通用权限管理系统

系统需求分析

作者				
日期				
修订				
备注				

目录

1、引言

1.1、概述

用户权限管理系统一直以来都是应用系统不可缺少的一个部分，每个员工，各个职位都有着自己的工作和责任，淡然也有着自己的工作权限范围。若每个应用系统单独对系统的权限进行设计，来满足不同系统用户的需求，将会浪费很多时间，所以来设计一个统一用户及权限管理系统是非常有意义的。

本系统旨在对应用系统的所有用户信息进行管理，为不同应用系统地用户分配属性和权限，并为应用系统提供接口以便其进行调用。

1.2、设计目标

系统的设计目标包括如下三点：

（1）对应用系统的所有资源进行权限控制，比如应用系统的功能菜单、各个界面的按钮控件等进行权限的操控；

（2）完善用户、角色、组织、资源、操作的管理功能，其中的组织管理模块只提供组织视图，不参与权限的控制管理。

（3）开发人员开发新的系统功能，通过资源和角色模块进行操作管理。使用系统管理员身份登陆，直接将访问路径作对角色资源授权给操作，实现资源访问控制管理。

1.2.1、总目标

本系统提供一个调用简单、可复用性高、满足一般需求的权限管理模块，并为需要对权限管理的系统节省开发本。

1.2.2、性能目标

1、要求系统可适用于不同操作平台。

2、要求系统的可维护性和实用性强。

3、要求系统有一定的检错能力。

4、要求系统支持多用户同时操作，但管理者与用户不能同时操作。

1.2.3、功能目标

本系统的设计目标是对应用系统的所有资源进行权限控制，比如应用系统的功能菜单、各个界面的按钮控件等进行权限的操控 。

2、系统结构

2.1 功能架构

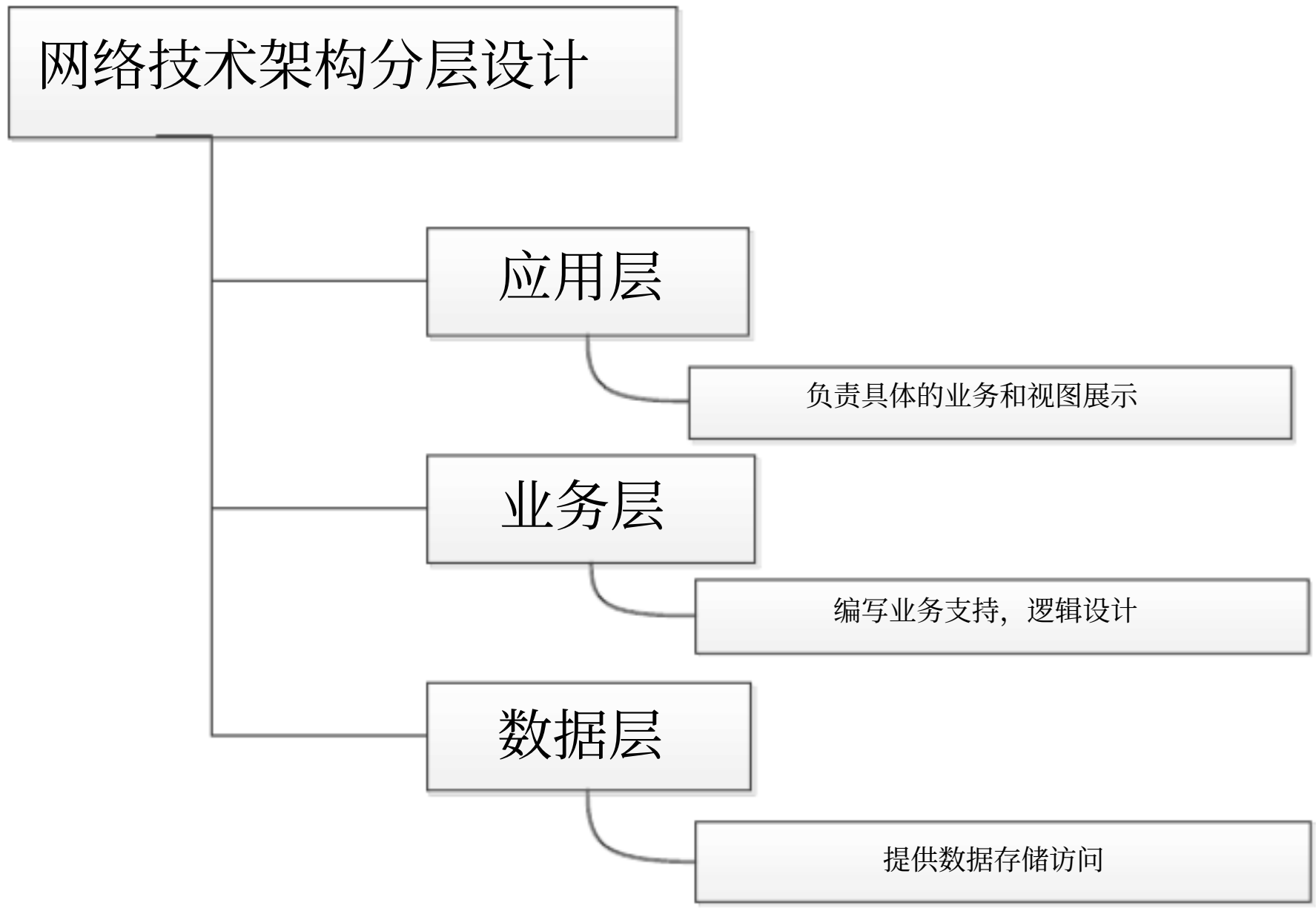
对于一个大的业务系统来说，如果要求管理员为其下员工逐一分配系统操作权限的话，是件耗时且不够方便的事情。 所以， 系统中就提出了对属性进行操作的概念， 为权限一致的人员分配同一属性，然后对该属性进行权限分配。

用户权限管理系统应该可以加入到任何带有权限管理功能的系统中。 就像是组件一样的可以被不断的重用，而不是每开发一套管理系统，就要针对权限管理部分进行重新开发。

传统业务系统中， 存在着两种权限管理， 其一是功能权限的管理， 而另外一种则是资源权限的管理，在不同系统之间，功能权限是可以重用的，而资源权限则不能。

2.2、技术架构

本系统是架构是一个三层架构，即浏览器和服务器结构。采用 Java 语言开发，封装对后台数据操纵的细节，并提供安全调用接口。 WEB 应用程序通过接口访问系统服务，执行用户操作并返回结果。下图为技术架构分析设计图：


网络技术架构分层设计
应用层
负责具体的业务和视图展示
业务层
编写业务支持，逻辑设计
数据层
提供数据存储访问

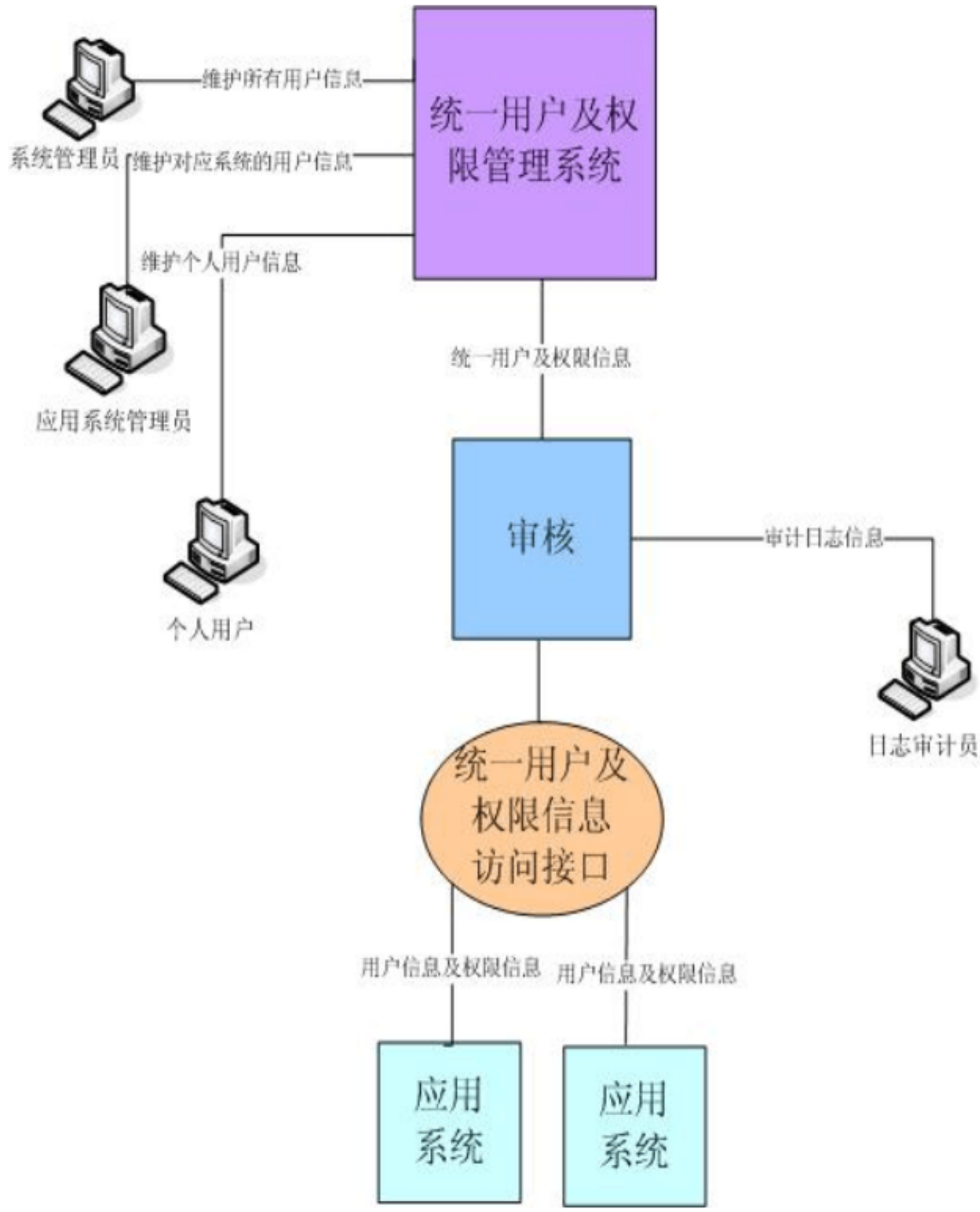
2.3、系统布局


控制区
功能导航区
工作区

3、系统功能

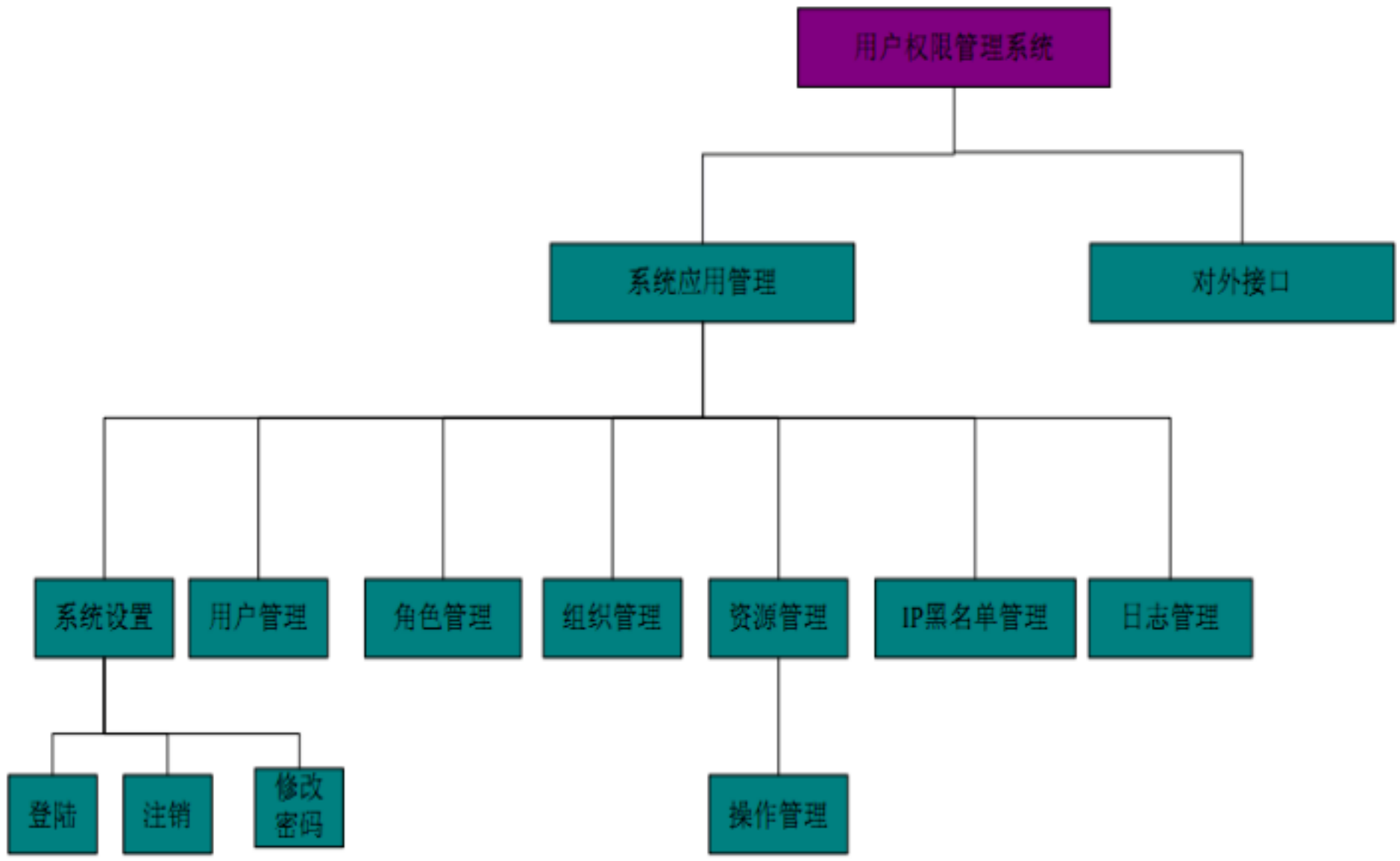
3.1 、功能概述

经过授权的用户可以正常合法的使用已授权功能，而对那些未经授权的非法用户无法登录系统。 系统管理员可以维护所有用户信息，普通管理员可以维护对应系统地用户信息，个人用户可以维护个人用户信息，并且可以为其他应用系统提供接口。如图：



3.2、系统功能模块分析

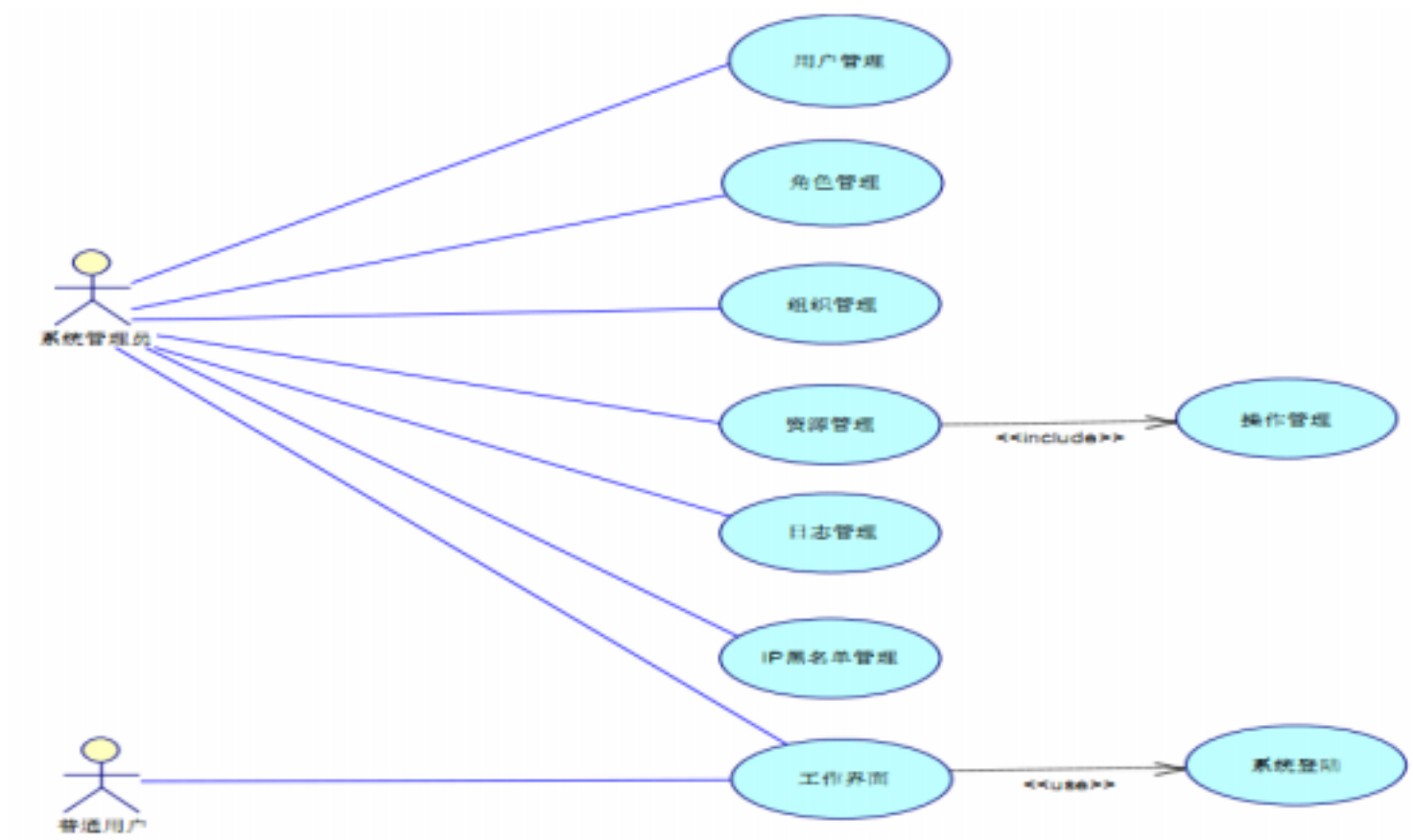
根据系统用例来划分功能模块，实现系统的应用管理以及对外数据接口，包括系统设置、用户管理、角色管理、组织管理、资源管理、日志管理以及 IP 黑名单管理。如图：

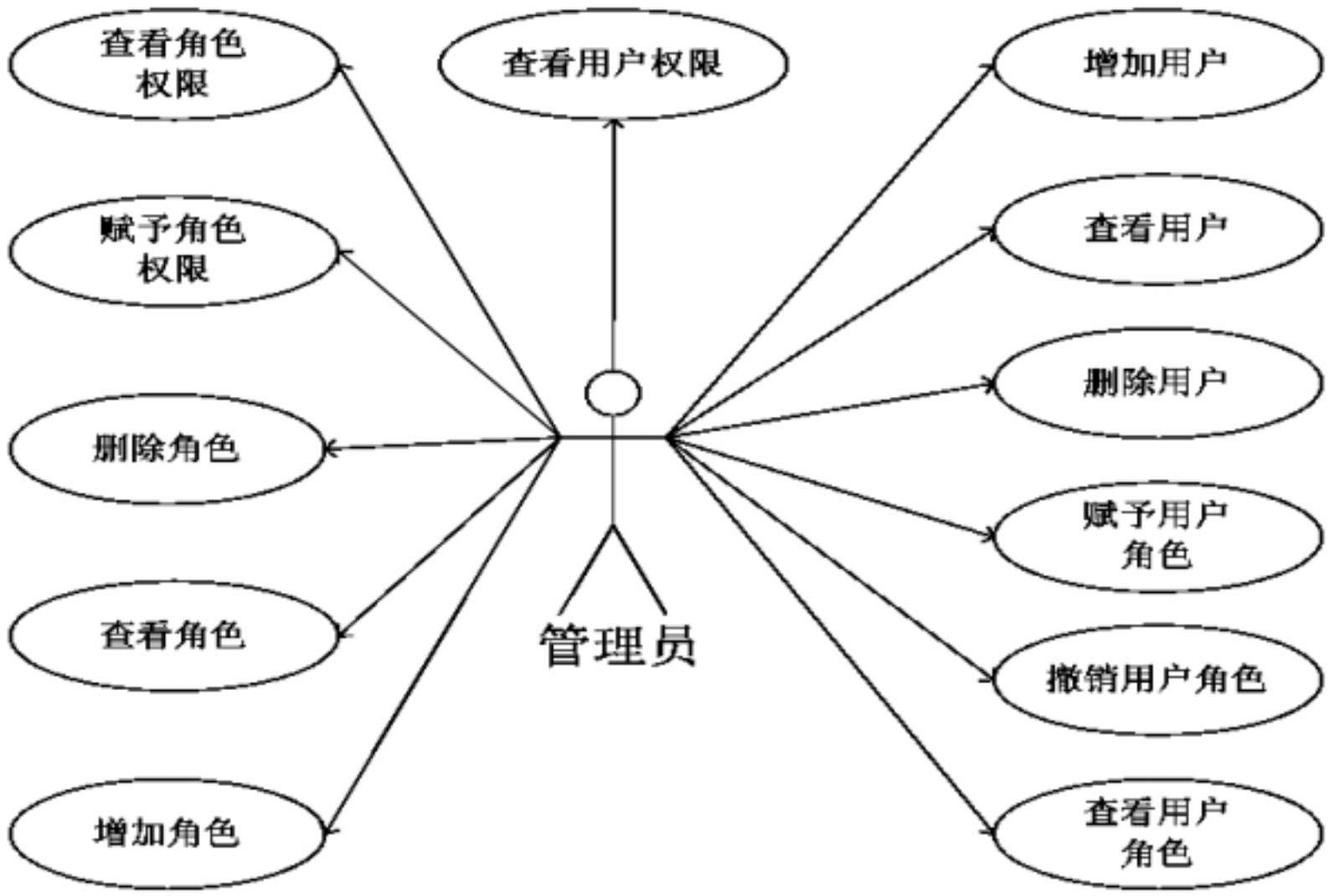


功能模块图

3.3、主要功能用例模型

系统业务用例图：





管理员的用例关系图

3.4、功能分析说明

3.4.1、系统用户

系统管理员：具有系统最高级别的权限，实行信息的全局管理与数据维护工作。

普通用户：由系统管理员分配权限，在角色权限范围内进行访问与操作。

3.4.2 、系统登陆

判断用户的 IP 来源是否在黑名单之列，对系统进行第一道防火墙保护。

对用户名和密码进行校验登陆。如果帐号和密码相匹配，则直接进入用户工作界面；否则，提示用户“ 用户名或密码不正确，请重新输入 ” ，窗口跳转回到用户登陆窗口。

3.4.3、工作界面

系统根据用户的权限对工作窗口进行初始化，不同角色的用户具有对应的工作窗口界面。

3.4.4、用户管理

系统管理员完成用户信息的录入、 维护以及用户授权工作， 并给用户指定组织机构。 系统应具备根据部门编号，用户编号，用户姓名来检索数据的功能。

3.4.5、角色管理

角色是一组用户的集合， 具有指定的权限完成特定的资源访问与操作行为。 为对有相似权限的用户进行分类管理，定义了系统管理员、管理员、用户、访客等角色。 角色具有上下级关系， 系统管理员通过角色授权分配权限资源， 那么， 下级角色的权限范围只能在上级权限范围实行进行授权操作。 角色管理包括角色信息录入、 信息维护、 将角色授权给用户、查看角色用户列表。

3.4.6、组织管理

与企业的部门或者机构对应，用于实现对用户的分组归类管理。组织具有上下级关系，可以实现无限级的子节操作， 管理范围包括组织信息录入、 组织信息维护、 察看组织员工等操作。

3.4.7、资源管理

资源权限是系统对用户访问的资源的路径 （包括图片、附件、 页面等）显示和访问进行控制。 资源具有上下级关系， 为了方便界面的渲染与加载， 资源的父子层次结构最好不超过 3 层。

3.4.8、操作管理

操作是资源访问控制相关的按钮控件或者操作，用于对资源权限进行更细粒度的管理。

4、平台安全性需求

4.1 程序设计安全性

程序设计的安全性 ,针对现在大多系统的分布式结构 ,因为同时要面向不同地理位置，不同网络地址，不同级别，不同权限的用户提供服务，稍不留神就可能产生潜在的安全隐患。

如下是最常见的由设计不当产生的安全漏洞分类：

1、输入验证漏洞：嵌入到查询字符串、表单字段、恶意字符串的攻击。这些攻击包括命令执行、跨站点脚本注入和缓冲区溢出攻击。

2、身份验证漏洞：标识欺骗、密码破解、特权提升和未经授权的访问。

3、授权漏洞：非法用户访问保密数据或受限数据、篡改数据以及执行未经授权的操作。

4、敏感数据保护漏洞：泄露保密信息以及篡改数据。

5、日志记录漏洞：不能发现入侵迹象、不能验证用户操作，以及在诊断问题时出现困难。

对于以上的漏洞，可用的防范措施有：

1、针对输入验证漏洞，在后台代码中必须验证输入信息安全后，才能向服务层提交由用户输入产生的操作。

2、针对身份验证漏洞，程序设计中，用户身份信息必须由服务器内部的会话系统提供，避免通过表单提交和页面参数的形式获取用户身份且要有登录验证码。

3、针对授权漏洞，在访问保密数据或受限数据时，一定要根据用户身份和相应的权限配置来判断操作是否允许。

4、针对敏感数据漏洞， 在储存敏感数据时， 一定要采用合适的加密算法来对数据进行加密。

5、针对日志记录漏洞，程序设计中，对改变系统状态的操作，一定要记录下尽可能详细的操作信息，以便操作记录可溯源。

4.2 程序部署及操作系统安全性

就程序部署及操作系统安全性而言，可用以下的防范措施：

1、无论部署于何种操作系统，需要保证操作系统在部署前，安装了全部的安全补丁， 关闭了所有不需要的系统服务，只对外开放必须的端口

2、定期查看所部署服务器系统安全通告，及时安装安全补丁。

3、定期检查系统日志，对可疑操作进行分析汇报。

4、应用服务器程序在服务器中文件系统中的目录结构位置应该尽量清晰。目录命名需要尽可能的有意义。

5、应用服务器程序不能以具有系统管理员权限的操作系统用户运行。最好能建立专门的操作系统用户来运行应用服务器

4.3、 数据库安全性

就数据库安全性而言，可用以下的防范措施：

1、数据库监听地址要有限制，只对需要访问的网络地址进行监听。

2、制定数据库备份制度，定期备份库中的数据。

3、数据库操作授权限制。

4.4、网络安全性

就网络安全性而言 ,可用以下的防范措施：

1、选用企业级防火墙。

2、根据具体网络环境，制定尽可能周密的防火墙规则。

3、需要在外网中传输的数据，应选用合适的加密算法进行加密。

4.5、 物理安全性

就物理安全性而言，可用以下的防范措施：

1、服务器应部署于专业的数据机房，做好机房管理工作。

2、对于支持热插拔的各种接口，需要在部署前在系统 BIOS中关闭。服务器在运行过程中，应该做好各种防护措施。

5、性能

软件性能是软件的一种非功能特性软件的性能是软件的一种非功能特性， 它关注的不是

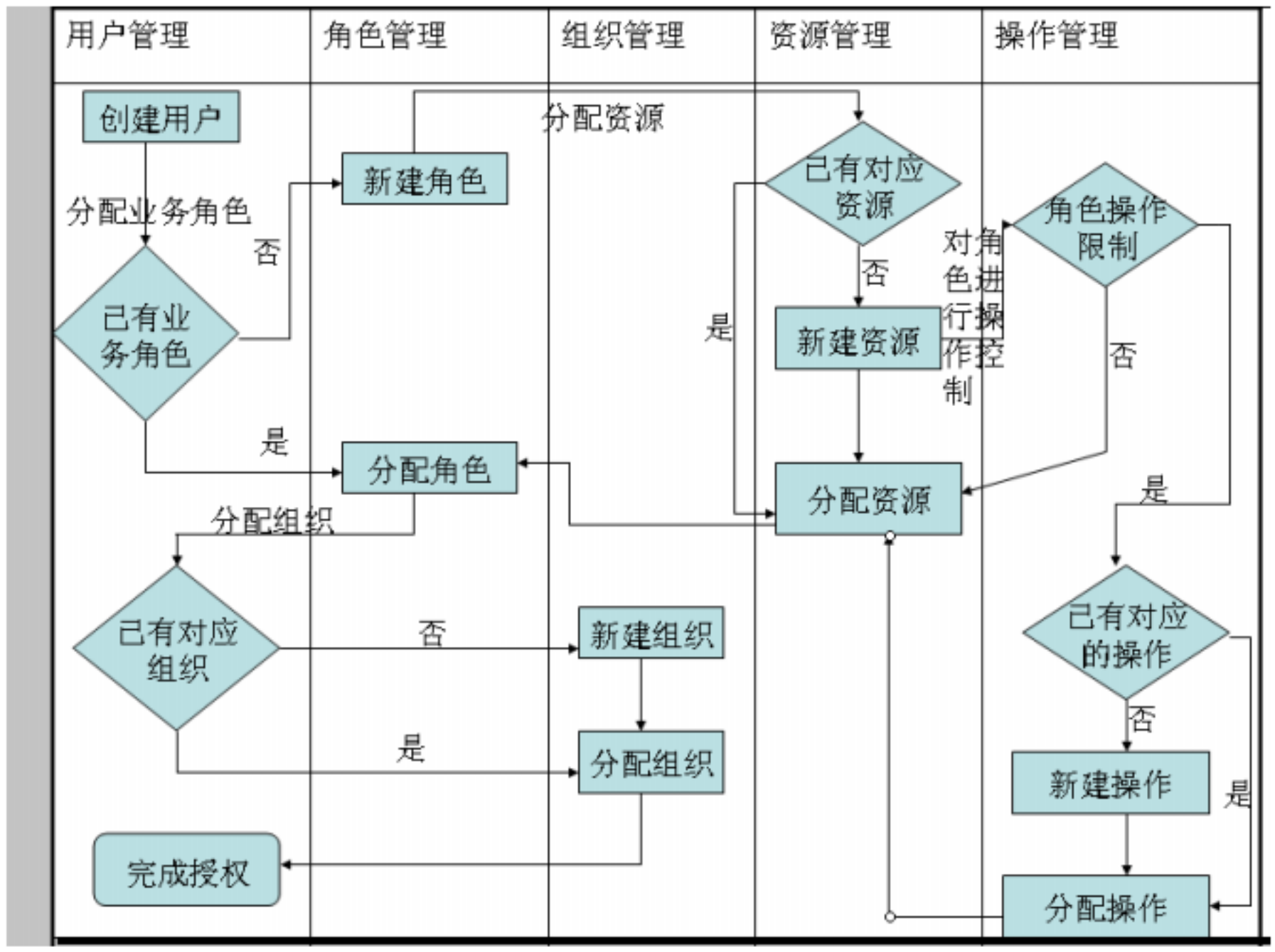
软件是否能够完成特定的功能，　　而是在完成该功能时展示出来的及时性。　　由于感受软件性能的主体是人，　不同的人对于同样的软件有不同的主观感受。　　而且不同的人对于软件关心的视角也不同。所以就需要一定的软件性能指标和软件性能视角来统一判断软件性能的好坏。

（1）该用户权限管理系统采用模块化系统，便于管理和维护。

（2）该用户权限管理系统可以实现多用户同时操作，　　但不支持管理者与用户同时操作。可以减少多个用户或管理人员之间因同时操作而产生的错误与冲突。

（3）要求系统的可维护性和实用性强。保证了该系统能最大程度满足用户的需求，并且使用方便，在出现问题时也便于维护和修改。

附：



用户授权流程图