关于组件的思考
摘自 UMLCHINA 论坛
模式小组-雷神整理

如何划分组件
目前都在提三层式的体系结构，或多层的。
例如 Windows DNA，强调表示对象、商业对象、数据对象我有个疑问是，如何划分组件？如何体现组件的重用？
譬如用户管理，
数据对象负责存取数据库中的表 user
商业对象调用数据对象，以和数据层隔开，不受数据库的影像
表示层当然就是负责界面了
那么这里的组件应该是什么？每个对象都是一个组件吗？
还是将它们合起来作为一个用户管理组件发布？
组件是可以发布给他人，供他人重复使用的。
可是一个基于数据库的应用怎能被重复使用？
难道在组件发布中还要说明数据库中的表名称、字段应该怎样设计吗？

回复：如何划分组件
在三层或多层架构中，核心思想并不是"组件如何划分"，而是系统的层次的划分。组件的划分可以根据开发人员的习惯或不同的应用任意划分，但层次的划分则体现了系统的逻辑结构。
组件，只是对象的一种封装。所以，看一个组件是否可以重用，关键是看它所封装的对象是否可以被重用。
大家都知道，面象对象的系统能更好地支持重用。例如 DNA 体系开发，正如我们在书上可以看到的可以划分为：表示层、业务对象层、数据访问层。如果在一个面向对象系统中，业务对象层一般都是封装有状态业务对象，而数据访问层则会包含支持业务对象层有状态对象的无状态对象；在这两层之间，一般的情况会架构一个支持数据持久化和事务处理的框架。表示层是可选的。
对于重用而言，在以上的描述中，框架肯定是支持重用的。按一般的说法，无状态对象比有状态对象能更好地支持重用，但在不同的角度或不同的抽象层次来看，都可能会有不同的理解。
回复：如何划分组件
我觉得你的问题是任何入门的人都会碰到的。可以分为下面几个部分考虑。
1：组件的粒度大小。组件实际上只是类的容器。按照常规的处理办法，功能上聚集在一起的类组合成一个组件。同时考虑要一些其他问题，如组件对系统运行环境的要求。例如，在 windows 平台上，安全机制是以组件包为界的。当然也有 callinrole 这样的接口。
2.组件的状态。在类的设计中，他们是属于数据持久性的问题。发布是当然也要发布它们。以用户管理来说，最好的例子是 windows2000 下的 ADSI。他是使用文件实现的。所以你的包应该有至少两个：一个是商务层的包，一个是数据层的包。数据层用于实现数据持久。同时也祝贺你，你这样做的话几乎可以支持任何后端数据存储。三层结构的好处显而易见。
回复：如何划分组件
happylo 说无状态的对象比有状态的对象能更好的支持重用
而且一般是指数据对象层使用无状态的对象
那么也就是说数据层的包在发布时，是不包括这些状态信息的
例如某个数据访问层，提供对数据库中表 user 的存取，提供的方法无非 add,delete,query 等等，当然还有参数。方法的实现中，必然含有具体的 SQL 语句，也就是说牵涉到表名、字段名，但在对象的接口中这些数据是无法体现的呀。但是对于另外一个要重用此对象的系统来说，在进行数据库设计时，user 的表名和字段名是自己设计的，此时，怎么能重用那个数据访问对象呢？
如何划分组件--设计与重用
我觉得你的问题可以这样来考虑。
1.你的数据层的对象会独立于商务层使用吗？在 User 表的例子中，回答是不会。所以你的数据层是且仅是由你的商务层创建并使用。你要给其他人暴露的接口是商务层的接口，所以你的第一个担心不必要。通常来说，商务层体现了你的组件提供的功能。考虑实际使用的问题，你的商务层组件可以提供一个初始化的接口，调用数据层创建相应

的库及表。
2.对于第二个问题。想象，当你将 window 类的 title 属性换成 Head 的时候，你仍然想要 window 类正确被你重用吗？重用你的组件的人如果想要扩充功能，可以利用组件的聚合和包容的能力。

如何纵向的和横向的划分系统？是否存在个模式

对于三层架构、多层架构，大家都知道概念。但一旦到具体应用时，一般的设计人员对于如何纵向的分层 、横向的划分模块与组件，可能会感觉无处下手、没有啥规律可以遵循。比如对于常见的 MIS 系统，往往对于数据库依赖性较强，一个经常困扰设计人员的问题是业务逻辑的划分。举例来说，对于单据的处理。一般是在界面层完成单据的录入，那录入数据的合法性校验是由界面层完成呢，还是由业务逻辑层完成呢，或者独立出一个数据校验层？另外对于单据数据的处理，在 c/s 结构中一般用存储过程实现，在多层架构下，对于影响系统性能的数据库操作肯定还是用存储过程实现，但对于一般的业务处理是用存储过程实现呢，还是在业务对象中直接调用 SQL 语句？用存储过程效率高但不灵活、不能继承、也不能调用其他的业务对象，移植也不方便。能否有个规律，可以判断业务逻辑在业务对象和存储过程中如何划分？ 此文全当抛砖引玉，希望大家能讨论出一个系统划分模式。

回复：如何纵向的和横向的划分系统？是否存在个模式

划分的原则：闭合和松耦合相结合，例如你上边的例子，对于单据，如果单据本身提供的苏举足以用来做审核了，那么就在表示层完成，如果需要数据层的支持，那么就要放在业务逻辑层来做了。
“但对于一般的业务处理是用存储过程实现呢，还是在业务对象中直接调用 SQL 语句？用存储过程效率高但不灵活、不能继承、也不能调用其他的业务对象，移植也不方便“，不明白你在说什么，其实用什么都差不多（如果时多层结构的华），主要要看你的系统资源了。直接 SQL 调用相对来说对环境要求的低，procedure 要昂贵一些。

回复：如何纵向的和横向的划分系统？是否存在个模式

n-tier 是 sun 主推的系统体系，目的在于可以增加系统的灵活性，并可以支持大规模并行开发。层次之间定义接口，但层次多带来的是工作量的增加，一般只笼统的分为表示层，业务逻辑层，数据层（三层结构）。对于模块和组件就需要实事求是的分析需求，然后抽象而定了。就单据的处理而言，单据应是个对象，单据包含数据和检验数据的方法，如果你认为单据对象是表示层，那么校验就在表示层处理。对于存取过程，如果简单（对一两张表操作），可能应属于数据层，否则应包含了业务逻辑。个人觉得系统的划分不存在模式之说，如果非要找个模式的话，“强内聚，松耦合”
业务与数据库、计算机系统之间隔着太多的实质性内容。根据一边的皮毛就判断另一边的皮毛应当是什么，未免把 OO 的内容看得一钱不值了。仔细分析之后，根据业务的独特特点才能分模块。在好先容忍“模块”概念的困扰，好好分析分析业务系统内部的复杂关系吧。如果 3、4 个月可以完成一个程序，根本没必要计较分层的问题。往往是分析问题时过分简化了对象间的关系，最后在用户的威逼之下系统不断增加实质的内容，分层自然就变成“多层”了。（比如你可能把所有发送电子邮件报表的部分抽象出来请别人来做）

求助：如何设计一个税务项目的三层结构开发方案

本人最近接到 boss 的一个棘手任务，由于本人在这方面的无知，恳请大家帮助。
任务描述：设计一套三层结构体系（c-s-s 和 b-s），应用于税务征管软件。内容包括：
1。应用服务器的选择；
2。客户端开发工具的选择；
3。中间层组件开发工具的选择；
4。将什么功能摆在中间层实现；
5。系统的整体设计方案。
项目背景和要求：本项目是一个税务征管的解决方案，项目要求如下：
1。在一个省内实现税务的自动化征管，企业是客户端用户，中间事务层，和数据库服务器都集中在省会的税务局内；
2。要求企业既能用网页方式报税，也能用客户端报税，但必须通过中间层，不能直接连接数据库；
3。税务局现有一台 IBM AIX 的什么型号的小型机，跑的是自带的 AIX Unix,想尽量利用起来，作为应用服务器（如果实在没法做，可以建议他们买台跑 Win 的服务器）；
4。数据库税务局已经购买了 Oracle 8i,不能更改。

公司的开发能力状况：
1。**以前积累了丰富的税务软件经验，对于业务熟悉精通；**
2。**以前开发的税务软件均采用** c-s **结构或** 2.5 **层结构，主要的开发工具为** pb7,vb6**；**
3。**个别人有** java **的开发经验，局限于** JSP**；**
4。**公司上层的开发思想是走低成本的路子，且尽量靠近目前程序员所熟悉的开发工具。**
情况描述如上，希望大家多多提出建议，谢谢！！！
三层就三层，不就是 MVC 吗
model（**数据**）
controller（**商业和控制**）
view（GUI）
三层结构基本包含：
数据连接与处理层，可以实现的方法是 EJB **或** JDO **的基本** JAVABEAN
商业层，处理商业的中间层，如 DAO **和** OperatorDB **基层和基于上层的** Extense
控制是连接 GUI **与商业的通讯，一般试验** Main Controller **的方式**
GUI **层，**GUI **包括显示逻辑与服务**
显示商业 TagLib **来简化** GUI **开发，使用** TagLib **引擎来处理具体的商业到** GUI **的通讯**

如果采用 C-S-S 结构
1。**应用服务器的选择，就用那台** IBM **机器吧**
2。**客户端开发工具的选择，**c,java **都可以**
3。**中间层组件开发工具的选择，中间件可以使用** Weblogic
4. **业务逻辑处于中间层**

如果采用 b/s 结构则更加方便了
数据层：使用他们的 Orcle **数据库就可以了**
中间层：使用 IBM **的** Websphere,**安装在** aix **系统（现有机器）上就可以了**
表示层（客户端）：网页，使用 JSP
开发工具：IBM Visual Age, Jbuilder, **文本编辑器都可以**
若要整体方案，我们合作吧，我们有资深多层体系系统的设计背景，你的系统其实很简单。

我一直用 java **作多层企业级应用开发。我给你提供一个采用 j2ee 的方案，仅供参考。**
j2ee **是一个健壮、开放的平台，它对企业级应用提供了一个完整的解决方案，**
1，**在客户端，你可以采用** web **型客户端，如** jsp/servlet/applet **等；或者桌面型客户端** java application**。在这两者中的选择取决于你的实际业务需要，一个要考虑的方面是，你们的业务是否很密集，如果是，那么** web **页面相比** application **响应较慢的缺点可能会引起用户的抱怨，但是** web **客户端的** 0 **维护性（近似的讲。如果客户连浏览器也不会装，那就没办法了）我想应该值得你考虑。我们一个电信项目出于客户端响应的考虑采用的就是** java application**。但是用** application **你可能需要投入学习** swing **组件的成本。还有，一个"胖"的客户端，尤其是** java application **要求对客户端机器有较高的配置。**
2，**中间件。利用你现有硬件，既然是** AIX**，那就用** websphare **或者** weblogic**，都很贵。你使用** oracle8i**，那么** OAS **也可以考虑，不过我没用过。无论哪一种中间件，只要他支持** ejb2 **那么对于你的开发来讲没有多大区别，不同的只是发布上的不同。还有一些免费的** j2ee **服务器，如果信任它，就用它。**
另外，好好的 AIX **为什么要改成** windows **？有点像把一匹宝马当驴子用来拉磨的感觉。**
3，**数据库没什么好说的，采用中间件，数据库的功能被弱化了。对了，**oracle **新推出了它的应用服务器,**oracle9iAP,**埃里森捧着它到处吹呢。你们使用** oracle**，说不定** oracle9iAP **和** oracle **数据库结合比** websphare/weblogic **要强大得多呢。**