Java企业应用系统框架的比较与选择 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/253/2021_2022_Java_E4_BC_81_E4_B8_9A_c97_253073.htm 引言 EJB的体系结构是J2EE的基础和核心，J2EE定义了整个标准的应用开发体系结构和一个部署环境，基于EJB的框架一度成为人们开发Java企业应用的首选。随着Java开源项目阵营的发展壮大， 一些基于POJOs(Plan Old Java Objects)的开源框架被越来越广泛地引入到Java企业应用的开发中来。根据复杂程度人们习惯把前者称为重量级框架，把后者称为轻量级框架。Java企业应用框架一般被划分为三个层次：表现层、业务逻辑组件层和持久层。本文主要对目前企业应用对应于这三个层次的两种类型的流行框架进行了细节比较，最后针对Java企业应用的系统框架选择提出作者的观点。 两种类型框架概述 1、基于EJB的重量级框架 由于 EJB容器能够很好的处理系统性能、事务机制、安全访问权限以及分布式运算等问题，基于EJB框架进行开发能保证企业应用平滑发展，而不是发展到一种规模就重新更换一套软件系统，且可以保证开发人员将大部份精力集中在业务逻辑的开发上。采用EJB框架开发的企业应用具有必须继承或依赖EJB容器的特点。EJB充分考虑到了顶级大型项目的需求，使用它几乎能解决企业级应用涉及到的所有问题，相应的基于EJB框架也是一个功能复杂的重量级框架。 J2EE1.4标准规定的EJB 2.1框架缺少设计且实现起来有些过于复杂。当前J2EE5.0的新规范提出的EJB 3.0的目标就是简化开发[1]，借鉴了一些基于POJO的思想，它相对于EJB2.1中两个重要的变化分别是：一是使用了Java5中的程序注释工具，注释取代

了过多的XML配置文件并且消除了严格组件模型需求；二是采用了基于Hibernate和TopLink思想的O/R Mapping模型。J2EE5.0的新规范中定义企业应用三个层次的标准实现为：表现层采用JSF（Java Server Face），JSF的开发流程的核心是事件驱动，组件和标签的封装程度非常高，很多典型应用已经不需要开发者去处理http。整个过程是通过IoC(依赖注入)[2]来实现的；业务组件层采用EJB3.0的Session Bean。EJB3.0允许开发者使用藕合松散的组件来开发应用。这些组件通过自己发布的商业接口来耦合，不必像EJB 2.1规范定义的那样一个Bean必须遵守的严格的组件模型，每一个EJB类必须从某一种抽象类中继承，并为容器提供了回调的钩子；持久层采用EJB3.0实体Bean持久化模型，吸收了Hibernate的一些思想采用O/R Mapping模式， EJBQL也有许多重要的改变。 2、基于POJOs的轻量级框架 在基于POJOs轻量级框架上开发的应用程序无需依赖于EJB容器可独立运行，对应于Java企业应用三个层次的轻量级框架技术分别都得到了一定的发展，这三个层次流行的框架如下： 目前比较流行的开源表现层框架主要有Struts和Tapestry。Tapestry与Struts应用框架不同的是，它是基于组件，而不是面向脚本语言（比如JSP和Velocity）的，组件是由一个定义文件(以XML的格式)、一个HTML模板、一个JAVA类构成的；业务组件层轻量级解决方案也不少，包括Spring、Hivemind等。但是目前使用最为广泛的还是Spring框架，Spring框架是一个基于IoC和AOP（面向方面编程）[3]的构架。采用IoC使得它可以很容易的实现bean的装配，提供了简洁的AOP并据此实现事务管理等，但是它不具备处理应用分布式的能力。Spring的核心要点是：支持不绑定到特

定J2EE服务的可重用业务和数据访问对象。这样的对象可以在不同J2EE环境（Web或EJB）、独立应用程序、测试环境之间重用；持久层框主要有Hibernate和各种JDO产品，以及iBATIS。Hibernate是一个开源的O/R Mapping框架，它对JDBC进行了非常轻量级的对象封装，可以应用在任何使用JDBC的场合，可以在应用EJB的J2EE框架中取代CMP，完成数据持久化的重任。iBATIS是一个简易的SQL Map工具，它是将手工编写的在xml配置文件中的SQL语句映射成Java对象。对应于三个层次的框架比较 1、表现层框架比较 MVC设计模式不再是某一种表现层框架的特点而是这几种框架的共性。Struts框架由于出现时间早，所以使用相对广泛，它的社区非常活跃，很容易找到很多现成的开源功能标签以供使用以及样例程序可供参考。但是它的组件在页面中显示的粗粒度，以及框架类的限制在很多情况下会表现得过于死板，给表示层的开发会带来一些额外的代码开销。JSF在很大程度上类似Struts，只是JSF的组件概念没有象Struts那样必须继承ActionForm的限制，JSF在事件粒度上要比Struts细腻。JSF有的另外一个优势就是其身后有Sun公司和其他的一些大公司的支持。Tapestry是一个完全组件的框架，Tapestry的组件可以被套嵌并包裹其它组件，因此可以组合形成一个更大的组件或逻辑页面。组件的行为模式为Web页面编程提供了很大的方便，事件处理也方便很多。所以，如果做一个对页面要求灵活度相当高的系统就可以考虑选用Tapestry。 表1 三种框架的表现层功能技术细节比较 框架 Struts Tapestry3.0 JSF View组件实现模式 标签库 组件，组件必须继承ActionForm 完全组件，分显式调用和隐式调用，组件必须继承BaseComponent 标

签库 组件，普通POJO无需继承 组件在View显示粒度 View页面只能显示与表单对应的ActionForm，配置中Action与ActionForm与 页面一般只能1:1:1关系。 可将组件嵌入页面任何一行，对使用组件数量无限制。 同Tapestry 页面跳转 使用标签库html:link中写明目标URL，URL名称需要对照struts_config.xml配置文件中的path命名，与组件Action耦合。 URL名称是目标的组件名称，不涉及URL和路径等操作，方便稳固。 类似Struts，也需要在配置文件中查找，与组件分离。 事件触发 通过表单提交submit激活，不能细化到表单里字段。 能够给于表单每个字段赋一个事件，事件组件必须实现PageListener接口 同Tapestry，事件组件必须实现ActionListener 2、业务组件层框架比较 EJB 2.1框架有些过于复杂了，有如下缺点：① EJB模型需要建立许多组件接口和实现许多不必要的回滚方法；②EJB的部署描述复杂而容易出错；③开发人员不能脱离EJB容器测试。对于以上缺点JCP (Java Community Process)制订的EJB3.0标准框架做了相应的改进，该框架为所有主要的J2EE厂商支持。EJB3.0和Spring两个框架结构都有一个共同核心设计理念：将中间件服务传递给耦合松散的POJOs。 EJB3.0框架与应用服务器高度整合，服务整合代码也包装在一个标准接口后面。EJB框架一方面有成熟的EJB容器支持，基于EJB框架的企业应用性能优良；另一方面EJB容器设计因为考虑了多方面的功能，所以在其内核上总是会显得臃肿，这也是一种重量表现。不需要的东西存在肯定会影响效率，EJB不能根据项目需求对EJB整体包括EJB容器进行可配置式的切割。 Spring框架处于应用服务器和服务库的上方，服务整合的代码属于框架，并暴露于应用开发者。

它与应用服务器整合的能力相对EJB3.0要弱。但是Spring框架模块的可分离配置体现了它优于EJB3.0的灵活性。 表2 EJB和Spring框架的具体细节比较 框架 EJB2/EJB3 Spring Framework 1.x 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com