火龙果--高级软件架构设计

2011年7月7日
2:01

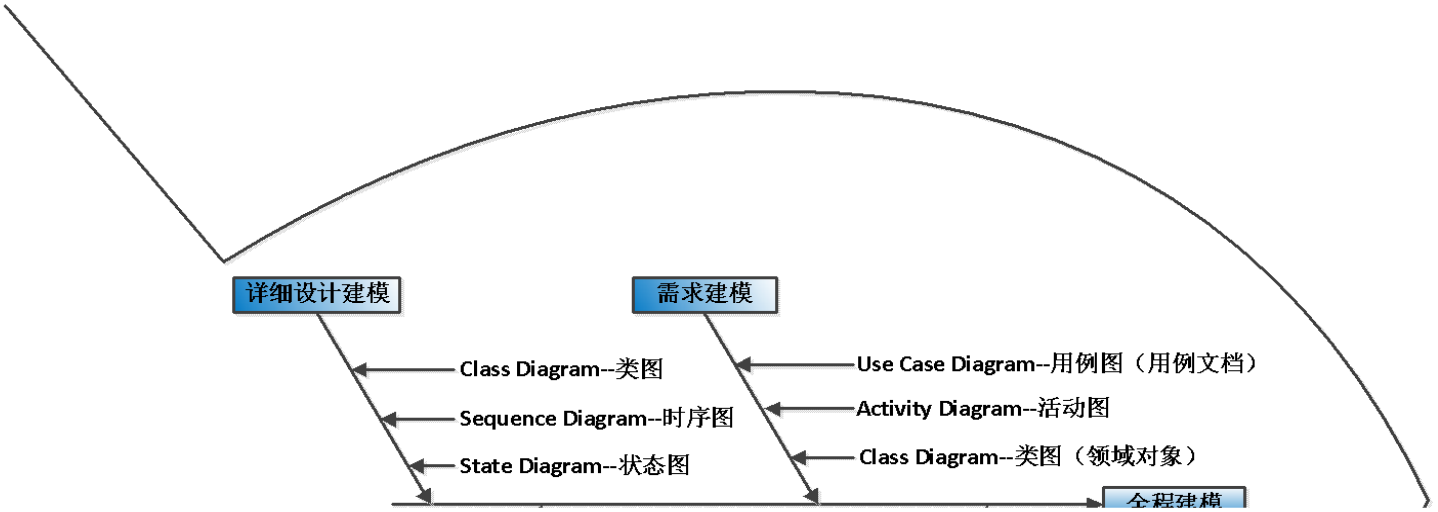
一、培训前问题
1、学员简介
2、用便签纸写下培训期望值
3、划分小组，确定小组长

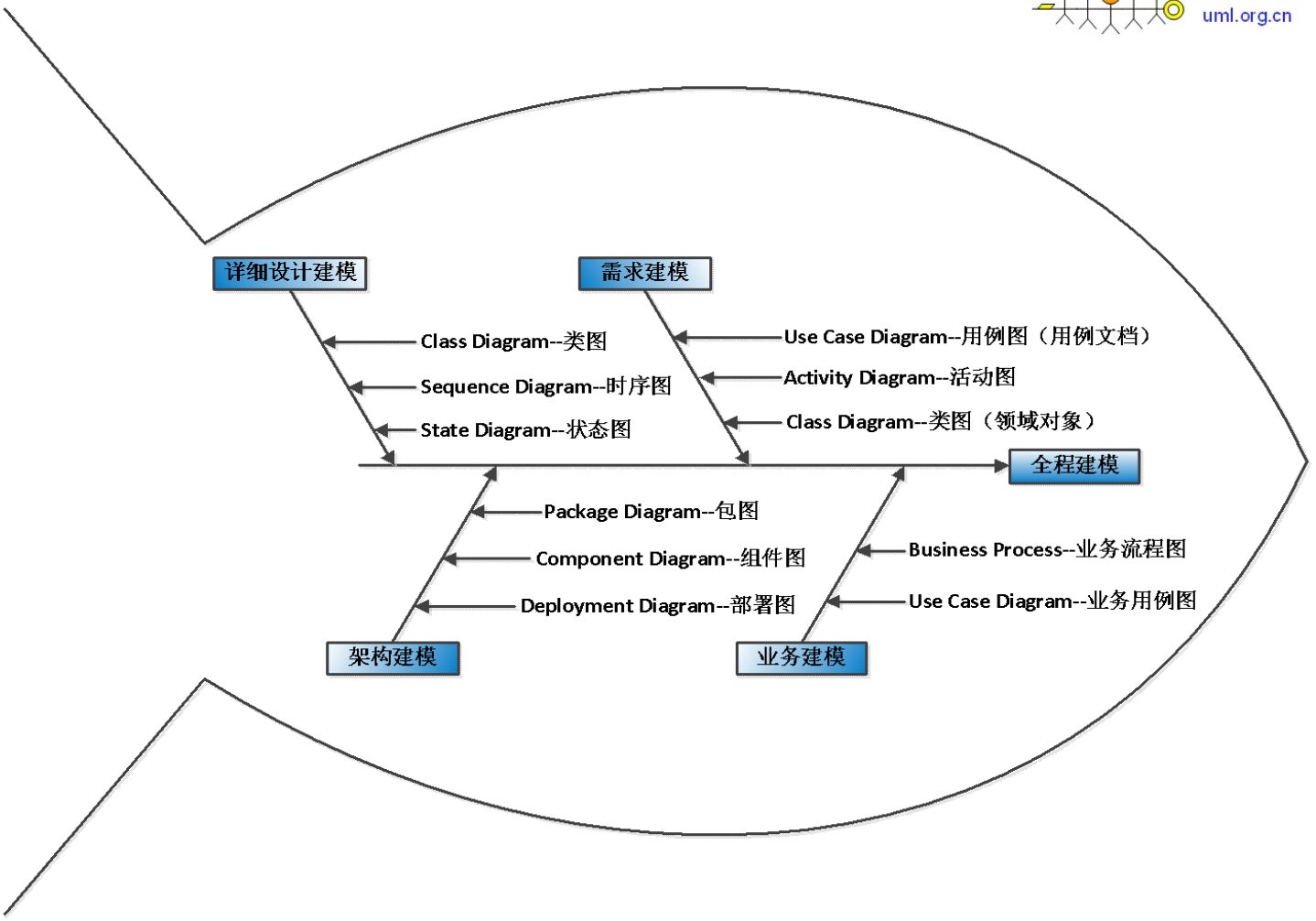
二、上课时间
09:00--12:00
13:00--16:00

二、Day1主题
1、开发过程分析？如何确保需求是完备的，为详细设计提供有价值的信息(强调指导和限制)
2、开发多条线索？
3、架构的重要性？
4、软件架构的重构？
5、开发团队的构成？
6、架构师的6项职责？
7、架构师和项目经理的分工和协作？。
8、架构的定义（RUP基于决策集合）？架构的文档？
9、成功架构的关键策略？
10、如何验证架构？
11、开发架构的过程？

三、分析目前架构设计的问题？
分析软件项目的延期原因？
架构问题
1、不能满足业务需求
2、功能可扩展性差
3、系统稳定性差
4、开发成本大
5、可维护性差：开发的过程管理（缺乏有质量地、和代码同步的设计文档）
技术选型有问题，无法进行有效地测试（没有单元测试）
6、可复用性差
7、学习成本高

四、专业化开发团队特点
1、开发流程化
2、过程控制化
3、项目管理化





五、需求层次：

业务需求：业务目标，KPI，业务模型

文档：产品型（Vision）

项目型（POS）Project overview statement

用户需求：用户需求文档（业务流程，业务用例，业务实体）分析出Feature List

业务流程图（跨职能流程图或UML活动图）

文档：用户访谈备忘录，用户需求分析文档

软件需求：功能性需求，非功能性需求（质量属性），设计约束

文档：SRS（需求规格说明书）

六、架构设计的准入条件

1、完备的需求规格说明书：

A、文档格式满足标准（RUP）

B、呈现的内容标准化：

功能性需求(量化功能点，交互式地描述功能点)、非功能性需求、设计约束

C、需求是否经过验证：需求原型，需求评审

D、是否有需求建模结果：UML用例图（注：使用用例文档描述业务细节）、UML活动图、领域模型图

高质量的文档？

1、文档模板

2、文档规范

3、范例

4、文档评审

5、统一的**分析设计方法**

RUP需求模板：

需求管理计划（rup_rmpln.dot）

业务需求文档（rup_vision.dot）

用户需求文档（rup_bucr.dot）

SRS-用例文档（rup_srs.dot，rup_ucspec.dot）--软件需求文档
补充需求文档（rup_sspec.dot）--软件需求文档
术语列表（rup_gloss.dot）

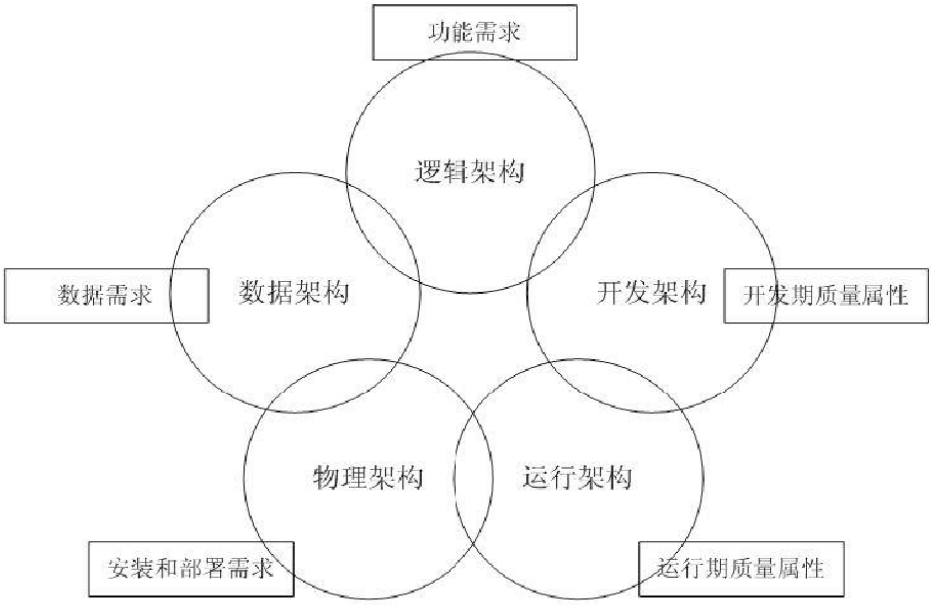
七、架构设计流程
输入：需求规格说明书（SRS），企业架构规范
输出：架构设计文档，原型，模型（UML模型，Visio模型）
A、概念性架构设计
确定关键需求
进行概念性架构设计（主要任务是分析）
1、分析关键用例
2、明确架构模式
3、质量属性分析
B、实际架构设计
细化架构：5种架构图分析设计
验证架构
方法：评审（开发期质量属性评价），测试（运行期质量属性评价）

八、架构模式
8种构架模式
实际项目关于架构模式的选择，采用多种架构模式
J2EE项目：Struts2+Hibernate+Spring
.NET项目：ASP.NET MVC+Unity+Entity Framework(LINQ)+WCF
MVC+分层+元模型

九、架构设计过程
A、概念性架构设计
1、需求分析要求：
功能性需求：交互式描述，清晰的责任
非功能性需求：定性，业务场景化，定量
例：性能、可靠性、可用性、安全

Performance	Throughput
	Response time

• Throughput
The rate at which the system performs its tasks. This can be expressed, for example, in
the number of transactions per minute.
Example: The system shall accommodate 1,000 booked flights per minute.
• Response time
非功能性需求中的响应时间设计：
确定在哪些功能，哪些请求响应步骤涉及对对响应时间的要求
How fast the system responds to events.
Example: Average system response time should be less than two seconds.
Example: Average time of returning a list of flights shall not be greater than ten seconds.



B、实际架构设计
1、架构设计五种视图
A、逻辑架构（业务架构）
-->标识领域范围
从需求规格说明书入手
通过遗留的软件增强认识
->A、标识业务范围（重点：业务切割）
A-1、基于角色
通过多种方式：
角色：注册用户、卖家、后台管理员、客服
如何确定角色，使用以下问题进行分析：
谁使用系统的主要功能？
谁改变系统的数据？
谁从系统获取信息？
谁需要系统的支持以完成日常工作任务？
谁负责维护、管理并保持系统正常运行？
系统需要应付（处理）哪些硬设备？
系统需要和哪些外部系统交互？
谁（或什么）对系统运行产生的结果感兴趣？
有没有自动发生事件

A-2、业务领域
同种类型行为、相同（相似）技术、环境、

A-3、组织+领域（ERP）：生产管理、仓库管理、采购、供应链、财务、客户关系管理、人力资源
系统-->子系统-->模块

B、形成初步分解的业务模块
C、结合架构模式
D、最终形成面向系统实现的子系统、模块

-->细化架构元素
A、结合SRS中用例（功能点）和模块进行细化组件
组件类型：
1）、业务组件：进行对象交互的责任分析，通过归纳把相关的责任定义成业务组件
提示：分析在用例文档中**系统的处理动作**，将操作**相同领域对象**的动作抽象成业务组件
注释：要定义业务组件和代码之间的映射规则，例如采用COM、CORBA、EJB组件

Java语言：一个业务组件<-->一个包，C#语言：一个业务组件<-->一个名
案例练习：
取款用例：分析用例文档中的交互、文档中系统执行的操作，将分析出的的业务上相关操作进行组合，定义形成业务组件
用户组件、帐户组件、交易组件

2）、服务性组件（技术型）
例：异常处理、事务管理、缓存、加解密、校验、通信、安全、日志、定时触发、状态、消息通信、系统监控、配置管理、工作流
日志框架：**Simple Logging Facade for Java (SLF4J)**
已有的日志实现：Log4J、common Logging、JDK Log

3）、展现组件（主要用于提供和系统交互）
如NI的展现组件产品(www.ni.com)，.NET平台的UI组件库（www.infragistics.com）

B、分析设计接口（接口指的是接口程序）
层间接口：基于业务或技术的交互要求来设计接口
组件间接口：基于业务或技术的交互要求来设计接口
注意：接口设计强调应满足系统的低耦合和良好的扩展性
例：接口程序在系统具体实现时，可用像Socket、Web Services、消息中间件、设计模式中的Facade模式实现。
a、User interfaces
b、Hardware interfaces
c、Software interfaces
d、Communications interfaces

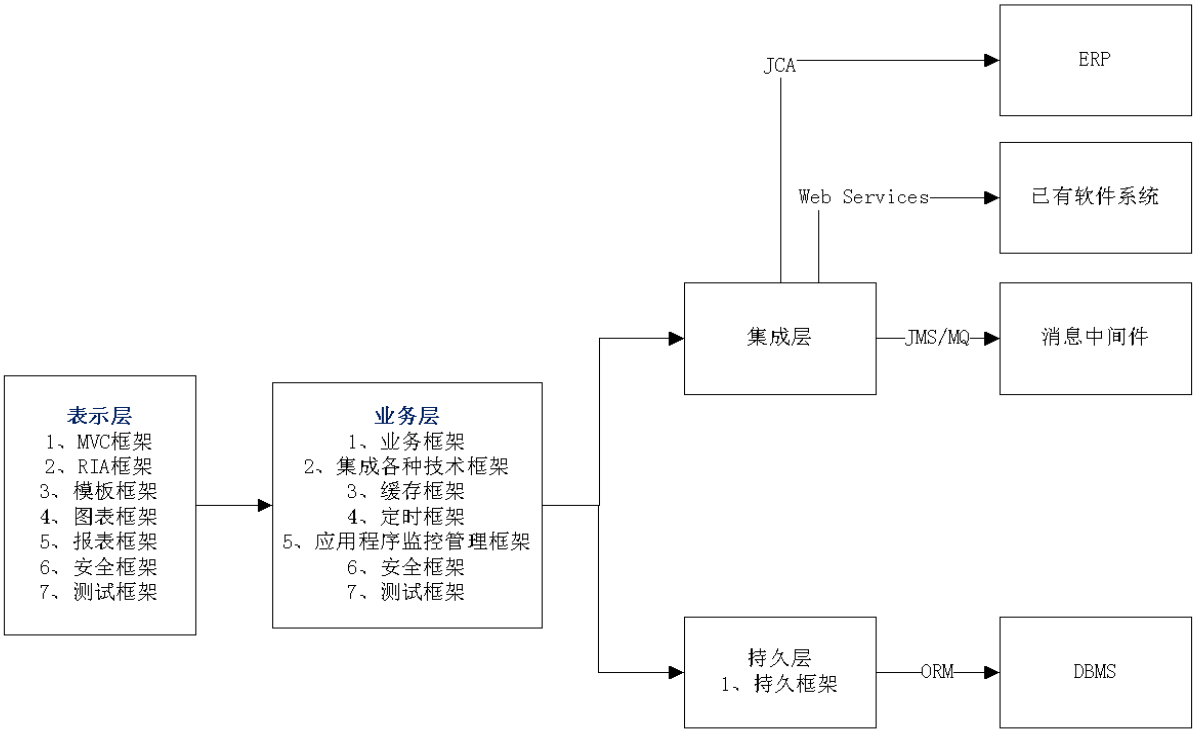
C、架构元素的变化分析
以核心架构元素展开变化分析，发现变化规律，重点要解决如何确保架构对变化的适应
例如：采用设计模式、OSGI(**Open Service Gateway Initiative**热插拔)、XML文件配置化(类之间的依赖基于抽象，依赖关系在配置文件中描述)、将变化信息移至DBMS（内存数据库）变化分析：
需求变更类型：UI变更，业务规则变更，业务领域变更，非功能性指标变更
如何消化变更：
解决方案方向：分层化、组件化、服务化、流程化

如何有效地做项目总结
1、需要对“需求变更管理系统”内容进行分析
2、需要对“Bug管理系统”内容进行分析
3、开展项目组内部会议

三、开发架构
1）、技术选型的策略
1、成熟：技术本身的功能性、可靠性，开发人员掌握的程度，学习成本
2、先进性
3、经济性

2）、技术选型的内容
1、选择技术路线：.NET，J2EE ,C++, LAMP(PHP)，Ruby on Rais，Delphi，PowerBuilder
2、编程语言技术选择
3、应用服务器选择
4、开发工具的选择
5、技术框架的选择
6、数据库的选择
7、支持开发的工具选择
列表：单元测试框架、重构工具、版本控制系统、Bug管理系统、持续集成服务器

重点：1、关键技术问题的攻关
2、多种手段验证技术方案（开发期质量、运行期质量）
开发架构产物包括：技术解决方案、技术规范（技术采用标准）



MVC框架：Java Swing / Struts2.X、Spring MVC3.X、JSF 2.X
RIA框架：Ajax、Flex、Java FX
图表框架：amcharts、JFreeChart
报表框架：JasperReports，BIRT，Crystal Reports，国产
安全框架：Spring Security
业务框架：Spring Framework、EJB、工作流
规则引擎：ILog、JBoss drools
缓存框架：ehCache、JBoss Cache、Open Terracotta(JVM级的开源群集框架)
定时框架：Quartz
监控：JMX
XML框架：StAX、XStream、dom4J
持久框架：Hibernate、iBatis、JPA
Web Services框架：CXF、Xfire

Microsoft .NET技术方案
1、Web层
ASP.NET，Sliverlight，ASP.NET MVC框架
2、业务层
WCF、WWF
3、持久层
Linq、Entity Framework
4、监控
WMI

C++代码内存泄露分析工具：IBM.Rational.PurifyPlus

UI方案选型案例

版本控制系统：ClearCase、SVN、TFS/VSS、Mercurial、Git
Bug管理系统：JIRA，Mantis，ClearQuest
单元测试框架：JUnit、TestNG、Nunit、Visual Studio MSTest、TFS、Google Test Framework
Gtest、GMock、Cmockery
持续集成服务器：Hudson、CruiseControl 、Microsoft Team Foundation Server
重构工具：开发工具已包含
将代码中注释生成帮助文档：Doxygen

单元测试覆盖率检查工具：Clover、Ncover

四、架构元素的管理设计
A、软件并行设计：
1、如何提升硬件的使用效率？多进程和多线程
2、关注优化进程和线程对资源使用：内存、CPU、磁盘
Java平台：java.util.concurrent包
参考书籍：《java并发编程实践》
.NET F# / Scala

3、分布式处理？：CORBA、COM+、EJB、Web Services、云计算（http://hadoop.apache.org/ ）

B、可靠运行设计
高可用性级别对应表

可用性级别	每年停机时间
99.999%	5分钟
99.99%	53分钟
99.9%	8.8小时
99.0%	87.6小时
95.0%	438小时

1、组件可靠运行设计
要点：1、有冗余　2、能恢复　3、能被隔离　4、能被监控
A、实现冗余的技术：双机热备份，软件Cluster（中间件服务器），
硬件Cluster（OS支持，磁盘阵列柜）
B、恢复：开发守护进程、线程，及时重启应用程序
C、隔离：设计上将不可靠的部分划分为独立的系统，将资源开销大的操作进行设计上的隔离（将操作运行的时间调整到系统空闲时间），当系统处理量到达上限时，停止新的请求的响应
D、对组件运行进行监控（SNMP、JMX、WMI技术）
监控内容：系统处理业务规模、系统处理时间、数据规模

采用线程或进程进行隔离
如何对组件运行失败的恢复
如何进行故障转移

C、可维护性设计
1、如何解决系统升级？（设计自动化升级程序，由升级服务器推送升级程序）
解决方案：用特定的技术平台实现自动化升级：Java（Java Web Start）,.NET（Smart Client）
2、如何解决系统扩展？（采用配置文件）
Java平台：JMX--MBean（热部署）、OSGi（热插拔）
Windows平台：DLL

D、安全性设计
1）数据有效性检查：语法（数据类型，取值范围）、语义（符合业务逻辑），在架构的各个层次要确保落实
通过ETL中的数据清洗活动，针对数据交换时采用专门数据校验模块进行有效性检查（XSD）
2）数据防窥：
数据加密：加密算法，证书加密
加密体系：对称密钥体系，非对称密钥体系
身份验证授权：统一身份验证（SSO单点登录），统一授权
基于操作日志的审核
安全产品：入侵检测

高性能网站架构演进 案例分析

一、架构演进的原因
A、业务的扩展
B、系统质量属性的升级

J2EE架构设计演变总结：
1、如何使系统架构更灵活地支持业务？
分层架构设计（增加新的应对业务变化的层次），SOA（ESB），业务功能组件化，使用工作流技术，基于配置的系统扩展（采用类似于IOC容器方式装配系统）
2、如何确保系统高可靠性？（可靠性就是指如何确保系统在碰到极端的情况下，应用可以快速恢复对外提供服务）
集群（避免单点失效）；开发守护进程监测JVM是否存在或挂起，视情况重启JAVA进程；检测端口是否存在；统一地异常管理，发生时迅速采用短消息通知；尝试切割不可靠的业务模块到另一个系统，避免整体不稳定
Web层对系统访问的用户数量进行限制，限制包括上传、下载的文件数量、尺寸；数据查询访问要分页，使用数据库连接池
3、如何在架构设计上体现性能？
使用硬件负载均衡器；前置ApacheServer；页面静态化；页面缓存；数据缓存（注意集群下需要使用分布式缓存）；多线程提升并发处理能力；使用JMS技术异步串行处理业务；减少或不使用Synchronized关键字；使用java.util.concurrent包的API实现多线程环境下的并发操作
控制Web Session中的内容的数量；

大数据量架构设计的优化
1、数据分批写入
2、优化写入的数据量（用编号/编码减少写入的数据量）
3、将多用户的并发写数据任务，改成串行的排队。FIFO
4、将数据处理方式改成在非工作时间的批操作。

Day3
一、培训目标
架构设计过程：物理架构、数据架构
架构设计案例分析
SOA架构
云计算

二、物理架构：服务器选型、部署
SRS中非功能性需求会对物理架构设计有要求
1、系统运行环境架构设计
服务器配置：（分析行业成熟的选择、处理业务量、业务访问密集程度等因素确定）
CPU：数量、主频
内存：数量
磁盘：硬盘->RAID->专用的磁盘陈列柜->SAN(高成本)->云存储（低成本）

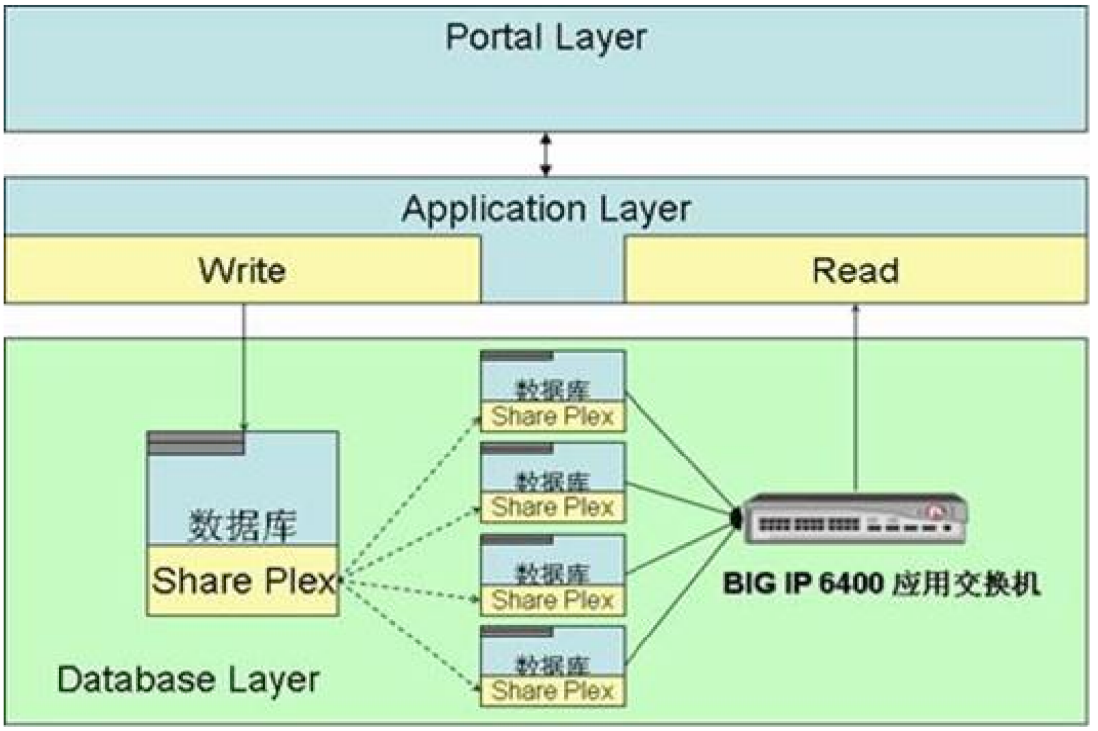
在IBM小型机上硬件经验：
我们的经验公式：
1．磁盘配置＝ SWAP＋当年交易笔数×（1+1.5+ 1.5 X 1.5）×每笔交易记录占用空间×（1.2历史加工数据）＋三年后客户数×每个客户占用空间
2．服务器内存＝1G+1G+（日峰值数×最大响应时间3秒×10m）/（N小时×60×60）
3．Cpu处理能力（TPMC）＝（4×并发数×60）/（50%× 峰值70%）
其它辅助硬件设备：网络设备（光纤交换机）、采集设备等
2、部署架构设计（网络拓扑）
注意：考虑网络带宽、网络安全的影响，
其它辅助硬件设备：网络设备（光纤交换机）、防火墙设备等

三、数据架构：满足数据需求
数据架构设计内容：
A、逻辑架构设计：E-R图
上层规划设计

1）规划整个系统运营中涉及的数据：从领域建模中的结果获取业务数据的分类
通过对SRS中业务分析：核心业务数据+支持业务数据
最终归纳出**数据分类**中初步的**数据项**
2）数据库表设计：范式原则
Refactoring Databases: Evolutionary Database Design

B、物理架构设计：高性能、高可靠性、高可用性、高可伸缩性
底层规划设计
IO：文件系统、数据块和数据页等
存储设备：RAID，磁盘陈列柜，NAS，SAN
数据库中设置：数据库分区
数据库设计：数据库分表、数据库业务分库，数据库读写分库
备份/恢复策略
内存：DBMS参数配置
CPU：DBMS参数配置

对于大访问量的网站，采用读写分离。比如ebay的读：写，比率是260：1。
采用读写分离有如下工具：
1、Oracle的logical standby
2、Quest公司的SharePlex
3、DSG公司的RealSync



二、SOA
SOA是一种软件架构
SOA的核心概念
SOA架构的特点：1、松散耦合 2、分离关注点（业务和技术）
SOA的引入分为四个阶段，成熟度模型
1、第一个阶段：现有业务系统或新系统中服务的发现、设计和实现（服务化）
2、第二个阶段：引入ESB，对系统提供的服务进行集中管理
3、第三个阶段：使用BPEL流程，以快捷的交付业务功能
4、第四个阶段：SOA治理

总线的优点：解耦、标准化、灵活地扩展

ESB的主要职能：服务的路由、协议的切换、内容的转换

将来SOA的集成主要应用四大领域
1、应用集成（服务集成）：面向应用系统的服务互操作（Web Services、IBM MQ、Active MQ）

2、界面集成：用统一的界面展示企业内多系统（Portal）
3、业务流程集成：使用BPEL技术将服务编排成流程（WPS，Oracle BPEL）
4、数据集成：面向海量数据异构数据库的汇集处理（数据交换：ETL产品）DataStage

开发SOA架构思路

1、设计整体系统总览图（分析待开发、待改造的业务系统的涵盖领域）
2、对IT系统进行系统内和系统间的服务的分析设计
思路：首先从系统间集成的要求来发现系统内的服务，然后进行分解成系统内的原子服务
3、根据业务要求进行设计制定流程,并指定流程中包含的活动和使用的服务。
4、定义使用外部服务，和进行新服务的开发
5、通过服务的组合和编排来推出新的流程
注：流程指的BPEL流程

基于SOA思想的技术架构

一、规范
SOA的规范：SCA、SDO、BPEL
流行的松散耦合技术规范：Web Service(WSDL、SOAP、UDDI)
Portal方面的规范：Portlet规范
二、技术框架
SCA、SDO的框架：apache-tuscany-sca，apache-tuscany-sdo

Web Service框架：Axis1/2;XFire;CXF;Spring WS
.NET平台：VS2005，需要安装Web Services Enhancements (WSE) 3.0 for Microsoft .NET
VS2008/VS2010，WCF（Windows Communication Foundation）
三、产品
ESB Server/Process Server/Portal Server/消息中间件
商业：
IBM系列：IBM WebSphere ESB Server/IBM Message Broker;IBM WebSphere Process Server;IBM WebSphere Portal Server；IBM WebSphere MQ
Oracle/BEA系列:Oracel Service Bus;Oracle BPEL Server;Oracle Portal Server

Microsoft系列：Microsoft BizTalk Server；Windows Workflow Foundation；SharePoint Portal Server；Windows Communication Foundation

开源系列：
ESB:JBOSS ESB,Mule；apache-servicemix
Process:JBOSS JBPM,ActiveBPEL
Portal:Apache JETSPEED...
消息中间件：ActiveMQ

云计算产品
云服务产品： Eucalyptus