目录

**[1 设计原则 2](#_Toc1448059788)**

[（1）科学性原则 2](#_Toc108685465)

[（2）兼容性原则 2](#_Toc1315510305)

[（3）系统性原则 2](#_Toc1437550270)

[（4）前瞻性原则 2](#_Toc1716359140)

[（5）适用性原则 2](#_Toc1847719476)

[（6）需求导向原则 3](#_Toc2007697512)

[（7）循序渐进原则 3](#_Toc2109022520)

[（8）可扩展性原则 3](#_Toc2123899905)

[（9）科学性原则 3](#_Toc912522901)

[（10）兼容性原则 3](#_Toc1591673880)

[（11）系统性原则 3](#_Toc59110481)

[（12）前瞻性原则 4](#_Toc1332409253)

[（13）适用性原则 4](#_Toc1990327902)

[（14）可靠性设计 4](#_Toc88279595)

[（15）可维护性设计 5](#_Toc1951436735)

[（16）需求导向原则 5](#_Toc1426948161)

[（17）循序渐进原则 5](#_Toc1767855878)

[（18）可扩展性原则 5](#_Toc1917485301)

**[2 国内外大数据研究现状分析 6](#_Toc2035847025)**

**[3 数据交换技术 9](#_Toc624001524)**

**[4 数据管理技术 14](#_Toc1430965567)**

**[5 数据分析技术 14](#_Toc568921816)**

**[6 微服务架构设计 15](#_Toc1271765068)**

[（1）服务发现与注册 15](#_Toc650759285)

[（2）负载均衡 16](#_Toc177088824)

[（3）服务容错 16](#_Toc2067013873)

[（4）服务网关 16](#_Toc459205992)

[（5）配置中心 17](#_Toc1966363873)

[（6）服务架构 18](#_Toc1051769828)

**[7 基于Java EE的B/S开发体系 20](#_Toc1157600739)**

**[8 数据转换方法 21](#_Toc1741262200)**

**[9 基于模型的推理 27](#_Toc1634137731)**

**[10 基于不确定的推理实现 28](#_Toc784483434)**

**[11 国内外数据交换现状分析 29](#_Toc1410966305)**

[（1）使用数据库自带工具进行转换 30](#_Toc1596257961)

[（2）利用中间数据库转换 30](#_Toc1941832203)

[（3）设置传送变量转换 31](#_Toc1064852362)

[（4）通过开发数据库组件或数据库中间件技术转换 31](#_Toc1992417683)

[（5）基于XML的数据转换 32](#_Toc851490510)

**[12 插件技术 33](#_Toc169977962)**

**[13 专家系统 34](#_Toc666356824)**

**[14 用户体验及交互设计技术 35](#_Toc331921863)**

**[15 软件开发规范流程 36](#_Toc1595720182)**

**[16 快速匹配技术 36](#_Toc1493315138)**

# 设计原则

（1）科学性原则

大数据体系在功能确定和结构设计过程中都应具备科学性原则，确保作物种质资源大数据体系的科学合理。在构建作物种质资源大数据体系过程中，应用Delphi 法（DOKE et al., 1995）对作物种质资源领域专家进行多次专业调查，确保体系的科学性。

（2）兼容性原则

构建作物种质资源大数据体系应充分考虑与现有资源、现有管理信息系统相兼容。充分考虑兼容性问题有助于降低新老系统间的冲突，提高老系统的利用率，降低建设成本，以达到系统间平稳过度的目的。

（3）系统性原则

在作物种质资源研究过程中，必然与生态和地理环境等外围要素互相作用、紧密关联，因此要构建作物种质资源大数据体系必须考虑与之相关的各类要素，按照它们之间的关联关系而组成一个系统性的有机体系。

（4）前瞻性原则

在科技飞速发展的当下，新技术应用极大促进了作物种质资源学科的进步。为了适应技术变革，构建作物种质资源大数据体系应把握未来作物种质资源科学研究和大数据技术的发展方向，应具有一定的前瞻性。

（5）适用性原则

大数据建设涉及到方方面面，其难度非常大。因此，作物种质资源大数据体系构建不应面面俱到，应将重点放在建设作物种质资源数据获取、分析和应用上，而基础设施平台的建设可借鉴当前成熟的大数据平台即可。

（6）需求导向原则

作物种质资源大数据体系构建的关键是明确作物种质资源研究过程中的需求问题，再将不同的大数据要素按一定的规则组合起来去满足、匹配作物种质资源研究需求。

（7）循序渐进原则

以现有数据和资源为基础，制定近期目标和远期目标。构建作物种质资源大数据体系应先梳理现有的数据，在此基础上先开发研究人员常用功能，如数据查询、关联分析和主成份分析等。再发具有智能算法的高级功能。

（8）可扩展性原则

由于作物种质资源研究随着业务发展而不断变化，针对研究人员提出的新需求，大数据开发人员可以独立模块的方式进行业务功能开发，做到灵活部署，扩展现有功能，提高大数据系统生命力。

（9）科学性原则

大数据体系在功能确定和结构设计过程中都应具备科学性原则，确保作物种质资源大数据体系的科学合理。在构建作物种质资源大数据体系过程中，应用Delphi 法（DOKE et al., 1995）对作物种质资源领域专家进行多次专业调查，确保体系的科学性。

（10）兼容性原则

构建作物种质资源大数据体系应充分考虑与现有资源、现有管理信息系统相兼容。充分考虑兼容性问题有助于降低新老系统间的冲突，提高老系统的利用率，降低建设成本，以达到系统间平稳过度的目的。

（11）系统性原则

在作物种质资源研究过程中，必然与生态和地理环境等外围要素互相作用、紧密关联，因此要构建作物种质资源大数据体系必须考虑与之相关的各类要素，按照它们之间的关联关系而组成一个系统性的有机体系。

（12）前瞻性原则

在科技飞速发展的当下，新技术应用极大促进了作物种质资源学科的进步。为了适应技术变革，构建作物种质资源大数据体系应把握未来作物种质资源科学研究和大数据技术的发展方向，应具有一定的前瞻性。

（13）适用性原则

大数据建设涉及到方方面面，其难度非常大。因此，作物种质资源大数据体系构建不应面面俱到，应将重点放在建设作物种质资源数据获取、分析和应用上，而基础设施平台的建设可借鉴当前成熟的大数据平台即可。

（14）可靠性设计

系统采用先进成熟的技术，以保证投资的有效性和延续性。支持常用的操作系统系统、常用的数据库系统、常用的应用服务器系统和常用的开发工具，保证系统能够正常可靠运作。

系统依托但不依赖现行政策，流程设计动态可调，伴随政策调整可以通过业务流程变更满足业务需求变化，用户界面可动态调整但不影响系统功能，确保系统运用适应性强，长期可靠。

同时，依据GJB/2102-1997《软件可靠性与安全性设计准则》、GJB1267-1991《军用软件维护》的有关要求，系统设计了容错能力、故障恢复能力、容灾备份能力等，对于各类非正常数据的录入、数据导入等操作，能够进行正确判断并提供合理的引导与处理，对用户的非正常操作，用户的误操作不能对系统运行造成影响，保证系统局部故障不会导致系统崩溃。

软件系统能够7×24小时连续不断稳定工作。

平均故障间隔时间（MTBF）≥1500h。

平均故障修复时间（MTTR）≤37min。

遵循注释代码行≥23.5%。

可在不影响使用的情况下进行软件版本升级或改进，保证系统可以稳定、平滑过渡。

（15）可维护性设计

根据系统不同层次、不同业务的用户需求，系统通过统一的界面管理进行界面集成，通过统一的用户服务进行权限管理，可为用户灵活配置功能权限，满足数据应用的需要。系统开发前后台开发语言一致，都使用JAVA语言进行开发，生成标准的系统安装程序，提升系统的可维护性。对于数据交换传输、节点接入、共享数据变更等核心业务都提供了监控管理功能，可有效辅助系统管理人员实施维护。对手工录入操作实施校验规则检查，增加边界值和无效值的判断，防止错误输入引起系统异常。对于系统各功能模块的配置、控制、监视、诊断等工作能够通过系统的管理工具方便的进行操作，便于各层次各业务用户直接维护。

（16）需求导向原则

作物种质资源大数据体系构建的关键是明确作物种质资源研究过程中的需求问题，再将不同的大数据要素按一定的规则组合起来去满足、匹配作物种质资源研究需求。

（17）循序渐进原则

以现有数据和资源为基础，制定近期目标和远期目标。构建作物种质资源大数据体系应先梳理现有的数据，在此基础上先开发研究人员常用功能，如数据查询、关联分析和主成份分析等。再发具有智能算法的高级功能。

（18）可扩展性原则

由于作物种质资源研究随着业务发展而不断变化，针对研究人员提出的新需求，大数据开发人员可以独立模块的方式进行业务功能开发，做到灵活部署，扩展现有功能，提高大数据系统生命力。

# 国内外大数据研究现状分析

国内外大数据研究现状全球已进入大数据时代，大数据为各行业带来巨大的商业利益已引起各国政府的重视，世界主要国家相继制定了相关政策来促进大数据的发展与利用（电子标准院，2018）。美国政府（2012）发布了“大数据研究与发展倡议书”，其目的是提高从海量数据中提取知识的能力，从而促进科学研究，改善美国国家安全环境并实现教育变革，推进相关研究机构进行科学创新。2016年5月，奥巴马政府再次提出了“联邦大数据研究与发展战略计划”，该计划重点说明了新兴大数据能力，并为制订联邦大数数据的研究与发展提供指导，它是2012年3月的―大数据研究与发展倡议书‖的一个重要里程碑，旨在利用大数据丰富的资源优势，建立一个大数据创新生态系统，该系统基于海量、多样和实时数据，进行分析、提取信息以及做出决策和发现潜在知识为国家带来新的能力，同时加快科学发现和创新进程，形成新的研究领域，促进新的经济增长。欧盟软件与服务技术平台和欧洲Big项目发起了一项关于在欧盟内实现大数据价值的战略研究与创新议程（姜禾，2014）。该议程就关于实现大数据价值需解决的挑战与需求问题向公众征询意见。议程建议建立欧盟合同制公私伙伴计划，增强泛欧洲范围内的创新能力。议程确定了管理数据的隐私与匿名机制、提高数据分析技术、优化数据处理框架、改进用户体验和提升数据管理等五大研究与创新领域。议程同时提出了加强大数据人才培养和标准化等问题。2020年2月19日，欧盟发布了《欧洲数据战略》的战略规划报告，涵盖了数据、技术和平台治理等领域的发展规划和立法框架。欧盟致力于建构“单一数据市场”，并将其打造成全球数据赋能社会的典范和领导者。其它发达国家也制定了大数据发展政策（胡瑛，2016），英国于2013年初投入近2亿英镑用于大数据建设以便在大数据浪潮中抢占先机。2013年6月澳大利亚政府提出大数据建设方案，希望通过大数据分析改善公共服务质量。日本及韩国都制定了大数据发展战略，从立法、技术和管理等方面确立大数据发展的基本原则和政策。通过分析可见，世界主要发达国家都认识到了大数据在社会生产中的重要性，并结合政府和市场力量共同推进大数据发展。

发达国家除了重视大数据政策引导之外也非常注重对大数据架构的研究。2013年1月，美国国家标准与技术研究院（National Institute of Standards and Technology，NIST）决定成立工作小组研究关于大数据的互操作框架。该框架能够从互操作性、可移植性、可重用性及安全性等多方面满足用户对大数据的业务需求。NIST大数据公共工作小组将工作分为三个阶段，分别对应三个版本。2015年9月，NIST发布了《NIST大数据互操作框架：1~7卷》的第一个版本。该版本包括《大数据定义》、《大数据分类》、《大数据案例与总体需求》、《大数据安全与隐私》、《大数据架构调查白皮书》、《大数据参考架构》、《大数据标准路线图》等七卷。2018年6月，NIST再次发布了《NIST大数据互操作框架：1~9》的第二个版本，在第一版的基础上增加了《大数据参考架构接口》和《大数据采用及现代化》两部分。当前该框架的第三阶段工作已经开展。2014年11月，国际标准组织的下属机构电工委员会（ISO/IECJTC1）也组建了大数据工作组用于制订大数据各项标准。目前该组织已编制了有关大数据的基本定义和架构等两项标准。大数据基本定义（ISO/IEC20546，2019）部分提供了关于大数据必要的术语、概念及词汇等内容，明确了与其他技术领域和标准工作的关系，促进大数据领域的交流和理解。大数据参考架构（ISO/IEC20547-3,2020）包括两部分内容。第一部分说明了大数据生态系统中各个角色的概念、角色之间的关系和活动类型的用户视图；第二部分定义了大数据体系结构层，解释了这些层中功能组件的功能视图和这些功能组件在用户视图中实现角色/子角色的活动。除此之外，国际电信联盟电信标准分局ITU-T也制定了与大数据基础设施有关的云计算需求、大数据元数据规范和数据溯源等大数据相关标准。

国内研究机构对大数据标准体系开展了充分研究。2015年，全国信息技术标准化技术委员会（简称信标委）成立了大数据标准工作组负责制订我国大数据领域的标准体系（佚名，2015）。目前信标委制定的多项标准已经对外发布实施，部分标准已进入申批过程。2018年3月29日，由中国电子技术标准化研究院主编的《大数据标准化白皮书（2018版）》对外发布，该白皮书在2016版的基础上新增了大数据政策、最新趋势和应用领域等内容，优化了大数据标准体系。2018年4月，有关大数据安全标准的《大数据安全标准化白皮书（2018版）》对外发布。该白皮书从安全的角度讨论了大数据面临的挑战，介绍了大数据安全法规，分析了安全技术在大数据中的应用等。最后从法规建设、技术研发和人才培养等角度给出大数据安全建议。中国信通院从2014年到2019年已经发布了四个版本的《大数据白皮书》。白皮书梳理了世界其他国家以及我国大数据在各行业的研究进展和趋势，并对存在的问题进行分析和讨论。大数据白皮书还分别探讨了技术、产业、数据资产管理和安全等四个方面的问题并给出意见和建议供政府及各行业进行参考。2019年3月，由国家多个部门联合主编和企事业单位成员参编的《工业大数据白皮书（2019版）》发布。该白皮书是在2017版的基础上提出工业大数据管理的相关方法，进一步优化了工业大数据标准体系，给出工业大数据标准化的指导意见。白皮书以如何应用好大数据技术来管理工业生产中的数据这一问题展开论述，提出了有关工业数据的管理方法，通过建立工业大数据标准体系解决大数据落地应用等问题。2018年3月，国务院办公厅发布了《科学数据管理办法》，该办法要求各科研单位按照相关标准规范开展研究数据的采集和管理工作。以此办法为基础，中国科学院构建了关于科学数据的规范体系（胡户霖等，2019），该科学数据规范体系包括数据、元数据、数据管理、数据质量和服务等标准规范，可为其他行业的科学数据管理提供参考。

除了政府部门对我国大数据标准化问题进行研究外，国内很多学者也发表了关于大数据标准化方面的学术论文。肖筱华等人（2014）分析了我国大数据标准化现状，通过分析大数据技术的特征及大数据技术体系现状，提出了技术标准体系构成，为大数据标准体系构建提供参考。张群等人（2017）分别介绍了国内外大数据领域的标准化研究现状，着重分析了信标委有关大数据的标准化工作，并对国内大数据标准体系研制情况进行总结。林旺群等人（2017）分析了安全要求较高的特定领域大数据标准化现状，研究安全级别要求较高行业大数据的特点和需求，提出了相应的大数据参考架构和标准体系，为该领域的大数据建设提供参考。此外，铁路（马小宁等，2016；吴艳华等，2019）、电力（段军红等，2015；郭乃网等，2017）和地质（谭永杰，2016；李敏等，2018）等行业也以自身特点开展了大数据标准体系研究。

# 数据交换技术

数据交换技术是实现数据交换的方法和规范的技术综合。由第一章绪论所述，数据交换技术很多，涉及到的范围也很广。数据交换技术有数据库管理系统提供的数据导入导出工具（SQL Server的DTX，ORACLE的Dblink等），有各种数据程序设计语一言如JAVA，VC，C#提供的数据库操作语句，有需要借助于某些描述语言XML。这些技术手段都在不同的软件环境数据库管理系统或软件编译环境发挥各自的功能。

（1）XML技术

可扩展标记语言XML是一种元置标语一言，它允许程序开发人员根据它所提供的规则，制定各种各样适合实际问题需要、可以自定义的置标语言XML。具有开发性、数据分离口和把文档的三要素独立开来处理等优势。

XML的数据内容与显示形式是完全分离的，非常适用于异构数据之间的数据交换。在数据交换方面具有独特的优势：第一， XML文件为纯文本文件，可用任何文本编辑器读取，不受操作系统、软件平台的限制第二，易表义，XML是一种完全面向数据语义的标志语言，具有基于Schema自描述语义的功能，容易描述数据的语义及元素结构，这种描述能为计算机理解和自动处理；第三，XML不仅可以描述结构化数据，更可以有效描述半结构化，甚至非结构化数据。以上优点使得数据交换更为廉价和方便，以作为数据交换的载体，已成为数据管理模式的发展趋势。相关技术为数据交换平台的信息安全提供了支持， WS-Security可以对所传输的数据进行加密、解密和数据完整性等操作，充分保证了数据的安全。

与XML相关的技术有：模式(Schema)与文档类型定义(DTD）。它们是对一类XML文档进行约束、确定其结构，元素、属性、数据类型，及XML文档中所使用的元素、实体、元素的属性，元素与实体之间的关系的约束。

（2）JDBC技术

JDBC是JAVA语言为了支持SQL功能而提供的与数据库相连的用户接口。JDBC是由JAVA实现的数据库访问中间件。它由一组用JAVA语言编写的类和接口组成。用户可以通过JDBC向各种关系型数据库发送SQL语句，只要使用JDBC提供的类对象或是接口即可，而不必为不同的数据库操作编写不同的程序。有了JDBC以后，对于数据库编程，用户只需要在JAVA中使用SQL语句，而不用考虑DBMS(数据库管理系统)的连接和访问过程。

简单地说， JDBC可做三件事与数据库：建立连接、发送操作数据库的语句并处理结果。JDBC API既支持数据库访问的两层模型(C/S)，同时也支持三层模型(B/S)。

（3）JMS技术

JMS是SUN及其伙伴公司提出的旨在统一各种消息中间件系统接口的规范。它定义了一套通用的接口和相关语义，提供了诸如持久、验证和事务的消息服务，它最主要的目的是允许JAVA应用程序访问现有的消息中间件。JMS规范没有指定在消息节点间所使用的通讯底层协议，来保证应用开发人员不用与其细节打交道，一个特定的JMS实现可能提供基于TCP/IP、HTTP、UDP或者其它的协议。现在JMS己成为J2EE平台的一个重要组成部分，应用程序开发人员可以将消息传递用于使用J2EE API的组件。

（4）基于JAVAEE的数据交换平台技术结构

现在已经市场化的数据交换平台，为了保证数据交换平台的通用性、灵活性、稳定性和可靠性，多采用基于JavaEE的技术框架来实现的。在软件体系结构上，采用比较流行的BS结构，即浏览器和服务器结构。

从软件技术上，如下图所示，将数据交换平台分为三个层次数据表示层、数据处理层和数据存储层。



图数据交换平台三层结构

在数据表示层，JSP中的代码被编译为Servlet并在java虚拟机中执行。通过JSP，配合JavaScript作为表示层程序设计语宕一。web容器选择响应速度快，使用开源的web容器。

在数据处理层， JavaBean运行在java虚拟机中，可以跨操作系统进行编译和执行，用JavaBean实现数据交换的逻辑处理。用XML文件或是文本文件描述信息。

在数据存储层，包括了数据交换相关的数据库和文件。数据库存储数据交换平台的系统管理数据。文件存储系统的配置数据。数据库管理系统路使用轻量级的数据库，如MySQL。因为MySQL具有轻量级、支持XML、执行速度快、代码开源等优点。

从应用上，数据交换平台划分为以下模块：

1.数据源管理

平台采用数据适配器来接收不同数据来源的数据。数据适配器通过统一的数据源描述屏蔽了各不同类型数据源的差别，通过定义统一的连接接口为上层交换提供统一的底层接口，也使新类型的数据源接入仅需要增加连接器即可实现。

2．设置管理

设置管理是管理数据源定义、交换数据之间映射关系和交换执行方式、时间定义的管理模块。为数据交换的执行提供详细的描述信息。

3．数据交换管理

数据交换管理是数据交换任务的执行模块，是平台根据定义的数据交换任务最终完成数据读取、校验、格式转换、编码、传输、解码、格式还原、数据写入等操作，实现数据交换。

4．权限控制

权限控制是实现对数据交换各种资源的访问及使用权限控制。管理员可以通过平台的管理控制台为数据源订阅发布源、任务设置访问权限。

5．传输控制

传输控制采用数字证书对压缩的数据作数据签名保证数据的完整性，安全性。采用数字证书对压缩的数据作数据签名，保证数据的完整性、安全性。

表我国安全可控桌面计算机技术体系主要构成

|  |  |
| --- | --- |
| 技术 | 代表企业 |
| 操作系统 | 中标麒麟、普华、天津麒麟、深度、思普、红旗、计德 |
| CPU | 龙芯、飞腾、申威、众志 |
| BIOS | 百敖、太极 |
| 可信3.0 | 可信联盟旗下已有上百家企业 |
| 虚拟化 | 华三、华为、云宏、深信服、同方、浪潮、有孚 |
| 数据库 | 南通、达梦、金仓、神通 |
| 中间件 | 东方通、金蝶、中创、普元、中和威 |
| 网络安全 | 华为、启明星辰、深信服、绿盟、360、亚信、卫士通、卫达 |
| 办公套件、流式文档 | 金山、福昕 |
| 企业管理软件 | 用友、金蝶、数码大方、宝信 |
| 浏览器 | 360、搜狗、QQ、百度 |
| 输入法 | 搜狗、QQ、百度、讯飞 |
| 排版、印章、版式文档 | 方正、文泰、蒙泰、福昕、书生 |
| 系统集成 | 浪潮、东软、中软、太极、同方、神码、东华、航信 |
| 第三方机构 | 软交所、有关评测中心、有关标准化机构 |

# 数据管理技术

为开展数据管理的目的，同时为了数据管理等用途的需要，应对数据管理全过程数据（包括基础数据、操作数据、事件数据、结果数据等）进行一体化管理，并且建立数据关联关系，提供不同条件及其组合的数据检索功能。

技术的解决途径：一是针对数据收集功能，能够在后台高速、自动地存储动态数据，采用多层缓冲的数据文件记录方法，用独立线程进行外部IO的方式进行写盘记录；二是采用面向数据库存储机制，支持海量数据的高效结构化存储访问；三是综合其他数据源及其采集存储方式，按照方案的方式进行统一标识存储。本系统对这三类数据采用虚拟表空间的技术方法进行数据的关联管理，从而能够进行一体化的存取手段。

# 数据分析技术

数据分析是系统建设的难点之一。数据涉及操作数据、事件数据、结果数据等需求、分析和应用问题。因此，必须注重各个环节对数据准确性、数据样式、数据内在关系的研究，并时刻保持在一定的数据运用背景下进行数据的抽取和分析应用。

技术的解决途径：系统采用对象过程方法论和元模型技术，构建指标体系概念建模工具，支持指标的定义与管理等功能，解决指标组合问题。首先收集和梳理开展化学侦察领域知识，基于对象过程方法建立形式化的应急概念元模型。其次，针对具体的处理事件，基于应急概念元模型建立任务空间概念模型，形成解决分析问题的行动方案空间，实现对数据的形式化表达。再次，建立关键分析要素，根据行动方案空间构建指标体系，实现指标的组合。

# 微服务架构设计

识别出业务服务之后，需要设计服务架构，考虑到微服务系统本质上也是分布式系统，所以需要设计相应的组件或服务，用来搭建微服务分布式环境。

（1）服务发现与注册

在微服务架构中，服务发现组件（也可称作服务注册中心）十分重要，因为微服务系统往往由多个微服务组成，同时还是分布式的，一个服务甚至会启动多个实例，所以需要服务发现组件来实现服务的发现与注册。本文设计使用Eureka框架实现服务发现组件，如下图所示，每个微服务在启动时，都会在服务发现组件中注册自己的网络地址信息，服务消费者就可以通过服务发现组件查询服务提供者的网络地址，通过地址调用服务提供者的接口。微服务与服务发现组件使用一定机制通信，当某个微服务长时间失联之后，服务发现组件就会注销该服务实例。并且当微服务网络地址发生变化时，会重新注册到服务发现组件，服务消费者无需人工修改服务提供者的网络地址。



图 服务发现架构

（2）负载均衡

微服务在生产环境中，各个微服务通常会启动多个实例来提高系统性能，如何把服务消费者的请求合理的分摊到多个服务提供者实例上，充分利用后端服务资源，也是微服务系统的关键点。本文设计使用SpringCloud中的Ribbon负载均衡器，为Ribbon配置服务提供者地址列表后，Ribbon就可以基于相应的负载均衡算法（如轮询、随机等）来自动的为服务消费者分配服务提供者实例。

（3）服务容错

在实际生产环境中，一个微服务系统往往需要实现很多业务功能，包含多个服务层，完成一个业务请求不可避免的会遇到服务之间的级联调用。如下图所示，服务消费者调用服务A，服务A依赖服务B，服务B依赖服务C，只有依赖服务依次正常返回响应结果，整个调用链才能响应成功。一旦某一服务产生故障或响应超时，就有可能扩散到整个调用链，影响到系统正常运行，这被称作雪崩效应。所以为了防止雪崩效应，需要为微服务系统设计一个强大的容错机制。本文设计采用SpringCloudhystrix断路器，通过为网络请求设置超时来及时释放超时资源，以及设定服务调用失败阈值从而停止请求持续访问失败的微服务。以此来防止级联失败，从而提高微服务系统的容错性和可靠性。



图 服务层级调用

（4）服务网关

外部客户端完成一个业务需要调用若干个接口，如果客户端直接与多个接口通信，会增加系统复杂性，不利于解耦。使用微服务网关来作为中间层，所有的外部请求都需要先经过微服务网关，然后再到后端微服务，可以减少客户端和各微服务之间的交互次数，也便于在网关中实现身份认证、动态路由、负载均衡等功能。

通过前文对系统功能模块的划分，已经得到了相对稳定且可以独立开发实现的服务划分，具体包括用户服务、通知服务、申请评审服务、会议服务和工作流服务五个业务功能服务。各微服务和网关之间的架构关系如下图所示。前端访问后端微服务，先统一通过网关然后才能转发到对应微服务。



图 评审系统服务网关架构

（5）配置中心

考虑到微服务分布式的特点，一个微服务系统往往会有多个服务，甚至同一服务启动多个实例，因而需要一种集中管理微服务配置的功能。如下图所示，SpringCloudConfig就为微服务的外部化配置管理提供了服务端和客户端实现。通过设计一个独立运行的配置管理服务，将微服务的配置文件统一存放在配置服务本地，所有微服务在本地配置文件中指向配置中心服务，在服务启动时，ConfigClient就会请求CongfigServer获取配置文件属性并进行缓存。通过配置中心的集中式配置管理，方便实现服务的多环境配置，服务运行时动态调整，以及配置更新后无需重启服务，服务能够自动更新配置。



图 配置中心架构

（6）服务架构

如下图所示，vue前端项目负责和用户交互，核心业务微服务负责实现业务功能，SpringCloud提供基础组件服务，支撑起整个微服务系统。



图 系统服务层次结构

在生产环境中，各组件服务之间的详细关系如下图所示。用户经过nginx代理访问系统前端，前端在调用后端服务完成用户请求。对后端服务的请求，需要先统一经过Zuul网关，然后才能访问评审系统各业务功能微服务，从而获取完成业务功能所需的数据，并将数据返回至服务网关，由服务网关进行整合最终返回至用户。如果服务存在多个实例，在网关中进行负载均衡，分摊请求到服务实例。五个业务微服务，包括Zuul网关服务和配置中心服务ConfigServe都需要在服务注册中心Eureka上进行注册，以便实现服务地址的注册和发现。配置中心ConfigServer负责统一管理各服务的配置文件，实现配置更新服务自动获取，避免重新启动服务。Ribbon负责实现服务负载均衡，Hystrix实现服务熔断。



图 系统总体架构

# 基于Java EE的B/S开发体系

相比传统的Web应用程序的三层模型，Java EE采用展示层、控制层、服务层、数据持久化四层，下一层为上层提供服务，同时屏蔽掉底层实现的差异性，提高系统的灵活性。

采用多层结构的系统，应用程序各部分之间实现了松耦合，每一层可以被单独改变，而无需其他层的改变，降低了部署与维护的开销，提高了灵活性、可伸缩性。业务逻辑集中放在服务器上由所有用户共享，使得系统的维护和更新变得简单，也更安全。数据库不再和每一个活动的用户保持一个连接，而是由应用程序组件负责与数据库打交道，降低数据库服务器的负担，提高了性能。目前，基于Java EE的B/S开发体系已在电子商务、电子政务和系统集成领域得到广泛应用。

# 数据转换方法

数据在从一个数据源移动到另一个数据源中，或者数据源类型改变了，数据通常会要求随着条件的改变而改变，以满足新的要求。典型的数据转换类型主要有以下几种：

1、重构，在载入不同的数据之前，将其重构为其他的形式以便解决不同的问题；

2、替换，替换经常是用于改变数据内容；

3、数据类型的转换。

目前，经常采用的异构数据转换方法可以分为两类：基于模板的转换和基于模型的转换。前者，基于模板的转换只能用于在关系数据库和文档之间传递数据，它不需要预先定义XML文档模式和数据库模式间的映射关系，而是直接将命令语句(如SQL语句)嵌入XML模板中，由专门的数据传输中间件来执行该模板，获得的执行结果用来替换模板中对应的命令语句部分，从而得到结果XML文档；后者，基于模型的转换则使用某种数据结构在XML文档模式和数据库模式间建立起对应关系，然后进行数据的转换工作。基于模型驱动实现数据在数据库和XML文档间的双向传输的关键是在数据库模式和XML Schema或DTD之建立双向映射。

（1）数据交换模型

异构数据库之间要进行数据交换，其核心就是解决各种异构所带来的问题。在上一章，对数据交换过程中单表结构上的数据类型、语义表示、数据长度、度量、精度和多表联系上的异构情况进行了分析。

数据模型是数据特征的抽象，是数据库管理的形式框架。数据模型包括数据库数据的结构部分、数据库数据的操作部分和数据库数据的约束条件。所有的数据库管理系统都是基于这样的概念和实现模式。借鉴这样的思想和实现模式，重点描述一种基于的数据交换模型以下简称该模型。该模型是在总结前人的模型发展而来的。该模型能够代表了所有基于XML的数据交换模型的共同特点。

该模型充分地吸取了XML的优点，用XML文件作为中间数据，利用XML丰富的表示形式和通用性自定义了数据的数据文档和结构文档，将数据表示和结构表示分开。

该模型如图所示。



图 数据交换模型

在该模型中，数据交换的过程是首先通过对源数据表和目标数据表进行比对和分析，分别得到其用中间数据表示的结构文档以下和数据文档；然后对源数据和目标数据的结构文档进行分析，按照转换的需求，得到数据转换配置文档在源数据和目标数据的结构文档和转换配置文档的基础上生成映射文档，最后结合源数据文档和映射文档完成数据的交换。

在图中，数据提取模块、中间数据模块、数据交换模块是该数据交换模型的三个主要部分，下面分析各个模块的功能。数据提取模块是整个交换模型的基础，主要是和用户直接交互，获取源数据和目标数据的匹配信息中间数据模块是整个交换模型的核心，主要是将源数据和目标数据表示成相应的数据文档和结构文档，按照用户需求，进行结构匹配操作，消除类型异构、语义异构、长度异构、精度异构、度量异构和联系异构等异构数据交换模块主要是将源数据文档按照数据映射文档和相关算法，将源数据写入目标数据库中。

（2）数据提取

数据提取模块是整个数据交换模型的基础，它的主要任务是与用户交互，获取数据源连接的相关信息，获取源数据表和目标数据表之间的数据结构对应关系，产生数据结构文档，为数据交换做好准备。

数据提取模块在整个数据交换模型中负责与用户进行交互工作，因为数据库连接方式的选择与系统的可扩展性紧密相关。因此，数据提取模块的设计模式选择是至关重要，要尽量减少数据之间的祸合，尽量考虑到系统的扩展性。考虑到整个模型今后在实际应用中的发展空间，为实际应用打下好的基础。同时，对数据提取时，需要对提取的数据都用中间数据表示，因此对XML文档进行处理时应根据文档的特点及编程要求来选择相应的编程模型。

数据提取部分包括模式提取模块、模式提交模块、数据卸载模块。模式提取模块的任务是从关系数据库抽取关系数据表的结构信息作为结构文档，抽取关系数据表的值信息作为数据文档，实现关系模式与XM模式的转换。模式提交模块的任务是在模式提取模块将关系模式转换为XML模式后，分别将结构文档和数据文档提交保存为文件，进行不需要的索引处理，以供数据交换时使用。数据卸载模块的任务是负责提取从数据交换模块返回的映射后的数据文档，并传送到目标数据库管理系统，加入具体的数据库中。

（3）中间数据

中间数据模块的作用是连接源数据和目标数据，利用XML作为存储和交换的中间媒介格式，生成源数据表和目标数据表的结构文档和数据文档，分别表示需要转换的数据和结构，完成从关系模式到XML模式的转换。为了方便下一步的数据交换，有时候，也可以将目标数据表认为是空。该模块通过对结构文档的匹配操作，来消除源数据和目标数据的类型异构、语义表示异构、数据长度异构、精度异构、度量异构和联系异构，是整个数据交换模型的核心。

关系数据库中的数据都是规则的二维结构关系表，所以能使用很简单的XML文档来表示。对象数据库中的数据，由于XML文档本身就是树形组织模式，所以也能使用XML文档来表示，只是比较复杂而已。利用XML作为交换的中间媒介格式，源数据和目标数据通过中间格式来进行表示和交换。一方面，利用XML来表示需要转换的数据本身另一方面，也利用XML来表示数据结构。

（4）数据文档

经过初步转换的源结构文档和目标结构文档，按照匹配算法进行结构匹配，输出数据映射文档。

（5）结构文档

关系表的数据可用一个数据文档表示，目前对关系表的结构和语义进行描述可采用DTD或XML Schema，即用DTD或XML Schema对DataXmlFile进行描述。但在实际应用中发现， DTD或XML Schema对复杂语义的描述能力有时无法满足需要。关系模式中采用了结构规范的数据表，我们可以用XML文档来表示关系表的结构。为描述关系表中的复杂语义，定义了一个特殊的XML文档来表示关系表的结构和语义信息，将该XML文档称为StructureXML文档。

StructureXMLFile的结构主要是根据实际应用中遇到的情况确定的。在StructureXMLFile中，关系表的名字用table元素中id记属性指明，dataxmlfilenname 指明其对应的DataXmlFile文档名。Field元素定义表中的每一个字段，分别用标准属性名id、type、length、precsion、unit、style来表示属性名、数据类型、数据长度、数据精度、单位、数据格式的语义类型。其中数据格式主要是解决同一种数据类型中包含几种格式，如日期型数据，有dd/mm/yy、mm/dd/yy、mm/dd/yyyy、yyyy-mm-dd等几种。other代表其他特殊情况，其属性名需要根据具体情况来命名，数量也需根据实际情况多少不等。也可在此属性中指定对字符属性拆分时的标准，如分隔字符，拆分位置等。field中的isnull属性表示该元素是否允许为空。若对字段不存在的语义，可省略相关语义标记。需要指出每一条记录不一定包括所有的属性结点。

（6）转换配置文档

用户通过比较源数据结构文档与目标数据结构文档之间的差异，定义好源数据与目标数据的数据转换配置文档。数据转换配置文档包含了源数据与目的数据交换时各字段之间的对应关系，解决各种异构的解决办法，也许包括系统预置的方法和用户自定义的方法。因为它的定义是面向用户的，所以应当尽量简单明白，方便用户了解、学习、掌握与使用。这里称该XML文档为结构差异XML文档， 记为DSXML文档。

DSXML文档分为两大部分：<field>和<table>。<field>部分定义整个数据交换过程中对所有表有效的属性名之间的对应关系，其中<synonymy>元素定义属性名之间的同义关系。<combination>定义属性之间的合并或是分裂关系。<table>部分定义与具体表有关的属性名之间的对应关系， <table\_structure>主要针对单表间的表-值冲突、属性-值冲突的情况，<table\_relation> 针对多表之间的联系冲突的情况。

（7）映射文档

映射文档是由源数据结构文档、目数据结构文档和用户定义的转换配置文档三者经数据分析处理而得到的。此文件是用户在进行实际的数据转换时的控制信息，是面向程序处理的，其格式应当方便程序编程。因为进行交换的数据往往非常大，如果采用基于对象的编程模型DOM来处理数据值文档，效率将非常低下甚至可能造成内存溢出。所以在这里，使用了基于事件的编程模型。我们记为MXML，它记录了源SturctureXmlFile的每个结点的匹配行为。

映射文档中，元素<tree>中source部分和target部分表示源表和目标表，元素<node>中id部分和tid部分表示源数据表和目标数据表对应的属性名。而且映射文档是有层次的，它在结点的层次上定义了源数据表和目标数据表结点及它们间的匹配类型。因为当两个或三个异构数据库中的表或是视图进行数据交换时可能会出现不止一种类型的冲突，所以在结点的层次定义匹配类型更加方便冲突的处理。

（8）数据交换模块

映射文档详细给出了数据交换过程中源数据表的每个字段是如何交换到目标数据表中。生成一个目标数据节点所需要的所有信息，包括源数据节点、目标数据节点、映射类型。在这个模块中，将复杂的XML数据映射操作分解为若干个相对简单的子操作，每个子操作的计算过程被封装在映射函数中，映射链将映射函数按执行顺序组合在一起，全部映射函数计算完毕后，将计算结果组装成目标XML数据文档。

在用程序设计语言，具体的编程实现时，根据每一条数据匹配规则可以构造出一棵映射流程树(简称映射树)，只是随着数据映射的类型不同，映射过程所需要的步骤有所不同，因此映射树的复杂程度有所不下同的是一个映射流程树的示意。映射树的根就是目标数据节点，映射树的叶节点由源数据节点组成，映射树的中间节点是映射过程所需要的映射函数组成，节点之间的边是映射链。映射计算过程可以依据映射流程树，按照映射链的指引，逐步调用映射函数对源数据节点进行逐步处理，最后完成到目标数据节点的映射，并写入目标数据文档。

一次完整的数据映射包括三个步骤:

1、数据获取

根据映射树中的链接叶子节点的边记录的路径从源XML文档中获取映射计算中所需要的数据。

2、映射过程

按照映射链所组合的映射函数序列，逐步对源XML数据进行相应的函数计算。

3、数据写入

所有映射函数计算完毕后，按照链接根节点边记录的路径将计算结果写入目标数据文档。

# 基于模型的推理

模型是知识表示的很重要的一种形式， 因为专家经常会基于自己的能力对一个系统的目标或成分如何互相连接（在讨论领域中无论物理的、因果的、相关的或者其他） 进行推理，叙述了人类问题解决者如何建立构思模型，从而帮助它们解决复杂的问题。这个概念是基于模型推理的基础， 一类人工智能技术用来通过分析系统的结构来解决问题， 正如某些模型所指定的。在许多方面， 本章是对构思模型讨论的延续。它也是对专家系统的延续， 因为基于模型的推理可以用来扩展和提高专家系统的能力。为了使专家系统的设计局限于逻辑IF-THEN 规则， 就会严重地忽略表示知识的强有力的方面； 这样的一个专家系统会比它用其他方法能够达到的智能性要小。

事实上， 基于模型的推理在很大程度上， 随着考虑处理有系统仅仅使用基于规则的推理和其他受限形式的推理（例如， 流程图、故障词典和决策树） 创建的空白方面开始存在。例如， 排除一个故障设备故障的基于规则的专家系统可以使用将征兆与潜在故障联系起来的规则。这样一个系统能解释到目前已经得到的经验联合（经常以一种偶然的或偶然事件的方式） 的原因， 可能会丢失一些故障。相比之下， 基于模型的推理会设法确定基于设备的基本结构什么是错误的。使用基于模型推理的系统， 如果结构正确的情况下， 可以规定更有规则的和广泛的范围。这就意味着基于模型推理可能发现新的故障， 并且在没有遇到问题之前解决问题。并且，使用基于模型推理的系统趋于不依赖设备。可以以一般地方式， 即它假设为很多不同的设备布局， 指定一个模型。基于规则的方法不能像这样灵活而是指定设备，意味着需要用不同的规则集去指定每一种新设备的行为。

# 基于不确定的推理实现

决策者在日常通常会面临着很多不确定因素。通常情况下，我们并没有足够的知识来解决手头的问题，因此只有尽力寻找方法来弥补我们在知识和技能方面的缺陷。另一些情况下，我们得到的信息并不精确，或者受到噪声影响；在一些更坏的情况下，我们甚至没有作出决策所需的任何数据。另一方面，或许存在这样的情况：虽然所有数据都是在我们的控制下得到的，但却不存在能指导我们如何处理和分析数据的精细准确的方法。尽管如此，决策者常常可以通过自身的直觉、创造力、想象力和意志力，进而成功地处理不确定因素，并作出较好的决策。

事实上，尽管在解决问题的过程中存在着复杂性和不确定性，但绝大部分人还是有能力作出合理的良好的决策。例如，对于医生而言，他们必须处理来自实验室的不确定的试验结果，以及指出某种疾病的不同程度的症状。尽管如此，医生也必须为病人作出诊断，以便提出一个合理的治疗方案。类似地，专家系统，或者任何一种智能系统，都应该能够描述和分析不确定数据；否则，对于现实生活中的一系列经常不完整、不一致、有噪声影响，甚至完全缺失的数据问题，系统往往无能为力。

事实上，不确定性遍及现实世界，并且也存在于各种技能领域。其根源可以归结于以下两类：不完备的领域知识和不完备的数据信息。在第一种情况下，某一领域的理论可能是并不完善的。因此，专家自然而然需要有一些探索精神，或者根据无科学依据的方法来达到一种令人满意的，但不必是最优的解决方案。在一个基于规则的专家系统中，技能是一系列IF-THEN规则的集合，知识工程师则被迫要建立规则前提和相应结论之间的具体关联。很遗憾，在很多领域这些关联是很微弱的，而且不能被清晰地且充分地理解（确实，科学是不断进步的，一些以观察或实验为依据的、新的、更强的理论不断地被发现，被完善）。此外，在很多领域涉及问题的解决方案经常会存在着相互矛盾的意见。因此，将多种多样的专家观点整合到“统一”的知识库中是一个巨大的挑战。

第二种情况，实际的案例数据可能是不精确的或不可靠的。不可靠的原因可能包含以下几种：由于传感器的不可靠性使得精确的测量是不可能实现的；所谓的实验测试结果只能是近似的；实验报告则为主观的且含糊不清；实验证明可能是相互矛盾的；同时人为的误差和偏见的分类会妨碍到最严谨、最一丝不苟的专家。即使是有可能收集到更精确的数据，这样做也是不切实际或者说是不合理的。一个恰当的例子：对于医生而言依靠一些类型的扩散性的外科手术去获得更好的病人数据是很可能的，但是这样的过程会很耗时，且对病人的健康是有危害的。其另一个弱点或缺陷就是，采集更多信息需要花费时间，而有时需要作出迅速的判断。例如在一些危急时刻，决策者必须依靠不完整的信息，来迅速地作出决策，完全没有足够的时间来收集到全部数据，也就没有足够的时间来对收集到的信息进行严格且谨慎的分析。

# 国内外数据交换现状分析

（1）使用数据库自带工具进行转换

很多数据库管理系统都提供了将外部文件中的数据存到数据库的工具。常见的有ORACLE导入导出工具和SQL Server的DTS工具。

ORACLE导入导出功能的主要作用是对ORACLE数据库管理系统中的数据表、视图或整个数据库进行逻辑备份，导出数据库管理系统中的数据，然后转储为二进制格式的文件(该文件符合的数据规范)，而且也可以将二进制格式的文转换为ORACLE数据库管理系统中的数据，重新生成数据表、视图或是数据库，从而达到数据跨平台移植的目的。

异构服务是集成在ORACLE数据库软件中的功能，它提供了从ORACLE数据库访问其他非ORACLE数据库的通用技术。熟悉ORACLE的用户都很清楚， ORACLE提供通过建立DB Link的方法访问非本地数据库，而异构服务提供了通过建立DB Link使你能够执行ORACLE SQL查询，透明地访问其他非ORACLE数据库里的数据，就像ORACLE访问远程数据库一样。

使用数据库管理系统的提供的数据转换工具的确可以完成一些数据交换的任务，但这些工具有一个共同的缺点，即必须使用数据库管理系统才能完成数据转换工作，通用性较差。也就是说，如果想进行ORACLE有关的数据交换，必须安装ORACLE数据库

（2）利用中间数据库转换

许多的数据库管理系统都提供从数据库管理系统中生成数据对象的脚本，对应有SQL语句，然后用户根据自己数据转换的需要，根据目标数据库管理系统的语法要求，将SQL语句进行修改，改为对应数据对象的脚本，最后在目标数据库管理系统根据脚本，生成新的数据。

这种转换方式也可以实现一定的数据转换功能，但是转换的过程过于复杂，转换的质量也不高。该转换方式也不适合大范围的应用。

（3）设置传送变量转换

这种方式是通过程序设计语言，利用能够操作的数据语句，设置传送变量，将源数据库中或读取到的数据传递给变量，然后再把变量的值传递给目标数据库中的数据。这种转换方式也能实现一定的数据转换功能，但是转换的过程也太复杂，转换的质量也不高，因此效率不高。

（4）通过开发数据库组件或数据库中间件技术转换

中间件是一种独立的系统软件或服务程序，分布式应用软件借助这种软件在不同的技术之间共享资源。中间件运行在操作系统、网络和数据库之上，应用软件的下层，总的作用是为处于自己上层的应用软件提供运行与开发的环境，帮助用户灵活、高效地开发和集成复杂的应用软件。

数据库中间件就是基于数据库管理系统，建立数据组转换的组件。数据库中间件将用于交换和共享的数据“打包”成本地数据库能处理的数据格式或一些特殊格式的能够处理的数据源。

如下图所示，每条双箭头表示一个数据库管理系统到另一个数据库管理系统的中间件，所有的中间件能够完成这种数据库管理系统之间的数据交换任务。需要10个数据库中间件。假设我们参与数据转换的数据库关系系统或是文本类型有N种，那么按照双向图相关理论，应该建立N\*(N-1)/2个数据库中间件。显然这样做很繁琐，效率不高。



图 数据库中间件

（5）基于XML的数据转换

可扩展标记语言XML的产生则正迎合了部门数据整合的需要，这是因为XML具有开放性、数据分离性和把文档的三要素独立开来分别处理等优势。所以现在大部分的数据交换平台是以XML技术为基础，结合其他数据技术来实现的。

基于XML的异构数据交换，是以XML文件为公共数据模型来实现异构数据库管理系统间模式转换和数据交换,如图所示，当某一数据库管理系统中的数据需要转换为其它数据库中时候，只需将它转换为XML文件，就能轻易的将其迁移到各种关系数据库中，具有很好的通用性和良好的扩展性。



图 基于XML的数据交换方式

随着XML技术的应用，出现了两种主要的基于XML技术的数据交换规范：ebXML规范和BizTalk规范。

# 插件技术

插件是遵循统一的预定义接口规范的程序，应用程序在运行时通过接口规范对插件进行调用，以扩展应用程序的功能。软件功能更新采用编译时（assembly-time）更新，首先进行源代码级别代码更新，然后对整个程序进行重新编译，最后发布应用程序的新版本。相对于编译时(assembly-time)更新，运行时（Run-time）更新无需对整个程序代码进行重新编译，插件支持“运行时(run-time)"功能扩展更新，即可通过公布插件的预定义接口规范，从而允许第三方开发插件对软件的功能进行扩展。

插件的本质在于不修改程序主体（平台）的情况下对软件功能进行扩展与加强，当插件的接口公开后，可以制作插件来解决一些操作上的不便或增加新的功能，也就是实现真正意义上的“即插即用”软件开发。

插件技术的优势：

实现真正意义上的软件组件的“即插即用”；

在二进制级上集成软件，减少软件重新编译与发布的时间；

便于扩展，可根据使用要求，随时扩展插件功能。

可较好实现代码隐藏，保护业务领域工作流程和部分核心数据。

“数据中心平台”在视图控制层专门内置了插件层，内置插件包括输入域插件、图标插件、窗体插件、页面插件，异构数据源的适配过程中采用了数据连接适配插件。

军用数据信息管理对于数据信息响应的速度实时性要求高，对系统升级维护的快速性要求高，因此，“即插即用”的插件技术是军用软件升级维护的较好选择，通过离线插件功能扩展开发编译调试测试、实时在线“即插即用”，无需暂停在线业务数据服务即可快速完成升级维护，升级维护边界高效。

# 专家系统

专家系统的两个主要的组成部分是知识库和推理机。知识库包括由规则表示的关于领域的知识。因而， 知识库实际上是数据库， 或是IF-THEN 规则的集合。知识库本身不能够为用户完成任何工作。第二个主要的组成部分（即推理机）， 使专家系统具有处理规则和做出专家建议的能力。推理机是专家系统的中央处理单元，理解这一组成部分对于理解专家系统是如何进行工作的是十分重要的。

CLIPS 是用来开发专家系统的工具。这代表了C 语言集成产品系统。CLIPS 在1986 年由NASA/ 林登·B·约翰逊宇航中心开发。虽然， 这一工具最初用C 语言编写， 但其语言与人工智能编程语言LISP 更为相近。当前在多数领域中使用CLIPS， 这意味着你可以从网站下载CLIPS 的免费版本去开发你自己的专家应用系统。此外， 也可以很自由地下载软件的文档。由于可以免费、高效和便捷地学习， 因此当今CLIPS 成为最为广泛使用的专家系统工具。CLIPS 对在相对短的时间内， 创造许多不同类型的专家系统应用将是一个非常好的选择。

如同多数专家系统的程序， CLIPS 包含三个基本组成部分：

1） 事实表： 用于存储数据的工作内存；

2） 知识库： 规则的集合；

3） 推理机： 对规则执行的控制。

# 用户体验及交互设计技术

Web中的用户体验是指用户在访问平台的过程中的全部体验，它包括平台是否有用，疑惑或者bug程度，功能是否易用、简约，界面设计和交互设计是否友好等方面。

技术的解决途径：一是采用一致性设计原则，对功能进行恰当的分类和组织，帮助用户探索和尝试，允许用户犯错误，提供实用的帮助。二是为第一次使用系统的学员提供向导式界面，帮助用户高效的完成指令输入，使界面符合用户的使用习惯和预期，减少用户出错和适时的提醒，减少用户的等待感。



图1 用户体验与网页设计层次图

# 软件开发规范流程

软件开发流程一般包括需求定义、需求分析、外部设计、内部设计、编码、单元测试、结合测试及系统测试8 个阶段。无论是业务开发流程还是架构开发流程， 都是“ V”字模型。也就是说，代码对内部设计的反应如何，需通过单元测试来进行验证；相应的功能结合测试对应的是外部设计； 业务结合测试对应的是功能定义与业务流程； 系统测试对应的是需求整理的内容。

在软件开发过程中有很多任务（工作） 对象，特别是大型软件开发。而对软件架构师来说主要关注的任务。另外，软件架构师还必须了解项目启动阶段与需求阶段的各种重要信息，如系统鸟瞰图等。

# 快速匹配技术

在Word 文档解析的基础上，字符串的模式匹配可以应用于搜索和查找特定的字符串位置。对于长度为n 的文本串T=T0T1T2…Tn-1，假设模式串P=P0P1P2…Pm-1 的长度为m(n ≥ m)，在T 中查找模式串P 首次出现的位置。如果在文本串T 总存在等于模式串P 的子串，则匹配成功，返回模式串P 首次出现在文本串T 中的位置；否则，称为匹配失败。这个过程称为模式匹配。

目前，国内外已提出了不少模式匹配算法。典型的BF 算法是从文本串T 的开始位置开始逐个字符依次匹配，算法的时间复杂度是O(n×m)，BF算法虽然简单但效率很低。KMP 算法在匹配过程中遇到不匹配时，不产生回溯，而是根据已有的部分匹配结果，令P 向右滑动到某个位置，然后重新开始进行匹配。这在一定程度上提高了匹配效率，减少了匹配次数，但时间复杂度依然达到O(n+m)。BM 算法[8] 是由Robert S. Boyer 教授和J Strother Moore教授提出的一种新的字符串匹配算法。该算法从模式串P 的尾部与文本串T 自右向左开始匹配。如果不匹配，则根据模式串P 决定的好后缀规则和坏字符规则进行右移，降低无效的匹配次数，然后继续与文本串T 进行匹配，直至匹配成功或者文本串T 匹配结束。

实践中，BM 算法比KMP 算法的实际效率更高，因而得到了广泛应用。BMH 算法进一步解决了BM 算法中模式串P 有可能左移的问题。在BMH 中，无论失配发生在当前文本串T 中的什么位置，都由文本T中和模式串P 的尾字符对应的字符决定右移距离。它摒弃了好后缀规则后，省去了求最大值的计算。