网络模块需求分析与设计方案

# 需求分析

网络模块是整个平台运行的基础。旧有的平台网络模块受限于FTP协议，只能通过文件进行服务器与客户端之间的通信，缺少面向连接的，实时、高效的通信机制。本网络模块拟实现基于TCP/IP协议的网络层，为平台提供面向连接的，实时、高效的服务器与客户端之间的通信机制。

网络模块进一步细分，又分为网络基础层、服务器端与客户端三大模块。网络基础层负责提供TCP/IP协议的通用服务，供服务器端与客户端调用。而服务器端与客户端则通过对套接字的调用，分别实现各自的功能。下面分别介绍各个模块的子需求。

## 网络基础层

网络基础层提供TCP/IP协议的通用服务。包括连接与通信等功能。这些服务全部通过套接字服务来实现。它将提供如下功能。

### 端口监听

该功能主要用于服务器端监听客户端发来的连接请求。通过对主机某端口的绑定实现监听，并且监听后能够接受用户的连接请求。

### 连接主机

该功能主要用于客户端向服务器发送连接请求。通过制定服务器的IP地址与端口号，向服务器发送连接请求，由服务器接受后，该连接便可用于以后的通信。

### 接受连接并建立服务套接字

该功能主要用于服务器接受客户端发来的连接请求，建立与该客户端的服务套接字，通过此独立的套接字与客户端进行通信。

### 特定类型数据的传输

该功能主要用于服务器与客户端之间的通信。传统的套接字服务只提供基于字节流的数据传输服务，该网络层将实现对于特定类型数据，如整形、浮点型、字符串类型的传输服务。进而服务器和客户端可以方便的通过该网络层组合收发复杂类型的数据结构。

### 文件传输

该功能主要用于服务器与客户端之间的文件上传、下载。

## 服务器端

服务器端建立在网络基础层之上，通过对网络基础层的调用，实现服务器端的各个服务。它包括如下功能。

### 主线程监听客户端的连接请求

服务器运行于多线程环境。其中有一个主线程，时刻监听来自客户端的连接请求。如遇来自客户端的连接请求，将开辟另外一个线程及套接字连接专门服务该客户端。

### 服务线程的建立

当主线程监听到一个来自客户端的连接请求，将创建一个新的线程服务该客户端。该服务线程将开始接受来自客户端的全部服务请求，包括用户注册、登陆，文件传输等。该线程将拥有全部针对该客户端的上下文信息，这样不同的客户端将能够互相独立的运行，他们的上下文环境不会互相干扰。

### 用户注册

该功能提供于服务线程之内。当客户端发送用户注册请求，服务器端将通过对用户数据的查询及判断，来完成用户注册功能。包括用户名检测，密码匹配等检测，并最终返回注册成功或失败的信息。

### 用户登录

该功能提供于服务线程之内。当客户端发送用户登录请求，服务器端将通过对用户数据的查询及判断，来完成用户登录功能。包括用户名是否存在，密码是否正确等检测，并最终返回登录成功或失败的信息。

### 文件传输

该功能提供与服务线程之内。在客户端已经登录的前提下，若客户端发来文件上传请求，则接收客户端上传的文件，并将之保存在用户独立的文件夹下。若客户端发来文件下载请求，则传输客户端请求的文件。该功能只是出于演示目的，未来会根据未来具体场景的具体要求，实现具体文件的上传、下载功能。

## 客户端

客户端建立在网络基础层之上，通过对网络基础层的调用，实现客户端的各个服务。它包括如下功能。

### 连接服务器

当用户有对服务器的请求时，即与服务器请求连接。通过预先设定好的服务器IP地址以及服务端口，向服务器端发送连接请求。若服务器接受请求，则连接建立成功，即可以进行接下来的通信。

### 用户注册

对于一个新的用户，需要先注册一个账户，以享受服务器提供的各种服务。在已经连接至服务器的前提下，用户可以指定自己的用户名密码并发送给服务器完成注册。

### 用户登陆

对于已经拥有账户的用户，可以将自己的账号用户名以及密码发送给服务器以登陆。

### 文件传输

在客户端已经登录的前提下，可以进行文件的上传和下载。用户可以将本地的文件上传至服务器上的用户个人文件夹下，亦可以讲服务器上的用户文件下载至本地。该功能只是出于演示目的，未来会根据未来具体场景的具体要求，实现具体文件的上传、下载功能。

# 设计方案

套接字服务最早起源于Linux操作系统上的网络编程，后来逐渐发展成一套标准。微软亦在Windows平台上提供了同样标准的库函数，即winsock。但该库只提供了基于字节流的套接字服务，于是在网络基础层，我们将对winsock进行必要封装，隐藏一些不必要的细节，向上提供给服务器端和客户端一套简单、明确、高效并且通用的接口。服务器端和客户端则提供给最终的应用程序一套专用接口，供应用程序最终访问到服务器端和客户端的各个服务。

## 网络基础层

主要通过两个类Socket和TcpSocket，对winsock库进行封装，便于服务器端和客户端调用。

### Socket类的设计

尽管平台所用到的网络基础层是基于TCP/IP协议的实现，但我们仍留出了必要的余地留给以后其他协议的兼容实现。该Socket类即出于此目的。它主要抽象了一个套接字服务的必要接口，并且并未实现部分协议相关的功能，这些功能将由具体的协议类负责实现。另一方面，他实现了那些行为不依赖于具体协议的功能，并将其实现。这样，后面的协议（如UDP协议）可以继承该类，只实现其协议相关的部分，便可以将具体的网络层替换成新的协议。Socket类主要提供如下接口。

#### 初始化

初始化该套接字。

#### 关闭

关闭该套接字。

#### 接收字节流

接收某特定长度的字节流。因不同协议有其不同的字节流发送方法，故在该类中不实现该接口，而将其彻底抽象出来，留待具体协议的套接字来实现。

#### 发送字节流

发送某特定长度的字节流。因不同协议有其不同的字节流发送方法，故在该类中不实现该接口，而将其彻底抽象出来，留待具体协议的套接字来实现。

#### 接收8位无符号整形

使用抽象的字节流接收函数，接收1个字节数据，并转型成8位无符号整形。

#### 发送8位无符号整形

使用抽象的字节流发送函数，发送1个字节数据。

#### 接收16位无符号整形

使用抽象的字节流接收函数，接收2个字节数据，并转型成16位无符号整形。

#### 发送16位无符号整形

使用抽象的字节流发送函数，发送2个字节数据。

#### 接收32位无符号整形

使用抽象的字节流接收函数，接收4个字节数据，并转型成32位无符号整形。

#### 发送32位无符号整形

使用抽象的字节流发送函数，发送4个字节数据。

#### 接收64位无符号整形

使用抽象的字节流接收函数，接收8个字节数据，并转型成64位无符号整形。

#### 发送64位无符号整形

使用抽象的字节流发送函数，发送8个字节数据。

#### 接收64位双精度浮点型

使用抽象的字节流接收函数，接收8个字节数据，并转型成64位双精度浮点型。

#### 发送64位双精度浮点型

使用抽象的字节流发送函数，发送8个字节数据。

#### 接收字符串

使用抽象的字节流接收函数，先接收32位无符号整数作为接下来字符串的长度，再接收该长度的字符串。

#### 发送字符串

使用抽象的字节流发送函数，先将该字符串的长度作为32位无符号整数发送，再发送该字符串。

#### 接收宽字符串

使用抽象的字节流接收函数，先接收32位无符号整数作为接下来宽字符串的长度，再接收该长度的宽字符串。

#### 发送宽字符串

使用抽象的字节流发送函数，先将该宽字符串的长度作为32位无符号整数发送，再发送该宽字符串。

#### 接收结果代码

使用抽象的字节流接收函数，接收4个字节数据，并转型成32位无符号整形。

#### 发送结果代码

使用抽象的字节流发送函数，发送4个字节数据。

### TCPSocket类的设计

TcpSocket类真正实现了基于TCP/IP协议的网络基础层特定功能。它包括监听、连接、接收等功能，以及基于TCP套接字的字节流收发函数，并在其中添加了文件传输功能。

#### 初始化

初始化该TCP套接字，包括创建真正的winsock套接字，设置套接字协议为TCP协议，并设置无延迟的传输。

#### 关闭

关闭真正的winsock套接字。

#### 监听

监听主机上某一端口。

#### 连接

连接至远程主机，需指定远程主机的IP地址及端口号。

#### 接收

接收一个连接，并开创建服务套接字。

#### 接收字节流

基于TCP套接字的字节流接收函数。

#### 发送字节流

基于TCP套接字的字节流发送函数。

## 服务器端

服务器端主要由类Server来实现，它封装了服务器端的各个功能，并通过网络基础层来实现。

### 初始化

初始化该服务器。包括初始化网络基础层提供的TcpSocket类对象。

### 关闭

关闭服务器。主要是关闭TcpSocket对象。

### 监听

主线程执行的函数。它调用TcpSocket对象来监听客户端发来的连接请求，在接收到请求后，随即开启独立的服务线程服务该客户端，并初始化该服务线程上下文。

### 服务线程

服务线程主要由内部结构ServiceThread来实现。它拥有一个主的服务函数，该函数主要接收各种命令，并且根据不同的命令调用不同的服务函数。这些服务包括：

#### 注册

响应客户端的注册请求，通过TcpSocket对象接收对应注册数据，查找数据文件判断注册信息是否正确，执行注册操作并返回结果。

#### 登陆

响应客户端的登陆请求，通过TcpSocket对象接收对应注册数据，查找数据文件判断登陆信息是否正确，并返回结果。

#### 上传

响应客户端的文件上传请求，通过TcpSocket对象接收文件，将文件上传至用户个人目录下。

#### 下载

响应客户端的文件下载请求，通过TcpSocket对象传输文件，将用户个人目录下的指定文件传输回客户端。

#### 其他

未来服务端的更多服务，将全部添加在该服务线程中。

## 客户端

客户端主要由类Client来实现，它封装了客户端的各个功能，并通过网络基础层来实现。

### 初始化

初始化该客户端。包括初始化客户端的TcpSocket类对象。

### 关闭

关闭该客户端。主要是关闭TcpSocket对象。

### 连接

使用TcpSocket对象连接至远程的服务器。

### 断开连接

断开与远程服务器的连接。

### 注册

向服务器发送一个注册请求，接下来通过TcpSocket对象来发送对应数据，并接收结果。

### 登陆

向服务器发送一个登陆请求，接下来通过TcpSocket对象来发送对应数据，并接收结果。

### 上传

向服务器上传一个文件，通过TcpSocket对象来传输文件。

### 下载

向服务器发送一个下载请求，来通过TcpSocket对象来发送要下载的文件名，并接收该文件。