# 服务端技术设计文档

## 开发环境

服务端开发，主要在windows下使用VS2008开发。发布执行环境为centos6.5。服务器为跨平台结构。

## 服务端工程结构

如图所示。



服务端整个工程可以分为两大部分。跨平台及网络通信模块与游戏功能模块。

## 跨平台及网络通信

### sdu

sdu跨平台接口，提供了整个服务器跨平台API接口的封装、常用系统性功能接口、通用定义等。

* 如，为了明确变量的定义，整数的定义必须使用

int8 uint8 int16 uint16 int32 uint32

等表示法。

* 如，为了跨平台支持，多线程的使用必须使用 CSDThread 类
* 如，便于时间相关的处理，提供了CSDDateTime类

### net\_manager

跨平台的网络通信模块。基于sdu。

该模块提供了window和linux两个平台上的服务器通信功能。window下使用Select。linux 下使用 epoll 实现。

具体设计细节可参考 <net_manager.vsd>

### transfer

协议层。提供了基于结构体的协议传输功能。

该模块定义了协议包结构，协议定义。为逻辑业务层提供基于结构体的信息通信机制。该模块提供了两个接口

TransferClientProxy 客户端协议层接口

TransferServerProxy 服务端协议层接口

逻辑层只需要调用此接口即可，无需关心网络传输的具体实现。

参考图 <Transfer传输层架构.vsd>

参考文档 <协议开发文档.doc>

## 服务器架构

游戏服务器，为三层架构。

gateserver负责与客户端通信，做协议转发。可分布式。

gameserver负责游戏业务逻辑。可分布式。

centerserver负责数据库访问，及部分中心控制功能。



服务器的正式线上部署结构



开发时，为了便于调试，可以只开启一个gateserver，一个gameserver，一个centerserver。



并且所有服务都支持断线重连，所以开发调试时。无需频繁开启关闭所以服务器。

## 游戏功能模块

### commonlib

为各游戏模块提供通用的功能，如一些数学算法，某些XML的解析、数据库访问基础功能，日志格式化等。

如果某些功能或函数在多个工程中都需要使用，可以放到此工程中。

### centerserver

中心服务器，与数据库连接。提供所有与数据库交互的操作(也有称DBServer)。另外一些不能采用分布式实现的功能也会放到centerserver中，如组队，公会，跳副本逻辑等。

### gateserver

也称前端服务器。与客户端长连接，负责客户端与GameServer之间的协议转发。既可以用来做网络的负载平衡；同时也提供与客户端协议层的加密、校验、压缩等功能。

### timeserver

主要用来做一些定时的高负载数据操作。如各种排行榜数据表的排序。同时也作定时通知类功能，比如各种全服活动的开启通知。

### collision

该处理游戏场景的物理相关行为。如场景的剖分格划分，每个剖分格中对象的维护。它为服务端的场景管理提供了基础功能。

### gameholder

游戏业务逻辑模块。游戏绝大部分的功能都在此模块中实现。也是业务逻辑和代码量最大的部分。该模块的具体细节请参考各模块的设计文档。

参考文档 <GameServer设计文档.docx>

参考文档 <gameholder.vsd>

### gameserver

GameServer游戏服务器的可执行体包装。该工程只是对游戏逻辑的一个包装，使其能够在windows和linux环境下执行。

### mfc\_gameserver

该工程使用MFC，将游戏服务器包装成一个可以在windows下有图形环境的执行体。主要是为了能够提供一个有图形环境的GameServer，便于调试战斗、AI、技能等和场景相关的逻辑。

