# 包工头模式

**包工头 – 工人模式：**

包工头工人模式我是从一本书上看到的《大规模分布式系统架构与设计实战》。“包工头-工人”模式会使用到Anycmd的User模块中去。AC是一个隐式的项目。一个项目只有一个包工头，一个包工头可以雇佣很多工人，一个工人在一个项目下只受雇于一个包工头。每一个工人都是自己的包工头。

这是军队管理的组织结构。

包工头可以把项目拆分成很多子项目，包工头可以把新的子项目承包给新的包工头，每一个工人都可以承包项目，每一个工人都有机会成为包工头，承包项目的工人立即成为该项目的包工头。

不跨越，不交叉。

项目一直往下包，最终的叶子是任务。任务由工人完成，工人完成任务后由该任务所属的项目的包工头汇总，包工头再向自己的上级包工头上报，一直汇总到最初的包工头。每一个包工头负责评估自己的工人们承包的项目是否完成和为自己雇佣的工人打分等。包工头领取酬劳，包工头为自己雇佣的工人分配酬劳。包工头承包的项目完成后包工头和工人们的关系就可以解散了，他们等待在新的项目中被新的包工头雇佣。

无论灵活的自主访问控制还是严格的强制访问控制都可以通过包工头 - 工人模式完成。

包工头工人模式实际上就是组织模式。这种模式到处存在，创造出了巨大 社会价值。组织结构模型和注释如下：



**包工头 – 手工仓库 – 工人模式：**

Anycmd也会支持手工仓库模式。手工仓库是围绕着User而建立起的资源聚合。概念难懂，如果我们这样说可能会更容易理解：手工仓库的实现实际上就是建立一个有UserID（主体）、ObjectType（客体类型）、ObjectID（客体标识）、DynamicExpression这样几个重要字段的元素，并维护出一个有这种元素组成的集合。

手工仓库控制了用户可以看见的数据集范围。每一个用户都只能访问分配给他的手工仓库中的数据。包工头负责往工人的手工仓库中放置资源（放置的是ObjectType和InstanceID这样的引用，需要系统的配合来将工人的手工仓库中的资源记录正确的展示，用户的界面应该是根据由当前用户的手工仓库中的记录和系统配置自动绘制出来的）。

**包工头 – 职介所 – 手工仓库 – 工人模式：**

如需详细了解请猛击[这里](http://book.51cto.com/art/201403/432471.htm)。应用不同，道理相同。  
anycmd

Anycmd是一个.net平台的完全开源的，完整支持RBAC的（包括核心RBAC、通用角色层次RBAC、静态职责分离RBAC和动态责任分离RBAC），将会支持xacml的通用的权限框架、中间件、解决方案。完整的RBAC规范所定义的能力只是anycmd所提供的能力集的一个子集。 如果您感兴趣的话现在可以先观察Anycmd的源码，期待您为Anycmd提供帮助确保她走在正确的道路上。

### 框架、中间件、解决方案是它的三种使用模式：

**框架模式：**引用一两个必要的dll或者相应的源码，它跟您的应用系统运行在一起，您有能力完全控制anycmd，您需要自己提供UI层，但anycmd自带的UI层也是可用的。通过面向anycmd遍布各处的扩展点编程使用者有机会有能力实现自己个性化的需求；

**中间件模式：**引用一两个必要的dll和一些资源文件，它可以跟您的应用系统运行在一起，它提供UI层但您也可以删除并自主提供，它按照最佳实践提供默认配置，您可以通过调整配置比如自定义插件来满足或接近满足您的需求。可以把anycmd中间件看作是一个独立的系统，只不过它可以和你的应用系统运行在同一个进程、同一个或不同的AppDomain。

**解决方案模式：**提供一整套AC最佳实践、方法论，使用者有走向最佳实践的意愿。  
如何使用

anycmd不仅提供了来自RBAC国际规范文档的IRBACService接口<http://git.oschina.net/anycmd/anycmd/blob/master/Framework/Anycmd/IRBACService.cs> 还提供了一整套稳定的、功能完备的、风格一致的、流畅的api。框架使用起来非常简单，限定在AC领域内，基本会做到在权限方面的每一个需求都刚好有一个流畅的风格一致的api。编程的时候只需要通过一套风格一致的流畅的api告诉框架我们希望做什么，然后框架就去做了。但是如果能够明白访问控制系统做事情的逻辑的话会更容易使用那些api。 事实上anycmd很容易地就完整实现了对IRBACService的支持，因为对IRBACService的实现不需要书写专门的逻辑，因为anycmd的api是比RBAC所定义的能力集更大的，只需直接委托给anycmd的api就完整实现了IRBACService。

### 大方向

* 时机成熟后anycmd采用社区开发模式，接受社区提交的代码，不必担心anycmd会有天没人维护。
* xacml标准非常棒，但有点学术，很有难度。xacml的用户是程序员和专业的安全管理员而不是普通的用户，所以xacml的开发优先级比较靠后，支持xacml的时候希望借助社区的力量。