DWAR聊天系统接口设计

1. 公共头结构定义

DWAR聊天系统采用UDP协议进行聊天数据通信，服务器端地址以及端口应当从服务器列表刷新时动态获取；

涉及到的数据结构定义如下：

struct HEADER {

uint32\_t length;

uint16\_t command;

uint16\_t result;

};

typedef unsigned long long id\_type

id\_type 数据类型为 无符号64位整数。

下列所有二进制命令结构定义全部按照4字节对齐，使用LITTLE-ENDIAN方式排列。

HEADER头结构为所有通信交换数据包的前缀，即任何命令包最前面都是HEADER头数据结构。其中length是整个数据包的长度及sizeof(HEADER)+后续数据长度，command为命令字（参考后续定义），result为相应命令时附带的命令执行结果，如果是请求命令或者此字段无意义时，默认填充0或者CC\_ERROR\_SUCCESS。

1. 聊天命令定义

参考command-def.hpp中的聊天命令枚举定义（ENUM\_CHAT2CLIENT），详细说明如下：

CHAT2CLIENT\_LOGIN 登陆

CHAT2CLIENT\_GET\_HISTORY 获取历史记录

CHAT2CLIENT\_TALKING 发言

CHAT2CLIENT\_LOGOFF 登出

CHAT2CLIENT\_GET\_ONLINE\_PLAYER\_NUMBER 获取在线人数

CHAT2CLIENT\_SYSTEM\_BOARDCAST 系统广播

CHAT2CLIENT\_NB 无意义

CHAT2CLIENT\_LOGIN\_RESULT 登陆结果

CHAT2CLIENT\_NOT\_LOGIN 未登陆

CHAT2CLIENT\_GET\_ONLINE\_PLAYER\_NUMBER\_RESULT 获取在线用户数结果

错误码定义：

CC\_ERROR\_SUCCESS

CC\_ERROR\_INVALID\_CHECKSUM 登陆时校验错误

CC\_ERROR\_INVALID\_DIGIEST\_LEN 登陆时签名长度错误

1. CHAT2CLIENT\_LOGIN 登陆聊天系统命令（客户端到服务器）

客户端登陆聊天系统，构造如下数据结构：

struct HEADER\_CHAT2CLIENT\_LOGIN {

HEADER header;

id\_type whoami;

id\_type group;

id\_type timestamp;

uint32\_t seed;

uint32\_t digiest\_len;

char digiest[0];

};

登陆系统使用算法简单验证，算法为SHA1+XOR；具体为整个头使用SHA1进行签名，然后对固定字符串依按字符进行异或，常量串为“The quick brown fox jumps over the lazy dog”。则改结构的填充为如下伪代码：

HEADER\_CHAT2CLIENT\_LOGIN header =

{

{

sizeof(HEADER\_CHAT2CLIENT\_LOGIN) + 20,

CHAT2CLIENT\_LOGIN,

CC\_ERROR\_SUCCESS,

},

xxx,此处填写从WEB接口获取到的玩家ID

XXX，此处填写从WEB接口获取的军团ID

ZZZ，填写当前时间错

SSS，随机值

20，SHA1固定签名长度20

}

随后对以上结构进行SHA1计算，得到结果后使用常量字符串依次异或得到结果，附加到该结构后面，发送数据到服务器相应的UDP端口，并等待处理服务器返回。如果服务器返回HEADER\_CHAT2CLIENT\_LOGIN\_RESULT结构，其中命令为CHAT2CLIENT\_LOGIN\_RESULT并且结果为CC\_ERROR\_SUCCESS则表示登陆成功，否则为登陆失败。

注意，所有命令都只有在登陆成功后服务器才能有效处理。

1. CHAT2CLIENT\_GET\_HISTORY 获取历史记录命令（客户端到服务器）

该命令数据结构定义如下：

struct HEADER\_CHAT2CLIENT\_GET\_HISTORY {

HEADER header;

id\_type whoami;

id\_type channel; //channel or user identifier

unsigned long long timestamp;

};

enum ENUM\_CHAT\_CHANNEL {

CHANNEL\_WORLD = 0,

CHANNEL\_ARMY\_GROUP,

CHANNEL\_PRIVATE,

CHANNEL\_NB,

};

其中channel为一个通道号，参考枚举ENUM\_CHAT\_CHANNEL，channel可以是CHANNEL\_WORLD或者CHANNEL\_ARMY\_GROUP或者CHANNEL\_PRIVATE，表明欲获取的历史记录通道类型。

Timestamp用来表明用户最新接收数据的时间错，也表明只想接收比该时间错更新的聊天信息；参考紧接的聊天命令，客户端应该使用保留一个最近一次接受到聊天消息的时间，在适当的时候用这个时间来通知服务器发送历史信息时进行信息过滤发送。

1. CHAT2CLIENT\_TALKING聊天命令（客户端到服务器、服务器到客户端）

该命令通信结构定义如下：

struct HEADER\_CHAT2CLIENT\_TALKING {

HEADER header;

id\_type whoami;

id\_type channel; //channel or user identifier

id\_type timestamp;

uint32\_t length;

uint32\_t camp\_id;

char data[0];

};

其中channel为CHANNEL\_WORLD或者CHANNEL\_ARMY\_GROUP，如果非该2种值，则channel被视为另一个用户ID，如果该用户在线则直接发送消息到该用户，否则记录到MYSQL数据库，在适当的时候进行传输；

timestamp在用户发送时无意义，可以忽略；服务器发送的时候将填充该字段标志该信息产生的时间信息。参考获取历史记录信息，用户接收聊天命令时应该更新自己的最后聊天时间为该值。

Length为data所指示的JSON序列化数据长度；

Camp\_id为阵营ID，用来实现聊天消息阵营区分；

1. CHAT2CLIENT\_LOGOFF登出命令（客户端到服务器）

该命令通信结构定义如下：

struct HEADER\_CHAT2CLIENT\_LOGOFF {

HEADER header;

id\_type whoami;

};

1. CHAT2CLIENT\_GET\_ONLINE\_PLAYER\_NUMBER 获取在线人数（客户端到服务器）

获取在线人数命令结构定义：

struct HEADER\_CHAT2CLIENT\_GET\_ONLINE\_PLAYER\_NUMBER {

HEADER header;

id\_type whoami;

};

该命令的相应命令结构定义为：

struct HEADER\_CHAT2CLIENT\_GET\_ONLINE\_PLAYER\_NUMBER\_RESULT {

HEADER header;

int32\_t world;

int32\_t group;

};

world/group分别为世界频道人数和用户所在军团人数。

1. CHAT2CLIENT\_SYSTEM\_BOARDCAST系统广播（服务器到客户端）

系统广播命令结构定义如下：

struct HEADER\_CHAT2CLIENT\_SYSTEM\_BOARDCAST {

HEADER header;

// uint32\_t type;

uint32\_t length;

char data[0];

};

length为data所携带的JSON数据长度；

data为JSON序列化数据。

1. CHAT2CLIENT\_LOGIN\_RESULT登陆结果（服务器到客户端）

参考登陆命令

1. CHAT2CLIENT\_NOT\_LOGIN客户端未登陆（服务器到客户端）

该命令结构为如下定义：

typedef HEADER HEADER\_CHAT2CLIENT\_NOT\_LOGIN;

即仅有一个HEADER头，在任何时候服务器接受到客户发送的除登陆命令之外的命令，如果命令中提供的whoami在服务器用户列表中无法发现的话则会相应这个数据包。

1. CHAT2CLIENT\_GET\_ONLINE\_PLAYER\_NUMBER\_RESULT获取在线人数结果（服务器到客户端）

参考获取在线人数命令。