  | label = neighbors[x][-1] |
|  | if label in classVotes: |
|  | classVotes[label] += 1 |
|  | else: |
|  | classVotes[label] = 1 |
|  | # 以少数服从多数倒叙排序Label |
|  | sortedVotes = sorted(classVotes.items(), key=lambda cls: cls[1], reverse=True) |
|  | return sortedVotes[0][0] # 选取类别频率最大的的作为测试样本的类别 |
|  | # 算法准确率计算 |
|  | def getAccuracy(self, testSet, labels): |
|  | correct = 0 |
|  | for x in range(len(testSet)): |
|  | if testSet[x][-1] == labels[x]: |
|  | correct += 1 |
|  | print('在{0}个测试数据中，共有{1}个标签预测正确'.format(len(testSet), correct)) |
|  | print('算法准确度为{0}%'.format(round((correct / float(len(testSet))) \* 100.0, 2))) |
|  | # 执行算法步骤 |
|  | def Run(self, split, k): |
|  | trainSet, testSet = self.loadDataset('data/testKNN.csv', split) |
|  | labels = [] |
|  | for x in range(len(testSet)): |
|  | neighbors = self.getNeighbors(trainSet, testSet[x], k) |
|  | Labels.append(self.getLabel(neighbors)) |
|  | self.getAccuracy(testSet, labels) |
|  | if \_\_name\_\_ == '\_\_main\_\_': |
|  | knn = KNearestNeighbor() |

### 测试案例库的学习

测试案例的学习与保存是CBR的一个优点， 是CBR系统保持知识更新与经验积累的关键所在。在对历史案例完成修改的基础上，得出新案例，此时，对新案例主要有三种处理方式：

* 新增

在保留历史案例基础上， 将新案例添加进案例库；

* 替代

将修正后的新案例替换历史案例，并将新案例更新入库，修正前的原历史案例不复存在；

* 丢弃

待新问题解决完毕，只保留原历史案例，并将修正后的新案例作为中间结果丢弃。

对测试案例的处理方案因情况而异，然而，如若处理不当，却容易导致案例库案例冗余、案例残缺、垃圾案例堆积等问题，影响案例复用是实际效果，在此基础上，一方面需要依赖复用者的经验，另一方面也可以通过一定手段使案例的学习更有效。

对案例的学习采用以下规则策略：

设有测试案例库C=(X1，X2，…，Xn)，，Xi代表某一历史案例，新案例N与历史案例Xi的相似度为，，K为专家设定的一个阈值，测试案例的学习策略如下所示：

（1），案例库中没有与新案例相似的历史案例，那么直接将新案例新增到案例库中；

（2），新案例与案例库中某历史案例完全相似，则保留原历史案例，新案例不加入案例库中；

（3），案例库中不存在与新案例相似度大于相似度阈值K的历史案例，则保留所有原案例，并将新案例新增入库；

（4），根据新旧案例差异完善原案例，并将改写合并后的案例更新入库。

### 基于软件需求的用例推荐

软件需求指导软件的开发，同时也是软件测试的依据，完善的测试应该覆盖全部软件需求。通过需求分析，将软件需求中的所有信息挖掘出来，按照需求之间的相似性进行分类，对于相似的需求，测试用例具有相似性。

软件需求与测试用例之间是密切联系的，一个软件需求可以对应一个测试用例，也可以对应多个测试用例。同样的，一个测试用例有可能覆盖一个软件需求，也可以覆盖多个软件需求。软件需求与测试用例之间的关系如下图6-5-4所示。



图 6-5-4 软件需求与测试用例关系

尽管软件需求与软件测试用例之间并不是严格的一一对应，但通过软件需求可以确定与之对应的软件测试用例。基于软件需求的测试用例复用的原理如下图6-5-5所示。



图6-5-5 测试用例复用原理

从上图可以看出实现测试用例推荐的关键点是对软件需求进行分类，基于软件需求分类的测试用例复用推荐框架如下图6-5-6所示。



图 6-5-6 测试用例复用框架

该框架的主要步骤如下：

（1）需求预处理

分别对已测软件需求预待测软件需求进行分词，去停用词等方法进行预处理操作。

（2）软件需求特征提取

对于已测软件需求，经过需求预处理之后，特征向量空间的高维问题还没有得到有效解决，特征选择作为特征降维的有效手段在软件需求分类中起着至关重要的作用。特征选择的作用是通过特征降维来加快整个系统学习的过程，从而提高软件需求分类的性能。

（3）需求向量化

对于待测软件和已测软件，根据禁忌搜索算法特征选择后得到的特征向量空间，进行需求向量化表示。

（4）待测软件分类

使用KNN分类器对待测软件需求进行分类，得到待测软件所属类别，并通过相似性计算得到已测软件中最相似的软件需求。

（5）测试用例复用推荐

通过需求分类，从已测软件中得到待测软件最相似的需求，将已测用例直接或间接应用于待测软件。

由于软件需求一般使用的是使用自然语言描述，所以第一步就是将自然语言转换为机器识别的信息。其基本的思想是将自然语言描述的软件需求转化为词向量。

词向量通过使用连续稠密的向量来刻画一个词的特征，不仅可以刻画词和词之间的相似性，还可以建立一个从向量到概率的平滑函数模型，使得相似的词向量可以映射到相近的概率空间上。

向量空间模型（Vector Space Model，以下简称VSM）基于统计语言假设：语言的统计特征隐藏着语义的信息。例如，两篇具有相似词分布的文档被认为是有着相近的主题，此假设产生很多衍生版本，如一篇文档的词频（而不是词序）代表了文档的主题；上下文环境相似的两个词有着相近的语义都是语言的统计特征。

设S是一段文本序列,它的概率可表示为：

因此，可以将文本序列的联合概率转化为一系列条件概率的乘积，因此对于文本处理的问题就变成如何去预测在已知一些词情况下的某个词的条件概率，上述公式可以简化为：

当N=2时，称为bigram模型，当N=3时，称为trigram模型。因此上述模型简写为Ngram模型，Ngram模型存在着状态空间爆炸的问题。能否用一个连续的稠密向量去刻画一个词的特征呢。Word2Vec就是用来处理这个问题的一个人工智能框架，它是一个神经网络语言模型（Neural Network Language Model,NNLM），NNLM的基本思想是：

（1）假定词表中的每一个词对应着一个连续的特征向量；

（2）假定一个连续平滑的概率模型，输入一段词向量的序列，可以输出这段序列的联合概率；

（3）神经网络将同时学习词向量的权重和概率模型里的参数。

下图6-5-7是一个NNLM的网络模型，此模型是一个简单的前向反馈神经网络来拟合一个词序列的条件概率。



图6-5-7 NNLM网络模型

在该模型中，首先是一个线性的词嵌入层（embedding），它将输入的N-1个one-hot词向量，通过一个共享的D×V的矩阵C，映射为N-1个分布式词向量（distributed vector）。其中，V是词典的大小，D是embedding向量的维度，C矩阵中存储了要学习的词向量word vector。其次是一个简单的前向反馈神经网络g，它由一个tanh的隐层和一个softmax的输出层组成。

通过将embedding层输出的N-1个词向量映射为一个长度为V的概率分布向量，从而对词典中的Word在输入context下的条件概率做出预估：

然后通过最小化一个正则化损失函数来调整模型的参数：

其中，模型的参数包括了embedding层矩阵C的元素和前向神经网络模型g的权重。这个模型同时解决了两个模型，一个是统计语言模型里关注的条件概率的计算，一个是向量空间模型里关注的词向量的表达。

采用Word2Vec计算词向量的过程为：

（1）选择语料库文件

（2）对语料库文件进行切词

（3）对切词后的文件，去除多余的其它字符

（4）将切词整理后的文件输入到Word2Vec中进行训练，得到模型

（5）装载训练好的模型，利用训练好的模型可以输入Word，可以得到词向量

从软件需求规格说明中抽取到的所有软件需求描述作为原始语料库文件A.txt；

使用JieBa切词将语料库文件切词形成B.txt，对于B.txt中的每一行linestr，切词的方法为：

jieba.cut(linestr,cut\_all=True)

从B.txt文件中去除特殊字符，得到可以训练的语料库文件B1.txt；

对语料库文件B1.txt进行训练，得到词向量模型文件B1.bin，方法如下：

word2vec(‘B1.txt’,’B1.bin’,size=20,verbose=True)

模型训练的结果为B1.bin，要查看该模型所对应出现的词向量，过程如下：

使用load装载训练好的模型model=word2vec.load(‘B1.bin’)，则model中的词向量表示为model.vectors。

对失效模式数据进行归类时，涉及到对软件问题描述进行相似性计算，句子相似性计算时，有以下几种方法：

* 编辑距离计算方法
* 杰卡德系数计算
* TF计算：term frequency（词频计算）
* TFIDF计算：IDF:inverse document frequency（逆文档频率）
* Word2Vec词向量计算

在本项目中，在Word2Vec词向量计算的基础上，利用余弦计算方法求取软件问题描述的相似性，设段落a的向量为、段落b的向量为,则a与b之间的距离定义为：

如何将需求描述中所有词的词向量转化为需求的向量呢？方法是将需求描述中的词向量使用TFIDF值进行加权平均。需求描述A中由n个词组成，词Ci的向量为VCi，则需求描述段落的向量为：

其中wi为词的TFIDF值，TF称为词频，IDF称为逆文档词频。

如果一个词很重要，它应该在文中多次出现，于是需要进行词频的统计，但词频统计时，由于一些副词如“的”、“是”、“在”等一类的词对分析语义毫无帮助，这些词被归为停用词（stop words），把停用词均过滤掉之后，只考虑剩下的有意义的词。这样又会出现一个问题，比如我们可能发现“软件”、“协议”、“串口”这三个词出现的次数一样多。这是否意味着这三个词的重要性一样呢？显然不是如此，因为“软件”是很常见的词，相对而言“协议”和“串口”则更能反映一段文字的重要性。如果这三个词在一段文字中出现的次数一样多，有理由认为，“协议”和“串口”的重要程度要大于“软件”，也就是说在关键词的排序上，“协议”和“串口”应该排在“软件”的前面。所以，我们需要一个重要性调整系数，衡量一个词是不是常见词。

如果某个词比较少见，但它在段落中多次出现，那么它很可能就反映了这段文字的特性，这正是我们所需要的关键词。用统计学表达，就是在词频的基础上，要对每个词分配一个“重要性”权重，最常见的词（“的”、“是”、“在”）给予最小的权重，较常见的词（“软件”）给予较小权重，较少见的词（“协议”、“串口”）给予较大的权重，这个权重叫做“逆文档频率”。

有了“词频”（TF）和“逆文档词频”（IDF）以后，将这两个值相乘，就得到一个词的TFIDF值。当某个词对段落的重要性越大，它的TFIDF值就越大。

如果一个词越常见，那么分母就越大，逆文档频率就越小。从TFIDF的定义可以看出，TFIDF与一个词在段落中出现的次数成正比，与该词在整个语言中出现的次数成反比。

通过需求描述相似性的判断，可以将收集到的已测系统软件需求项，按照相似度聚类，聚类后可以与待测软件需求进行匹配搜索。

有了词向量与软件需求相似性的计算方法之后，对待测软件就可以进行数据整理和分析归类了，其主要流程如下：

（1）从已测的软件需求中采集需求描述信息。

（2）抽取所有已测软件的软件需求，形成“.txt”格式的文件Corpus.txt，该文件被称为语料库文件。

（3）对语料库文件Corpus.txt进行切词，形成语料库切词文件Corpus\_cut.txt。

（4）从语料库切词文件Corpus\_cut.txt中去除停用词，得到用于进行词向量模型训练的训练语料库文件TCorpus.txt。

（5）对训练语料库文件TCorpus.txt进行Word2Vec词向量模型的学习训练，得到训练成熟的模型Model。

（5）对每一已测软件需求的描述段落Paragraph进行切词，去除停用词，得到软件需求描述的切词文件ParaCut.txt。

（6）计算切词文件ParaCut.txt每个词的TFIDF值，对每个软件需求描述文件ParaCut.txt，求取其软件需求的向量ParaVec。

（7）对软件需求的向量ParaVec进行聚类，每一类为ParaVecC。

（8）对于新的待测软件需求，计算该软件需求的词向量值，使用KNN算法进行归类。

（9）对于归入某一类的软件需求，搜索查找该需求对应的测试用例进行测试用例推荐。

# 云测平台与测试工具的经验与技术积累

北京旋极信息技术有限公司从XX年开始就从事软件测试工具、云平台、大数据、人工智能系统等相关的研发工作，拥有XX项发明专利、XX项软件著作权，建立有CNAS软件测评实验室，开展有XXXX、、、XXXX、等软件测试工作，现将典型的研发经验与技术积累介绍如下：

*这部分旋极信息提供*

## 软件测试工具的研发案例介绍

介绍所开发的软件测试工具，什么时候开发的，取得了什么成果，卖给了谁，相关证书等。工具大体的介绍

## 私有云平台的研发案例介绍

## 大数据的研发案例介绍

## CNAS软件测评实验室建设案例

自己的软件测评实验室介绍：成立时间、人员多少、所建立体系组成，经营模式。

## 软件测试案例介绍

做过哪些测试，测试的基本情况介绍，被测软件特征，取得的效果等。

# 测试工具选型与设计

## 测试工具的分类

根据业主单位对软件测试工具的需求，软件测试工具分为白盒测试工具、黑盒测试工具、灰盒测试、测试设计工具4大类共9种，如下图8-1所示。

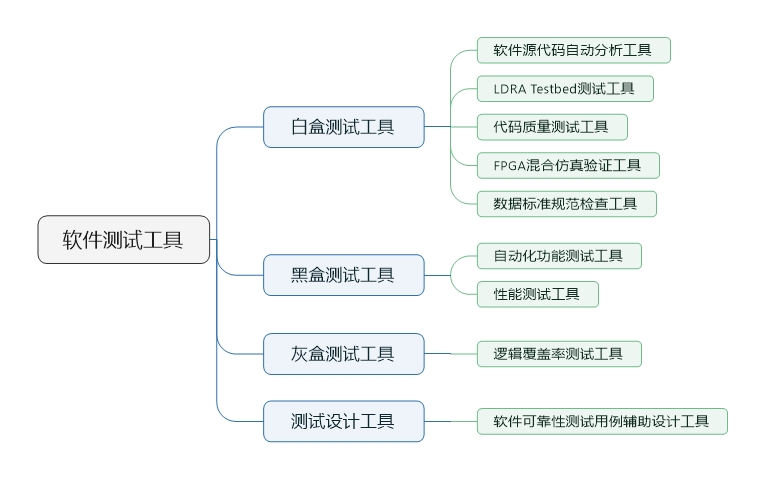
**

图8-1 测试工具的分类

（1）白盒测试工具

该类测试工具必须依赖于分析被测对象的内部结构和编码特征，主要的技术特征是利用代码编码与分析技术，开展软件测试的工作，属于白盒测试的工具为：

* 软件源代码自动分析工具
* LDRA Testbed测试工具
* 代码质量测试工具
* FPGA混合仿真验证工具
* 数据标准规范检查工具

（2）黑盒测试工具

该类测试工具不依赖于对被测对象内部结构特征和编码特征的分析，其主要的技术特征是模拟数据输入到被测系统中，观察被测系统的运行，以判断被测系统的运行情况，被测软件需要全要素动态运行起来，属于黑盒测试工具的为：

* 自动化功能测试工具
* 性能测试工具

（3）灰盒测试工具

该类测试工具兼具白盒测试工具与黑盒测试工具的特点，其技术特征既要能分析被测系统的结构信息和编码信息，又要能在被测系统全要素运行的状态下通过测试的数据结果，判断测试的运行效果，属于灰盒测试工具的为：

* 逻辑覆盖率测试工具

（4）测试设计工具

在白盒测试工具、黑盒测试工具、灰盒测试工具中，都是直接进行测试运行的工具，虽然这些工具也兼具测试设计的功能，但这些工具中的设计通常都是基于对被测系统的自动分析开展的，虽自动化程度较高，但测试设计的针对性则较差，而且偏重于从被测系统的代码特征开展测试设计。

在装备软件的验收性测试中，可靠性指标是用户最为关心的技术指标，针对可靠性指标回答的要求，需要开展有针对性的测试设计工作，属于测试设计工具的为：

* 软件可靠性测试用例辅助设计工具

## 软件测试工具选型与设计的原则

北京旋极信息技术股份有限公司作为专业的软件测试系统及工具提供商，基于多年的软件测试工具供货和提供经验，对于本次招标的软件测试工具，采用成熟性、行业认同度高、开放性三大原则进行选型和设计：

（1）成熟性原则

有成熟的商用软件测试工具，则选用成熟的软件测试工具，成熟的主要标准是软件版本发展已经历很多个版本轮次的开发，已形成至少3个版本以上的发布，软件测试工具的功能、性能都经过众多测试和实际应用的考验。

（2）行业认同度高

优先选用国内外行业认同度高的软件测试工具，要求该工具的市场占有和在行业中的应用都比较高，具有丰富的应用案例，这样可以获得更多的学习资源、减少学习的曲线，利于使用测试工具开展测试工作。

（3）开放性原则

由于所选用的软件测试工具需要实现在测试云平台中的集成，软件测试工具具有开放性编程的接口。当商用的软件测试工具完全不满足要求，同时新开发难度较小时，优先选用自主开发的模式开发相应的软件测试工具。

根据上述三大原则，确定出业主方所需要的9个软件测试工具的选型和开发模式如下表XX所示。

表1 软件测试工具选型与自主设计情况

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 软件测试工具 | 测试工具类型 | 语言特征 | 供货模式 |
| 1 | 自动化功能测试工具 | 黑盒 | 非嵌入 | 选型产品 |
| 2 | 性能测试工具 | 黑盒 | 非嵌入 | 选型产品 |
| 3 | 软件源代码自动分析工具 | 11111 |  | 选型产品 |
| 4 | LDRA Testbed测试工具 |  |  | 选型产品 |
| 5 | 软件可靠性测试用例辅助设计工具 | 111111 |  | 选型产品 |
| 6 | 代码质量测试工具 |  |  | 选型产品 |
| 7 | 数据标准规范检查工具 | 白盒 | 非嵌入 | 自主研发 |
| 8 | 逻辑覆盖率测试工具 | 灰盒 | 嵌入式 | 选型产品 |
| 9 | FPGA混合仿真验证工具 | 白盒 | 嵌入式 | 选型产品 |

## 自动化功能测试工具选型

自动化测试工具的主要含义：？

自动化功能测试工具的国内外厂商都有哪些？把每个厂家的简要介绍一遍

再把这个厂家工具简要介绍一遍？

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 工具名称 | 厂商 | 国别 | 主要功能 | 市场占有率 |
| 1 |  |  |  |  | 高、中、低 |
| 2 |  |  |  |  |  |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
| 5 |  |  |  |  |  |
| 6 |  |  |  |  |  |
| 7 |  |  |  |  |  |
| 8 |  |  |  |  |  |
| 9 |  |  |  |  |  |

工具的功能对比及指标的满足情况。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 功能指标 | 工具1 | 工具2 | 工具2 |  |
| 1 |  |  |  |  |  |
| 2 |  |  |  |  |  |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
| 5 |  |  |  |  |  |
| 6 |  |  |  |  |  |
| 7 |  |  |  |  |  |
| 8 |  |  |  |  |  |
| 9 | 性价比 |  |  |  |  |

最终建议选用的工具列表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 软件测试工具类型 | 测试工具名称 | 厂商 | 软件运行模式 | License数量 |
| 1 |  |  |  | 单机版UKEY、软key | 2 |
| 2 |  |  |  | 服务器板 | 1 |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
| 5 |  |  |  |  |  |
| 6 |  |  |  |  |  |
| 7 |  |  |  |  |  |
| 8 |  |  |  |  |  |
| 9 |  |  |  |  |  |

备注：单机版License共2个，1个在云平台中使用，1个在单机上使用

## XXX测试工具选型

## 数据库标准规范检查工具设计

在信息系统开发中，普遍采用关系型数据库系统进行数据的设计，而数据库的逻辑设计好坏影响着数据库及其应用系统的整体性能，决定了数据的完整性、准确性和一致性能否得到保证。如果数据库逻辑设计不合理，那么数据库的调优，对于数据库的性能提升将十分有限。

数据库标准规范检查工具的主要功能是：根据预定义规则文档，对数据库的表结构和表数据进行特定规则的标准检查，高效的查看具体数据库的落实情况，检查数据库逻辑设计是否合理，显示不合理的逻辑结构设计，对数据库表、字段、类型进行检查，检查表字段是否存在必要的字典等情况。

### 关系型数据库设计规范

关系型数据库是信息应用系统的核心组成部分，不合理的关系型数据库设计会加大编程难度，引起操作繁琐、性能 降低、空间浪费等不良后果，甚至影响应用系统的安全与稳定，合理设计关系型数据库十分有必要。而逻辑结构的设计是关系型数据库设计过程中的重要环节，逻辑结构设计的好坏直接决定并影响了数据的完整性、准确性与一致性。

关系型数据库设计存在5个方面的设计规范，分别为：

（1）数据完整性设计规范

* 每张表必须定义主键
* 某一字段如果为另一表的主键，则该字段应定义为外键；
* 表中字段的类型定义必须与其实际含义和可能的值匹配；
* 对表中的字段应根据实际使用要求对字段属性进行约束，如非空、唯一性约束等；
* 如某一字段值需满足特定要求，例如有特定的取值范围等，则应为其定义合适的约束条件；

（2）表逻辑设计规范

* 应采用第三范式（3NF）的设计方法，最大化保证数据的完整性；
* 对于频繁使用的表（如字典表），在内存空闲空间较多且数据量不大如不超1000行时，可采用缓存的方式保存，提升表的使用性能；
* 创建表时需指定到相应的数据表空间，确保不同数据存储在不同的表空间；
* 主键字段个数不能过多，尽量不修改主键值；
* 在数据库中实现数据完整性的校验，避免在应用中对数据进行完整性校验；
* 避免使用字符类型存放时间或日期类数据；
* 避免使用字符类型存放数值类型的数据；
* 避免表中字段数值类型直接使用INT型，应明确写明字段的取值范围；
* 应尽量减少使用大字段，如BLOB，CLOB，LONG，TEXT与IMAGE等。

（3）表分区设计规范

* 应对有需要的大表进行分区，以提高性能和可维护性；
* 当表的大小接近或超过4GB时，可考虑对其进行分区；
* 对于OLTP系统，当表的数据量非常庞大时，应考虑对表进行分区，对于硬件性能较好的服务器，可适度放宽要求。

（4）字段逻辑设计规范

* 应对通信地址等特定的信息采用多个字段来表示，增加灵活性；
* 应使用角色实体来定义关联属性，方便创建时间关联关系；
* 数字类型与文本类型的字段长度应保证充足；
* 对删除记录的操作需用统一的特定字段标注，而非直接删除记录。

（5）索引设计规范

* 不要索引大型字符字段；
* 不要索引常用的小型表；
* DML操作频繁的表应尽量少建索引；
* 尽量不要将经常修改的字段作为索引字段；
* 选择性高的字段适合建立索引；
* 如某一字段或字段组合经常在WHERE子句中使用并且满足该字段或字段组合查询条件的行数占表总行数的比例小于等于5%时，适合创建索引；
* 应在频繁使用DISTICNCT关键字查询的字段上建立索引；
* 进行表连接时，应在连接字段上建立索引。
* 复合索引创建时应把最常用的字段放在第一位，而将不常用的字段排后；
* 当索引字段的记录重复较多而DISTINCT记录值又较少时，应建立位图索引；
* 尽量避免对OLTP系统的一张数据库表创建过多索引，例如超过10个；
* 由于位图索引会影响DML操作的速度，因此OLTP系统中尽量不使用位图索引；
* 如查询中需使用函数，且满足该查询条件的记录 数比例很小，建议创建相应的函数索引。

创建索引完毕后，正确使用索引才能使其发挥作用，使用索引的规范如下：

* 应避免在字段上进行类型转换操作，否则无法使用该字段上的索引；
* 避免对索引字段进行任何计算操作，对索引字段的计算操作会引起索引的失效；
* 尽可能增加查询的条件，限制全范围的查询，当索引效率很低时，应避免使用索引；
* 尽量避免模糊查询，如必须使用模糊查询时，尽量使用前端匹配的模糊查询；
* WHERE条件中对索引字段尽量使用等值“＝”进行比较查询；
* 尽量使用前导字段（复合索引的首字段）作为查询条件；
* WHERE条件中对索引字段的查询条件应保证比较值的类型和字段类型一致。

索引在创建和维护过程中需注意如下事项：

* 创建索引时数据和索引应放在不同的表空间；
* 创建分区表索引时，尽量创建本地索引；
* 对于经常执行删除操作的表上的索引应定期重建索引；
* 由于索引重建时会阻塞 DML操作，应选择在业务空闲时进行，尽量减少其对业务产生影响；
* 对于键值频繁更新的索引，也应定期进行重建；
* 删除无用的索引，避免多余索引降低数据库DML操作的执行速度。

### 数据库标准规范检查工具的需求分析

数据库标准规范检查工具核心的功能需求是对数据库关系设计进行检查，检查其是否满足关系型规则的设计规范，根据对使用需求的分析，数据库标准检查工具的需求主要如下：

1. 功能性需求

* 具有标准库表的维护功能，主要包括标准库的创建、修改、查看、查询、删除等功能；
* 具有检查规则集的维护功能，主要包括规则集的创建、修改、新增规则、删除规则等功能；
* 具有数据源的选择与创建功能，可以进行数据源的连通性测试；
* 具有数据库规范检查功能，可以进行关系型数据库的设计规范检查；
* 具有检查结果展示功能，用图形化和表格化的方式展示出检查结果；
* 具有测试报告模板管理和测试报告输出功能。

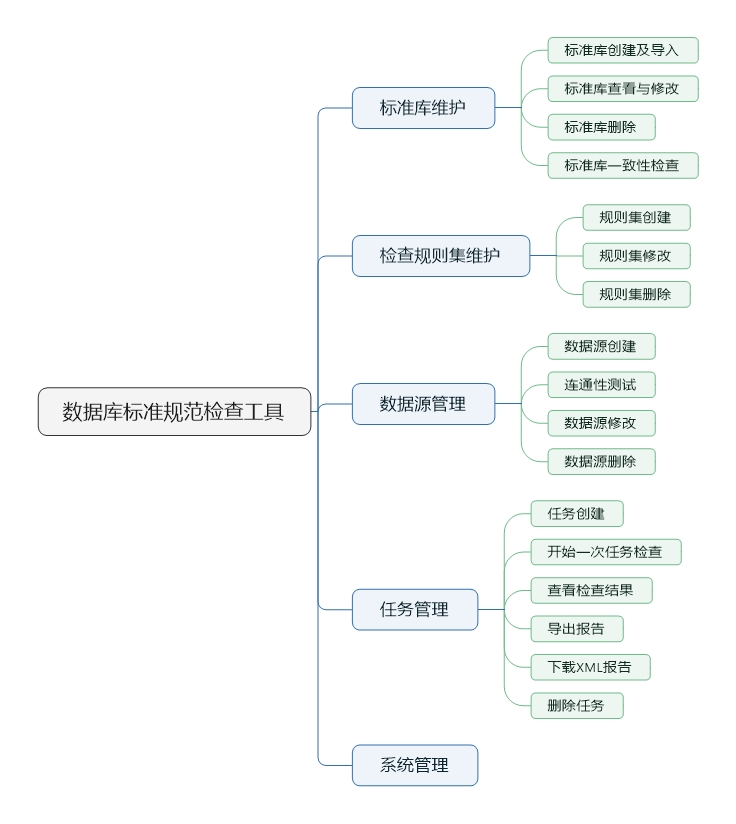
1. 性能需求

性能需求按照业主方所提供的技术指标，主要包括以下几点：

* 可支持 Oracle、 SQL Sever、 MYSQL、达梦、金仓等多种关系型数据库
* 支持多种数据标准
* 单检查规则测试时间：在100M局域网环境中，在标准库数据表中数据小于1000个时，单检査规则测试每分钟检查不少于100张数据表。
* 测试报告导出时间：在问题数小于1000个时，测试报告导出时间小于3分钟。

### 数据库标准规范检查工具功能设计

按照对数据库标准规范检查工具的需求与指标的分析，我们所开发的X-DBCheck的功能组成如下图XX所示。



图XX X-DBCheck功能组成图

系统主要由标准库维护、检查规则集维护、数据源管理、任务管理以及系统管理5个功能模块。

* 标准库维护

对数据库标准库进行维护和管理。可以进行标准库的创建、导入、查看、修改、删除和标准库一致性检查。标准库导入可以从文档中导入。标准分为数据项标准和数据字典标准两种。已经上传的标准再次上传会进行增量上传。

* 检查规则集维护

用于维护使用的规则集。包括了规则集的创建、修改和删除功能。规则集创建界面可以填写规则集名称、规则集描述、可以选择规则、设置规则的严重等级。选择规则从预置的总规则集中进行勾选。

* 数据源管理

用于维护需要进行检查的数据库数据源。包括了数据源的创建、修改、删除功能和连通性测试功能。数据源添加界面可以填写基本信息（数据源名称、适用项目、选择数据源类型（支持oracle和mysql）、用户名（数据库）、密码（数据库）、数据源连接、描述）等信息。

* 任务管理

用于对规则检查任务进行管理。包括了任务创建、开始检查、查看结果、导出报告、下载XML报告、删除任务等功能。创建任务时需要填写以下信息：任务名称、任务标识、数据源、选择规则集、选择标准库。

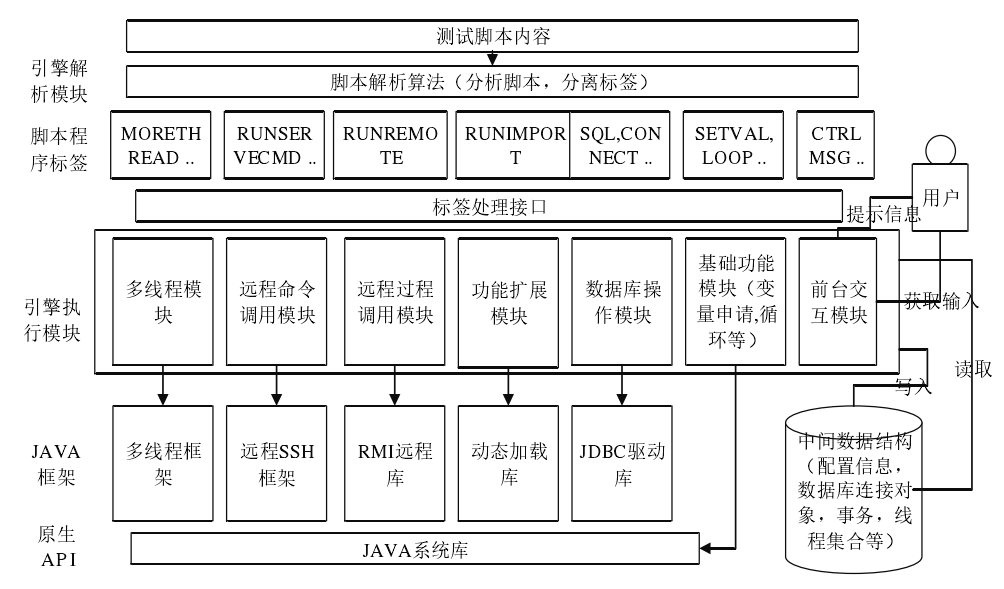
* 系统管理

包括了系统基础信息管理功能，主要有：报告模板管理、用户管理、授权管理、数据备份功能。

### 数据库标准规范检查工具逻辑设计

数据库标准规范检查工具为新开发的软件，为保证该工具达到SaaS软件所要求的第3级成熟度的要求，通过架构设计来保证该工具能天然适合于进行云化集成。

工具的主体部分为数据库检查的脚本解析引擎，所有脚本的运行都通过引擎运行，脚本化的方式确保了云化集成的方便性和可行性，从上到下为原生API、Java运行框架、引擎执行模块、脚本程序标签和引擎解析模块。如下图XX所示。



图XX X-DBCheck逻辑结构图

* 最高层

脚本程序标签层，该层对应脚本中设计好的所有标签，每个标签都是完成一个功能。此层最接近XML脚本程序代码，脚本中的语法和各类标签都是直接会被标签处理模块识别并调用中间层的测试引擎模块。

* 中间层

引擎执行模块，该层是脚本语言与JAVA平台的连接层，也就是脚本需要完成的功能都要通过中间层的各个模块来与JAVA平台交互，从而实现脚本中要完成的功能。也就是脚本的实现在底层都是借助于JAVA平台的实现。

* JAVA平台框架

由于我们是针对数据库测试，并且需要操作系统做交互，所以很多测试功能为了达到安全可靠，我们的中间层测试引擎模块都是直接与JAVA平台己经有的一些框架和库来交互，JAVA平台框架能够提供更好的对某个测试功能的封装，比如借助一些己有的多线程框架，可以方便的获取线程对象，创建线程池等。

引擎解析、执行模块作为SOA服务安装部署到服务器中，服务为一个多租户的服务模型。

### 对技术指标满足情况的论证

（1）可支持 Oracle、 SQL Sever、 MYSQL、达梦、金仓等多种关系型数据库。

Oracle、 SQL Sever、 MYSQL数据库为常见的国外数据库，达梦、金仓数据库为国产数据库，但无论是国外的数据库还是国产的数据，这几种数据库都是关系型数据库，数据库的访问接口形式都符合关系型数据库的标准，所有的数据库都提供了JDBC的访问接口，对表空间的读取都可以通过SQL语句来进行。

在X-DBCheck的系统架构中，使用Java框架来访问数据库，访问的接口都一致，因此对该项指标的要求满足没有问题。

（2）支持多种数据标准。

X-DBCheck提供了规则检查标准的维护功能，可以实现对不同数据标准的维护，只要甲方提供出数据标准，就可以将数据标准通过规则维护的方式形成检查的标准，以此来支持对多种数据标准的检查。同时，系统还提供了符合全军数据工程所要求的211K、装备数据库标准等多项标准的规则集。

（3）单检查规则测试时间：在100M局域网环境中，在标准库数据表中数据小于1000个时，单检査规则测试每分钟检查不少于100张数据表。

该项为性能指标要求，由于我们所开发的X-DBCheck为完全可以云化的产品，作为SOA服务提供，我们在100M局域网环境下对我们所开发的脚本解析与执行引擎进行了测试，测试结果如下表XX所示。

这部分请云祥编个数吧

表XXX 单检查规则测试时间

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 检查的规则 | 数据库表个数 | 数据记录量 | 最大时间 | 最小时间 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

从测试的结果来看，测试执行引擎的速度远远大于100张数据表/每分钟的要求。

（4）测试报告导出时间：在问题数小于1000个时,测试报告导出时间小于3分钟。

我们的X-DBCheck生成测试报告使用快速XML技术，该技术的主要特征是：

。。。。请张洪请把这个技术写一段把，半页6-7行就够

下表是我们生成含有1000个问题的Word报告所需要的时间。

# 系统的应用模式

刘敬民搞（5页以上）

# 系统的使用要求

系统的使用要求指标分环境适应性要求、可靠性要求、维修性要求和保障性要求4个方面的技术指标。

## 环境适应性要求与设计

环境适应性（Environmental Adaptation）是指产品在服役过程中的综合环境因素作用下能实现所有预定的性能和功能且不被破坏的能力，是产品对环境适应能力的具体体现，是一种重要的质量特性。

业主方的环境适应性要求如下：

a）硬件设备存储环境温度：-10℃~+50℃

b) 硬件设各工作环境温度：0℃~35℃(空调开)。

c) 硬件设备工作环境相对湿度：30%~-80%(+25℃）。

设计方案中的所有硬件设备均采用商用成熟的硬件设备，根据硬件设备的型号选择，所有硬件设备均可存储环境温度、工作环境温度、工作环境湿度的要求，同时系统设计方案中配备了30KW的精密列间变频空调，可以确保数据中心机房环境情况如下：

* 温度控制在20℃~+30℃之间
* 相对湿度控制在30%~-50%之间

下表XX给出所有硬件设备的环境指标及指标满足情况。*（请朱亮亮给出）*

表XX 硬件设备的环境适应性指标及其对使用要求的满足情况

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 设备名称 | 设备型号 | 环境指标情况 | 环境指标要求 | 满足情况 |
|  | 服务器 |  | 存储环境温度 |  |  |
|  | 工作环境温度 |  |  |
|  | 工作环境相对湿度 |  |  |
|  | 服务器 |  | 存储环境温度 |  |  |
|  |  |  | 工作环境温度 |  |  |
|  |  |  | 工作环境相对湿度 |  |  |
|  | 服务器 |  | 存储环境温度 |  |  |
|  |  |  | 工作环境温度 |  |  |
|  |  |  | 工作环境相对湿度 |  |  |
|  | 服务器 |  | 存储环境温度 |  |  |
|  |  |  | 工作环境温度 |  |  |
|  |  |  | 工作环境相对湿度 |  |  |

## 可靠性要求与设计

产品、系统在规定的条件下，规定的时间内，完成规定功能的能力称为可靠性。任务可靠性，是指产品在规定的任务剖面内完成规定功能的能力。其中，任务剖面是指产品在完成规定的任务这段时间内所经历的全部事件和环境的时序描述。

在信息系统中，全系统的可靠性由设备、软件及它们之间的信息交互共同完成，其可靠性受设备、软件和通信信道的可靠性影响，且与通信网络拓扑有关。

业主方的对可靠性定义的指标为任务可靠性，其要求如下：

* 任务可靠性要求：单测试任务连续测试工作时间不小于48h。

根据对任务可靠性的要求，单个测试任务连续工作时间不小于48h，为连续2个全天的任务可靠性。

为满足业主单位所提出的任务可靠性要求，所采用的可靠性设计原则主要包括以下几点：

* 服务器集群所选用的13台服务器尽可能选用可靠性高的服务器型号；
* 集群管理软件对服务器计算节点、存储节点的故障状态进行全方位监控，在某服务器发生故障时，可以在服务器集群中及时切换选用无故障的服务器节点继续提供服务；
* 建立高可用的云平台系统，通过LVS负载均衡与双机热备技术实现系统的高可用；
* 采用分区划分模式，确保某一区的宕机不影响另一区的使用；
* 监控和实时采集所有商用软件的Log日志文件，从Log日志文件分析软件的运行状况，对于出现故障的软件及时释放，以减少缺陷的传播和蔓延；
* 对于自研的软件，包括数据库标准规范检查软件、软件资产库管理系统、测试云运行支撑系统、测试数据采集与分析系统，在软件开发时遵循GJB2786A的软件开发过程要求，设计与编码时充分考虑容错性设计，过程按照需求分析、设计、编码、代码规则检查、代码缺陷检测、功能测试、性能测试等过程进行，确保每一个环节都高质量地完成；
* 软件采用冗余技术实现软件的容错性设计，主要方法包括可包括信息冗余、 时间冗余和结构冗余；
* 在软件测试业务中，平台中的LDRA Testbed、Cobot、ParaSoft Test等都能实现代码的分析功能，这为在工具的云化集成时多软件的融合使用提供了冗余的条件；
* 在线监控与运维自动化，通过在线监控软件测试云平台的运行情况，包括CPU的利用率、内存的使用率等，及时对出现宕机的节点进行剔除，对出现故障的节点及时重启、迁移任务。

## 维修性要求与设计

维修性设计是指产品设计时，设计师应从维修的观点出发，保证当产品一旦出故障，能容易地发现故障，易拆、易检修、易安装，即可维修度要高。

业主方的维修性要求如下：

* 硬件设备平均故障修复时间(MTTR)≤30min
* 关键件和重要件可修复、可更换

为满足甲方对维修性的指标要求，我们针对全系统的维修性，做如下考虑：

* 简化产品及维修操作，要对产品功能进行分析权衡，使其构造简单，尽可能减少
* 产品层次和组成单元的数量，合理安排各组成部分的位置。
* 具有良好的可达性，产品的配置应根据其故障率的高低、 维修的难易、 尺寸和质量的大小以及安装特点等统筹安排，凡需要维修的零件部件， 都应具有良好的可达性；对故障率高而又需要经常维修的部位及应急开关，应提供最佳的可达性；
* 提高标准化、互换性程度，优先选用标准化的产品，设计时将故障率高、容易损坏、关键性的零部件或单元具有良好的互换性和通用性；
* 具有完善的防差错措施及识别标志；
* 检测诊断准确、迅速、简便；
* 符合维修的人机环工程要求。

通过上述维修性设计的原则来保证全系统维修性满足维护的需要。

另外，在业主所要求的第一项指标“硬件设备平均故障修复时间(MTTR)≤30min”中，我们对所选用的所有硬件设备，统计分析了其平均故障修复时间，见下表XX所示。

下面的表请朱亮亮给出

表 XX 选型硬件设备的平均故障修复时间

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 硬件设备名称 | 硬件设备型号 | MTTR | 结论 |
|  |  |  |  |  |
|  |  |  |  |  |

从上表可以看出，所选型硬件设备的平均无故障修复时间MTTR最大值为XX，最小值为XX，整体均满足维修性指标的要求。

## 保障性要求与设计

业主方对保障性要求为：

硬件设备采用市电供电，电压：AC220× (1士10%)V、频率:50X(1士5%)Hz。

该项要求需要对所有硬件设备的供电要求进行检查，下表XX给出了我们所选型硬件设备的供电要求。

下面的表请朱亮亮给出

表 XX 选型硬件设备的供电要求

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 硬件设备名称 | 硬件设备型号 | 是否是电可用 | 电压 | 频率 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

从上表可以看出，所选型硬件设备均可以采用市电供电，电压和频率都满足市电的供电要求，同时全系统中采用了UPS电源系统，对数据中心服务器、终端、网络设备进行供电，有效提高了服务器、终端、网络设备供电的有效性。

# 系统的可实现性分析

云计算的提出到现在已经很多年了，作为公有云平台的提供商，目前已有腾讯云、华为云、阿里云等成熟的商业化产品。软件测试云平台将软件测试服务在云环境下进行提供，整个平台采用成熟的五层技术架构，该架构是云平台通用的架构。目前，硬件虚拟化技术、软件虚拟化技术都已有成熟的应用，虽然很多软件测试工具基本还处于单机版的运行状态，但也已经出现了很多开源的软件测试工具，这些开源的软件测试工具都能在云环境下运行，这些成功应用为本项目的实现的技术途径奠定了基础。

## 技术方案的可行性

本设计方案采用典型的云平台技术架构，分为硬件基础设施、基础架构即服务、平台即服务、软件即服务、终端应用层共5层架构，这种架构是典型的云平台技术架构，这种架构已在多种云平台应用中得到验证。

在基础硬件设施层所选用的服务器、网络设备全部采用华为的设备，华为是生产销售[通信设备](https://baike.baidu.com/item/%E9%80%9A%E4%BF%A1%E8%AE%BE%E5%A4%87/10476282)的[民营](https://baike.baidu.com/item/%E6%B0%91%E8%90%A5/6092699)通信科技公司，作为全球领先的信息与通信[技术](https://baike.baidu.com/item/%E6%8A%80%E6%9C%AF/13014499)解决方案供应商，华为的软硬件产品已经得到了广泛的市场检验，可以确保项目的落地，技术措施上是可行的。

云基础软件支撑层主要采用我公司自主研发的x-inCloud产品，x-inCloud基于开源KVM设计，实现了资源虚拟化、虚拟机的管理和配置、虚拟网络管理、运行与安全管理等功能，该云基础软件支撑层已经发展了多个版本，在技术实现上得到了应用的检验。

公司在软件测试云支撑运行环境的开发方面，专门在公司内部建立了软件测试的云平台，该平台已在多个软件的测试中发挥了作用，软件测试云平台解决了测试环境按需定制、测试工具服务提供、测试过程管理、测试结果分析等功能。平台的开发为公司积累了本项目开发的经验，可以确保在技术上实现所需要的指标。

## 风险及风险的解决措施

该项目需要在甲方单位搭建出私有云环境，在该环境中进行软件测试云平台的部署，涉及到硬件的采购、软件测试工具的采购、软件的开发、系统的集成等工作，项目的成功主要存在的风险包括：

（1）硬件供货方面的风险及解决措施

硬件所选用的大部分为华为的硬件，服务器CPU型号为英特尔至强E5-2650处理器，中美贸易战可能为硬件采购供货带来风险。

经咨询华为代理商，供货商可以确保硬件的提供，公司商务层面已经与供应商签订了备货协议，可以降低难以供货的风险。

（2）已有测试工具集成的风险及解决措施

在业主单位已有的测试工具中，ETest为凯云联创（北京）科技有限公司的产品，该产品的技术资料与流行的测试工具相比，技术资料公开的程度较少，可能会导致集成困难。

旋极公司已经与凯云联创（北京）科技有限公司就ETest的云化集成问题进行了技术沟通，ETest原厂对开展云化的改造工作比较积极，可以确保解决该方面的风险。

（3）系统集成验证的风险及解决措施

软件测试云平台在军内单位的应用案例尚不多，系统集成后是否能达到相应的预期效果存在一定的实施风险。

在项目实施过程中，将严格按照系统工程、软件工程的要求开展硬件的采购、系统的设计、软件的开发、内部测试、外部测试、系统验证等工作，确保建成的系统稳定可靠。同时，通过在典型的分布式信息系统软件装备试验和电子信息装备软件测评过程中，应用软件云测试平台开展应用验证工作。

## 已有的研究成果及支撑材料

旋极信息在软件测试管理系统、软件测试工具、云基础平台等方面做了大量的研究工作，主要研究成果包括：

1. 软件测试管理系统

请把管理系统介绍介绍，把测试管理系统的界面、功能组成框图、使用手册（扫描件）等作为证明材料附上。

1. 测试技术服务及流程管理平台

该平台为旋极信息与中科院签署的基于基础云环境所开发的项目，项目旨在解决测试技术服务及流程管理平台将测试过程中原来依靠人工流转的测试技术过程、管理过程，利用电子化的方式进行统一管理，实现测试流程自动流转，并自动化调用其它服务完成一系列测试工作，平台设计遵循测试系统的智能、标准的云架构。如下图所示。



//请把支撑材料提供，主要包括合同首页、软件界面、手册扫描图片等

。。。。。

其他材料旋极多提供一些吧，包括为别人做的测试软件都可以！！

# 系统建设远景建议

刘敬民搞（2页以上）

# 技术指标的满足情况分析

李佳洁搞（30页以上）