目录

第0页

第 1 章　　设计依据与原则

本项目涉及到系统必须以实用为原则。　　采用成熟的并且通过实践考验的先进技术和解决方案。

1.1 功能性

与一组功能及其指定的性质有关的一组属性，具体包括：

适合性：与规定任务能否提供一组功能以及这组功能的适合程度有关的软件属性。

准确性：与能否得到正确或相符的结果或效果有关的软件属性。

互用性：与同其他指定系统进行交互的能力有关的软件属性。

依从性：使软件遵循有关的标准，约定，法规及类似规定的软件属性。

安全性：与防止对程序及数据的非授权的故意或意外访问的能力有关的软件属性。

充分考虑系统的安全防护，具备较强的数据管理机制和控制能力

1.2 可靠性

与在规定的一段时间和条件下，　　软件维持其性能水平的能力有关的一组属性，　　具体包括：

成熟性：与由软件故障引起失效的频度有关的软件属性。

容错性：与在软件故障或违反指定接口的情况下，维持规定的性能水平的能力有关的软件属性。

易恢复性：与在失效发生后，重建其性能水平并恢复直接受影响数据的能力以及为达此目的所需的时间和能力有关的软件属性充分考虑性价比。

1.3 易用性

与一组规定或潜在的用户为使用软件所需作的努力和对这样的使用所作用的评价有关的一组属性，具体包括：

易理解性：与用户为认识逻辑概念及其应用范围所花的努力有关的软件属性。

易学性：与用户为学习软件应用所花的努力有关的软件属性。

易操作性：与用户为操作和运行控制所花努力有关的软件属性。

1.4 效率

与在规定的条件下，软件的性能水平与所使用的资源量之间关系有关的一组属性，具体包括：

时间特性：与软件执行其功能时响应和处理时间以及吞吐量有关的软件属性。

资源特性：与在软件执行其功能时所使用的资源数量及其使用时间有关的软件属性。

1.5 可维护性

与进行指定的修改所需的努力有关的一组属性，具体包括：

易分析性：与为诊断缺陷或失效原因急为判定待修改的部分所需努力有关的软件属性。

易改变性：与进行修改，排除错误或适应环境变化所需努力有关的软件属性。

稳定性：与修改所造成的未预料结果的风险有关的软件属性。

易测试性：与确认已修改软件所需的努力有关的软件属性。

1.6 可移植性

与软件可从某一环境转移到另一个环境的能力有关的一组属性，具体包括：

适应性 :与软件无需采用有别于为该软件准备的活动或手段就可能适应不同的规定环境有关的软件属性。

易安装性 :与在指定环境下安装软件所需努力有关的软件属性。

遵循性 :使软件遵循与可移植性有关的标准或约定的软件属性。

易替换性 :与软件在该软件环境中用来替代指定的其他软件的机会和努力有关的软件属性。

1.7 标准化

本项目涉及到的各个系统模块设计、系统性能、代码编写等应符合中国有关软件项目的标准化的要求：

1.软件开发过程中作业标准化。

2.确定每个作业的表示形式。

3.确定每个文档资料的格式。

4.规定组符号。

5.根据软件开发经验，制定出大家能够接受的开发原则和进度。

第 2 章　　系统总体架构设计

2.1 总体设计要求

根据市场反应情况和目前软件系统主流的设计思路和方向，本系统总体设计要求如下：

系统采用 B/S 架构进行设计。

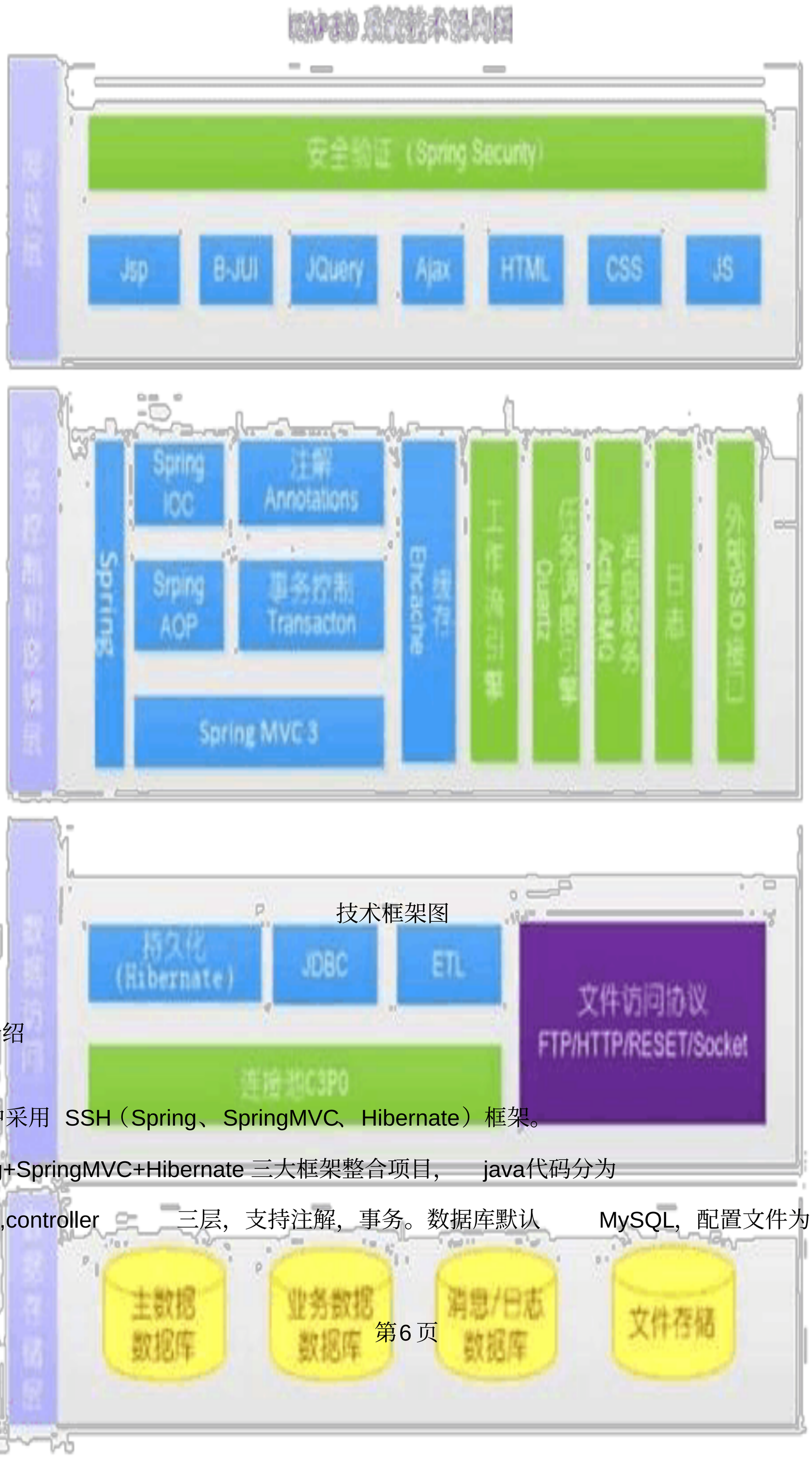
基于 J2EE 平台开发。

采用主流技术框架　SSH（Spring、SpringMVC、Hibernate）。

系统支持主流的关系型数据库：　Mysql、Oracle、SqlServer等。

2.2 系统技术架构

2.2.1技术架构图



技术框架图

2.2.2框架介绍

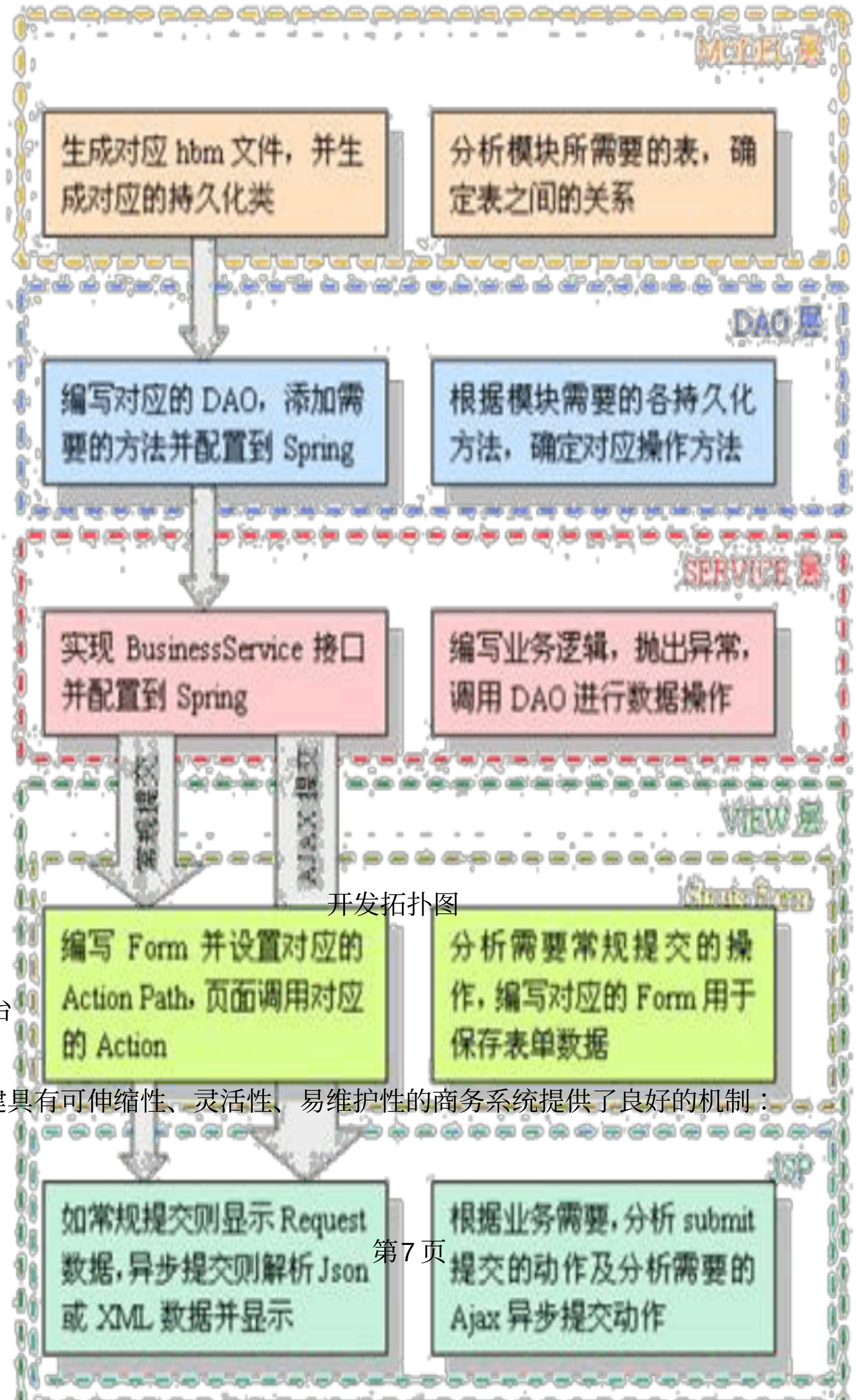
系统中采用 SSH（Spring、SpringMVC、Hibernate）框架。

Spring+SpringMVC+Hibernate 三大框架整合项目， java代码分为 dao,service,controller 三层，支持注解，事务。数据库默认 MySQL，配置文件为 src下的

config 资源包中的 db.properties ，以 KEY VALUE 形式保存数据库连接属性，方便移植修改。

Hibernate 是一款优秀的 ORM 框架，能够连接并操作数据库，包括保存和修改数据。Spring MVC 是 Java的 web 框架，能够将 Hibernate 集成进去，完成数据的 CRUD。Hibernate 使用方便，配置响应的 XML 文件即可。

2.3 系统业务逻辑结构



开发拓扑图

2.4 J2EE研发平台

J2EE为搭建具有可伸缩性、灵活性、易维护性的商务系统提供了良好的机制：

J2EE是一套全然不同于传统应用开发的技术架构，包含许多组件，主要可简化且规范应用系统的开发与部署，进而提高可移植性、安全与再用价值。

J2EE核心是一组技术规范与指南，其中所包含的各类组件、服务架构及技术层次，均有共同的标准及规格，让各种依循J2EE架构的不同平台之间，存在良好的兼容性，解决过去企业后端使用的信息产品彼此之间无法兼容，企业内部或外部难以互通的窘境。

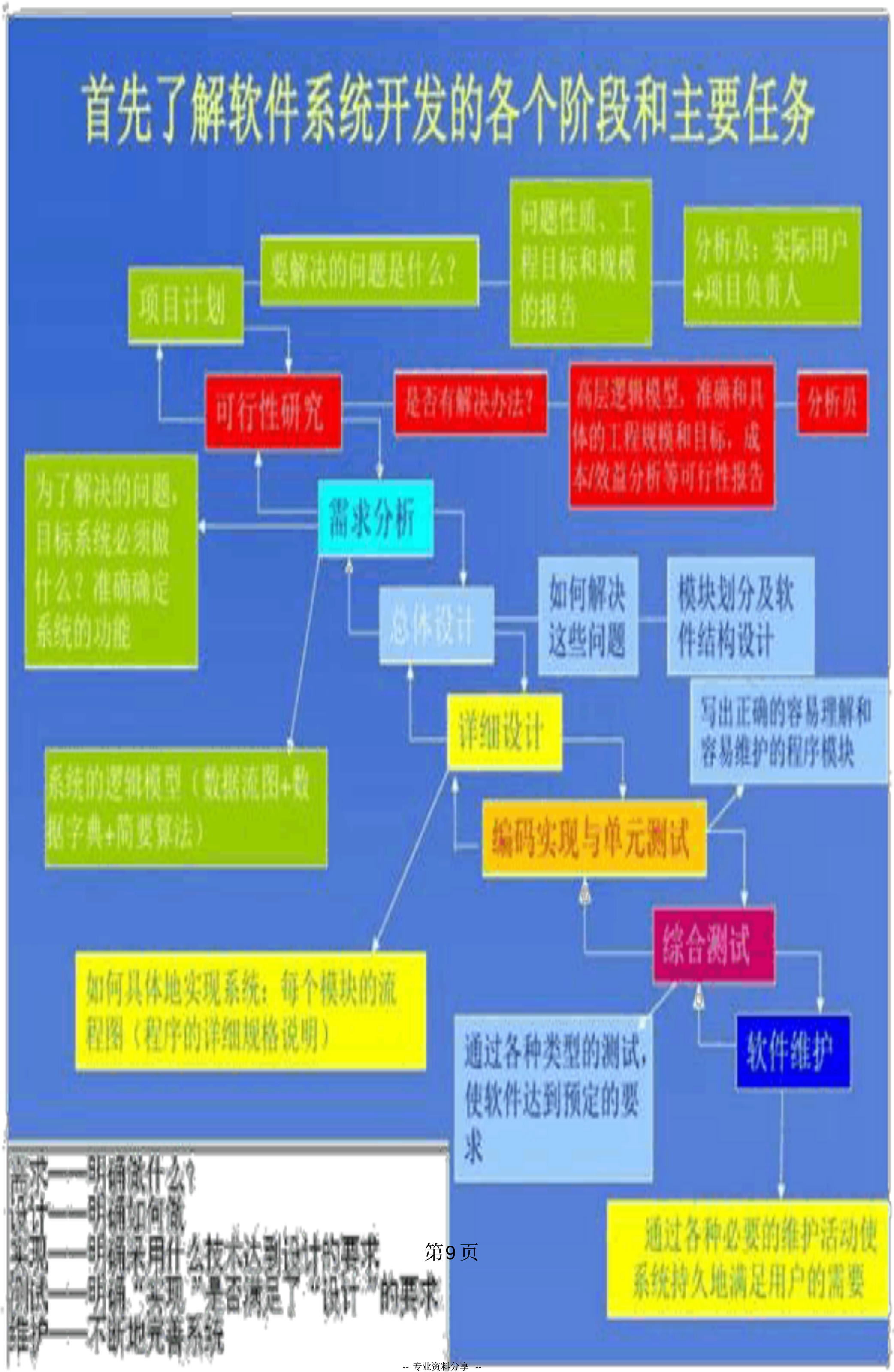
J2EE组件和“标准的”Java类的不同点在于：它被装配在一个J2EE应用中，具有固定的格式并遵守J2EE规范，由J2EE服务器对其进行管理。J2EE规范是这样定义J2EE组件的：客户端应用程序和applet是运行在客户端的组件；Java Servlet和Java Server Pages (JSP)是运行在服务器端的Web组件；Enterprise Java Bean (EJB)组件是运行在服务器端的业务组件。

2.5 Web应用服务环境

严格意义上Web服务器只负责处理HTTP协议，只能发送静态页面的内容。而JSP，ASP，PHP等动态内容需要通过CGI、FastCGI、ISAPI等接口交给其他程序去处理。这个其他程序就是应用服务器。

比如Web服务器包括Nginx，Apache，IIS等。而应用服务器包括WebLogic，JBoss等。应用服务器一般也支持HTTP协议，因此界限没这么清晰。但是应用服务器的HTTP协议部分仅仅是支持，一般不会做特别优化，所以很少有见Tomcat直接暴露给外面，而是和Nginx、Apache等配合，只让Tomcat处理JSP和Servlet部分。

2.6 系统流程设计


首先了解软件系统开发的各个阶段和主要任务
项目计划
要解决的问题是什么？
问题性质、工程目标和规模的报告
分析员：实际用户+项目负责人
可行性研究
是否有解决办法？
高层逻辑模型，准确和具体的工程规模和目标，成本/效益分析等可行性报告
分析员
为了解决的问题，目标系统必须做什么？准确确定系统的功能
需求分析
总体设计
如何解决这些问题
模块划分及软件结构设计
详细设计
写出正确的容易理解和容易维护的程序模块
系统的逻辑模型（数据流图+数据字典+简要算法）
编码实现与单元测试
综合测试
如何具体地实现系统：每个模块的流程图（程序的详细规格说明）
通过各种类型的测试，使软件达到预定的要求
软件维护
通过各种必要的维护活动使系统持久地满足用户的需要
需求——明确做什么？
设计——明确如何做
实现——明确采用什么技术达到设计的要求
测试——明确“实现”是否满足了“设计”的要求
维护——不断地完善系统

第 3 章　　关键技术解决方案

3.1 基本技术介绍

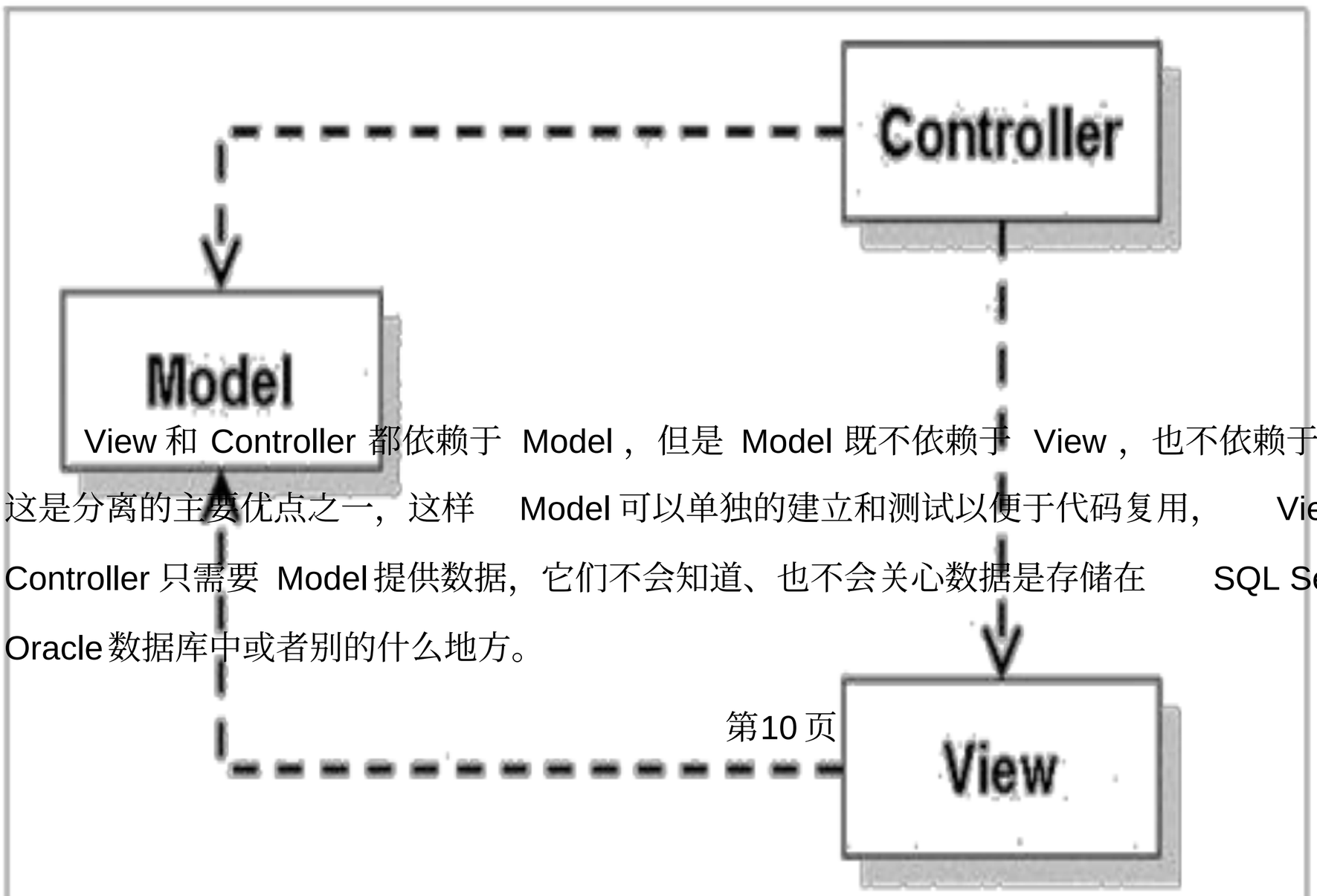
基于当前 Web 应用程序开发面临的问题，项目结合目前比较流行的开源框架　　SSH（Spring、Struts、Hibernate)，具体讨论其基本相似性及有关基本概念，提出了一种开发 JavaEE Web应用的轻量级解决方案，此系统架构可以在短期内搭建结构清晰、可复用性好、可扩展性好、维护方便的　Web 应用程序。

MVC 模式

MVC 模式是一个用于将用户界面逻辑与业务逻辑分离开来的基础设计模式，它将数据处理、界面以及用户的行为控制分为：　　Model（模型）－ View（视图）－ Controller　　（控制器）。 Model ：负责当前应用的数据获取与变更及相关的业务逻辑。可用　　JAVABEAN 来体现；View：负责显示信息。可以使用　　JSP、VELOCITY 模板等技术。

其优点有：

Controller　：负责收集转化用户的输入。常用一个　　SERVLET 来实现；



View 和 Controller 都依赖于 Model，但是 Model 既不依赖于 View，也不依赖于 Controller，这是分离的主要优点之一，这样　Model 可以单独的建立和测试以便于代码复用，　View 和 Controller 只需要 Model 提供数据，它们不会知道、也不会关心数据是存储在　SQL Server还是 Oracle 数据库中或者别的什么地方。

3.1.1三层技术

3.1.1.1 三层结构框架及功能

由于传统的二层 C/S 结构存在以下几个局限：它是单一服务器且以局域网为中心的，所以难以扩展至广域网范围或 Internet 的大型应用模式；难以管理大量的客户机；受限于供应商，整个系统与特定的应用程序联系紧密；软、硬件的组合及集成能力有限。因此 ,在乐清电子政务应用系统中以三层结构体系为主。

三层结构是将应用功能分成表示层、业务逻辑层和数据层三部分。其解决方案是对这三层进行明确分割，并在逻辑上使其独立。各层说明如下：

表示层—担负用户与应用间的对话功能， 通过浏览器模式实现表示层， 组成的 B/S 结构；或使用可以自动更新的瘦客户端软件实现表示层，组成基于三层体系的“ 客户／服务器 ”结构；

业务逻辑层—包含了具体的业务处理逻辑程序相当于应用的本体；

数据层—负责管理对数据库数据的读写。主要是利用大型关系型数据库进行迅速、大量的数据处理。

3.1.1.2 选用三层结构的优点

选用三层结构具有以下优点：

系统管理简单 ,大大减少客户机维护工作量。

基于 B/S 结构的应用模式无需客户端维护工作；基于“ 客户／服务器 ”结构的客户端可以实现自动更新下载，也无需客户端维护工作。

具有灵活的硬件系统构成

对于各个层可以选择与其处理负荷和处理特性相适应的硬件，方便的实现负载均衡。清晰、合理地分割三层结构并使其独立 ,可以使系统构成的变更非常简单。因此 ,被分成三层的应用基本上不需要修正。

提高程序的可维护性

三层 B/S 结构中，应用的各层可以并行开发，各层也可以选择各自最适合的开发语言。因为是按层分割功能，所以各个程序的处理逻辑变得比较简单。

进行严密的安全管理

涉密的关键应用的安全管理非常重要。在三层 C/S 结构中，识别用户的机构是按层来构筑的，对应用和数据的存取权限也可以按层进行设定。例如，即使外部的入侵者突破了表示层的安全防线，若在功能层中备有另外的安全机构，系统也可以阻止入侵者进入其他部分。

3.1.1.3 中间技术

消息中间件

采用消息中间件技术、基于 J2EE的三层结构构建面向各级单位的数据交换体系中。消息中间件是位于平台（硬件和操作系统）和应用之间的通用服务，具有标准的程序接口和协议。针对不同的操作系统和硬件平台，它们可以有符合接口和协议规范的多种实现。消息中间件起到了一个“平台＋通信”的作用，一方面使进一步的开发工作可以构建在一个统一的开发环境（平台）之上，不必关心具体的网络编程技术细节，大大简化了设计和编程工作；另一方面，中间件完全负责消息通信，用户只需关注于业务系统的运行、开发，有效地提高了效率。

消息中间件通信传输类型：

可靠传输可以在保证报文的正确性的前提下实现相对的实时传输。每个报文有相对的生命周期，在网络超时或者接受方宕机时终止发送请求，即报文有可能丢失或非顺序到达。可靠传输对处理机和网络的开销较小，一般适用于对传输速率要求较高的准实时系统，而对报文的丢失有一定的冗余度。

确保传送可以保证信息的无丢失、按顺序传送。在信息的发送者与接受者之间的网络出现中断或者接受者方的机器出现故障，在网路恢复连接后，仍然能保证在故障时期内的所有信息按顺序的正确到达。确保传送的高可靠性是以较多的资源开销（处理机、网络）作为代价的。因此，确保传送一般是用于传送频率比较低，但传送可靠性要求高的信息传输，如重要文件的传输等。该传输类型类似于电子邮件的传输方式。

数据中间件

在综合数据支撑平台中，为了整合桌面型数据库成为一个可共享的具有用户和权限管理的虚拟数据库，需要采用数据中间件以屏蔽掉数据节点分布、数据库表异构特性，实现虚拟数据库合理的软件层次结构。

3.1.1.4 安全应用技术

为了在电子政务系统的应用层、网络层实施细粒度的访问控制，实现对用户的身份鉴别、实现信息的保密性、完整性、真实性和抗抵赖性等保护，采用当今流行的高强度安全策略——数字证书技术。应用系统可以基于数字证书以及相关的经国家有关部门认可的密码算法认证登录系统的用户的真实身份，进行数字签名和验证签名，采用数字签名技术解决抗抵赖性和数据完整性的问题，利用安全系统提供的加密算法，解决信息的保密性问题。

对重要数据库的访问，还要通过安全代理，对访问者的身份基于数字证书进行高强度的认证，对其访问应用系统的请求进行确认，如果该用户没有访问的权限，其访问请求将被安全代理拒绝。同时，在安全代理服务器上还可以完成包括包过滤、加密、解密等技术，从而实现权限确认和数据的密存密传功能。

3.2 技术路线的可行性和解决关键技术的途径

三层应用构架是一种成熟的开发模式，可以应用到电子政务中，针对行文应用的特殊要求，建议 Domino 平台这一成熟的体系，以确保电子政务的正常运作。

Java技术是一种成熟的技术，已经得到广泛的应用，J2EE 技术规范已经得到大的中间件生成厂商如 BEA 公司、IBM 公司的产品化支持。

中间件技术是软件产品的发展方向，现在市场上已有大量的产品可供选择，因此在结合电子政务需求开发数据中间件是可行的，在数据交换体系中采用消息中间件已是可行的，符合发展方向。

安全应用技术是电子政务中的一种重要指标，国内许多单位进行过大量的研发工作，有的已形成了产品，因此也具有可行性。

虚拟数据库是解决数据共享、 系统平滑过渡的必又之路， 结合数据库技术和中间件技术，一定能达到目标，创优质工程。

3.3 数据资源解决方案

对不能（不方便）共享的桌面型数据库，为暂时维持现有应用不变且又能提供数据资源共享，提出了一个完备的基于整体应用的数据库解决方案——即虚拟数据库解决方案。其基本思想是将分散的、局部的桌面形数据库（ Foxpro 、 Access）利用网络资源以及虚拟数据库应用将它们在逻辑上统一起来，实现呈现给用户一个完整的、统一的数据库访问模式，同时提供数据资源的用户和权限管理功能， 即对用户以及应用程序来说就好像访问大型关系型数据库一样方便地访问数据资源，而不是在访问分散于不同服务终端的数据库，所有的处理都将在虚拟数据库构架中完成，不需要用户或应用程序涉及任何底层的输入。

3.4 高性能页面响应解决方案

从系统角度来理解软件，确定对所开发系统的综合要求，并提出这些需求的实现条件，以及需求应该达到的标准。 这些需求包括： 功能需求（做什么），性能需求（要达到什么指标），环境需求（如机型， 操作系统等），可靠性需求（不发生故障的概率）， 安全保密需求， 用户界面需求，资源使用需求 （软件运行是所需的内存、 CPU 等），软件成本消耗与开发进度需求，预先估计以后系统可能达到的目标。

3.5 安全性解决方案

安全性测试主要是测试系统在没有授权的内部或者外部用户对系统进行攻击或者恶意破坏时如何进行处理，是否仍能保证数据和页面的安全。测试人员可以学习一些黑客技术，来对系统进行攻击。 另外，对操作权限的测试也包含在安全性测试中。具体测试内容如下：

o 执行添加、删除、修改等动作中是否做过登录检测。

o 退出系统之后的操作是否可以完成。

o 所有插入表单操作中输入特殊字符是否可以正常输正常存储，特殊字符为： ！?# ￥%,, — * （） ~ —— -+=[]{} 、| ；：‘ ”？ / 《》 <> ，。

- 在带有参数的回显数据的动作中更改参数，把参数改为特殊字符并加入操
- 测试表单中有没有做标签检测，标签检测是否完整。

第 4 章 系统安全解决方案

4.1 物理安全

保证计算机系统安全，可靠地运行，确保系统在对信息进行采集、传输、存储、处理、显示、分发和利用的过程中不会受到人为或自然因素的危害而使信息丢失、泄漏和破坏，对计算机系统设备、通信与网络设备、存储媒体设备和人员所采取的安全技术措施，实体安全包括环境安全，设备安全和媒体安全三个方面。

环境安全包括受灾防护、区域防护，设备安全包括设备防盗、设备防毁、防止电磁信息泄露、防止线路截获、抗电磁干扰、电源保护等，媒体安全是媒体数据和媒体本身。

4.2 网络层安全

为保护数据处理系统而采取的技术的和管理的安全措施，保护计算机硬件、软件和数据不会因偶然和故意的原因而遭到破坏、更改和泄露。

4.2.1防火墙策略

防火墙指的是一个由软件和硬件设备组合而成，在内部网和外部网之间专，用网与公共网之间的界面上构造的保护屏障，是一种获取安全性方法的形象说法，它是一种计算机硬件和软件的结合，使 Internet 与 Intranet 之间建立起一个安全网关（ Security Gateway），从而保护内部网免受非法用户的侵入，防火墙主要由服务访问规则、验证工具、包过滤和应用网关 4 个部分组成，防火墙就是一个位于计算机和它所连接的网络之间的软件或硬件，该计算机流入流出的所有网络通信和数据包均要经过此防火墙。

4.2.2 拒绝服务攻击的防范

分布式拒绝服务 (DDoS:Distributed Denial of Service) 攻击指借助于客户 / 服务器技术，将多个计算机联合起来作为攻击平台，对一个或多个目标发动 DDoS 攻击，从而成倍地提高拒绝服务攻击的威力。通常，攻击者使用一个偷窃帐号将 DDoS 主控程序安装在一个计算机上，在一个设定的时间主控程序将与大量代理程序通讯，代理程序已经被安装在网络上的许多计算机上，代理程序收到指令时就发动攻击，利用客户 / 服务器技术，主控程序能在几秒钟内激活成百上千次代理程序的运行。

第 5 章 网络系统设计

5.1 基本要求

本系统所有涉及软件要求基于 J2EE 平台开发，并且达到以下要求：

系统将采用 B/S 结构。

系统将采用多层架构的体系结构。

系统中采用 SSH（Spring、 SpringMVC、 Hibernate）框架。

5.2 应用设计

本方案采用多层架构技术，实现项目的可扩展性、可维护性，以及结合其他相关技术保障项目能成功实施。 MVC 模式是一个用于将用户界面逻辑与业务逻辑分离开来的基础设计模式，它将数据处理、界面以及用户的行为控制分为： Model （模型）－ View （视图）－ Controller （控制器）。

1、 Model：负责当前应用的数据获取与变更及相关的业务逻辑， 可用 JAVABEAN 来体现。

2、 View：负责显示信息，可以使用 JSP、VELOCITY 模板等技术。

3、 Controller ：负责收集转化用户的输入，常用一个 SERVLET 来实现。

5.3 存储设计

提供高可靠性的数据存放， 通过存储系统的可靠性设计以及磁盘镜像、 RAID 技术，保证存储介质内数据的可靠性。

第 6 章　　软硬件环境设计

6.1 硬件环境

6.1.1服务器硬件环境配置

服务器端：			
	硬件：		
		机型：	
		CPU：	E5-4603 2.20GHz
		内存：	8.00GB
		硬盘：	3TB
	软件：		
		操作系统： Windows Server 2008 R2 Enterprise 数据库： MySQL 5.5.43 支撑软件： Apache、jdk 、 TeamViwer、 rar	

6.2 软件环境及开发环境

软件环境解决方案主要包括操作系统的选择、数据库环境、开发工具及程序设计语言、测试工具、版本控制工具。

6.2.1操作系统的选择

Windows:向后兼容性、广泛的外围兼容性、多显示器支持、多任务处理等。

主流操作系统对比表

序号	内容	UNIX	Windows	Linux
1	可管理性	较好的可管理性	很好的可管理性	可管理性好，且开放源代码，必要时可进行源码级修改。
2	可维护性	系统维护难度较大，服务器可靠性高，支持 24 小 时长 时 间不 间断 运行。	系统维护难度较小，维护软件简单易用，但是服务器整体稳定性稍低。	系统维护难度较大，有 维护 软件 工具 可选，服务器稳定，支持连续 24 小时不间断运行。

第18 页

6.2.2开发工具及程序设计语言

代码编写： MyEclipse

编写语言： Java（后台）、 B-JUI(前端)

数据库开发： MySQL5.5.43

6.2.3测试工具

功能测试自动化： QTP、 Selenium、Loadrunner、Jmeter等。

测试管理工具： MQC、禅道、 JIRA 等。

6.2.4版本控制工具

版本控制工具： SVN

版本控制是对已做成的软件在发展过程中的一种质量管理，各大公司对自己的软件均有一套版本控制方法。我们开发的软件系统绝不是“一锤子买卖”，推出了第一期软件的试用版，还会有第二期软件补充进来，两期软件到一定阶段都将定为正式版，而且今后还会继续发展，到一定时候还要更新。何时定为正式版，何时宣布版本升级，都需要有明确的要求和界限，两个版本之间的任何修改和维护都需要一套管理办法。升级也好，更新也好，都需要考虑与原来版本的兼容，以保护用户的投资利益。