# 工会（诸侯国）系统

# 详细设计说明书

**项目信息：**

|  |  |  |
| --- | --- | --- |
| 项目名称 | 项目编码 | 项目级别 |
|  |  |  |
| 客户名称 | AM 姓名 | PM 姓名 |
|  |  |  |

版本信息：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本 | 日期 | 修改内容 | 审核 | 作者 |
| 0.1 | 091103 | 初始版本 |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

[工会（诸侯国）系统 1](#_Toc245293023)

[详细设计说明书 1](#_Toc245293024)

[1引言 3](#_Toc245293025)

[1.1编写目的 3](#_Toc245293026)

[1.2背景 3](#_Toc245293027)

[1.3缩写 3](#_Toc245293028)

[1.4参考资料 3](#_Toc245293029)

[2、组队系统的内部结构 3](#_Toc245293030)

[3 主体流程 6](#_Toc245293031)

[3.1 启动，初始化 6](#_Toc245293032)

[3.2 主体循环 7](#_Toc245293033)

[3.3 消息函数 7](#_Toc245293034)

[4 主要的交互流程 9](#_Toc245293035)

[4.1 创建工会 9](#_Toc245293036)

[4.2 解散工会 10](#_Toc245293037)

# 1引言

服务端的工会系统相关的操作请求

# 1.1编写目的

本文档的编写目的是详细介绍服务器端的工会系统的实现机制，为将来在此系统上开发的人员提供支持。

# 1.2背景

整个组队系统是用标准C++语言编写的。

# 1.3缩写

LS：LoginServer

GWS：GameWorldServer

RS：RegionServer

# 1.4参考资料

GameWorldServer.chm

RegonServer.chm

# 2、组队系统的内部结构

在GWS服务端结构

// 工会（诸侯国）

class CUnion

{

friend class CUnionMgr;

public:

CUnion(CUnionMgr \*pUnionMgr, DWORD unionID, DWORD captainID, std::string name, string enounce);

~CUnion();

// bool Serialize(CG\_CmdPacket \*cmd);

//得到ID

DWORD GetID() { return m\_core.id; }

//得到名字

string GetName() { return m\_core.name; }

//得到标题

DWORD GetCaption() { return m\_core.captainID; }

//得到工会数据

SUnionData\* GetData() { return &m\_core; }

// 查找公民

SUnionCitizen\* Find(DWORD playerID);

// 增加公民

bool AddCitizen(DWORD playerID, short level, BYTE bOnline = true, EUnionOfficial official = UO\_Guarder);

// 删除公民

bool RemoveCitizen(DWORD playerID);

// 设置称号

bool SetCitizenNickName(DWORD playerID, char\* NickName);

// 任命官职

bool Appoint(DWORD playerID, EUnionOfficial official);

//工会玩家数量

UINT CitizenCount() { return (UINT)m\_core.arrCitizen.size(); }

// 升级诸侯国

bool UpdateLevel();

// 增加/减少积分

bool AddScore(int count);

// 设置诸侯

bool SetCaptain(DWORD playerID);

// 设置名字

bool SetName(string name);

// 设置标志

bool SetIcon(char\* data, int dataLen, DWORD &outIconID);

// 设置公告

bool SetBulletin(string bulletin);

// 向另一个诸侯国宣战

bool InviteWar(DWORD unionInvited);

// 玩家上线时的回调

void OnCitizenComming(DWORD playerID);

// 玩家下线时的回调

void OnCitizenGoing(DWORD playerID);

SUnionCitizen& operator[](int index) { return m\_core.arrCitizen[index]; }

private:

SUnionData m\_core;

CUnionMgr \*m\_pUnionMgr;

};

// 诸侯国管理

class CUnionMgr

{

public:

CUnionMgr();

~CUnionMgr();

//初始化

bool Init();

void Tick();

bool IsInited() { return m\_bInited; }

// 建立新诸侯国，成功时返回新诸侯国ID，失败时返回值为0

DWORD CreateUnion(DWORD playerID, short playerLevel, std::string name, string enounce);

// 删除诸侯国

bool DeleteUnion(DWORD unionID);

//查找工会

CUnion\* FindUnion(DWORD unionID);

//得到工会ID

DWORD GetUnionID(DWORD userID);

//移除废弃的工会

bool RemoveInvalidUnion();

// 玩家上线时的回调

void OnCitizenComming(DWORD playerID, int& unionID, DWORD regionID);

// 玩家下线时的回调

void OnCitizenGoing(DWORD playerID, DWORD unionID);

// 删除角色时的回调

void OnDeleteUser(DWORD userDBID);

// 创建角色时的回调

void OnCreateUser(DWORD userDBID);

//增加新的区域

void OnAddNewRegion(int regionID);

public:

vector<CUnion\*> m\_arrUnions;

hash\_map<DWORD, DWORD> m\_mapUserToUnion;

private:

bool m\_bInited;

};

# 3 主体流程

# 3.1 启动，初始化

在GWS服务端 开始执行Main函数，在其中的startup函数中

// 诸侯国，依赖于DB

if (!g\_UnionMgr.Init())

{

ERR("Union startup FAILED\n");

return false;

}

# 3.2 主体循环

在GWS里组队的主循环如下：

由于工会系统都是事件触发性的逻辑处理，和时间性无关，故不需要在主帧循环里做任何操作。

在RS的部分也不需要在主帧循环里做任何操作。

# 3.3 消息函数

在GWS服务端接收消息的函数如下

//接收到RS服务端消息

GW\_BEGIN\_CMD(cmd\_r2g\_union)

{

GW\_UnionCmdProcess(packet);

}

GW\_END\_CMD;

各消息执行函数

GW\_UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_create);

GW\_UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_dismiss\_other);

GW\_UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_change\_nickname);

GW\_UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_join);

GW\_UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_other\_join\_response);

GW\_UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_bulletin);

GW\_UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_appoint);

GW\_UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_invitewar);

GW\_UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_surrender);

GW\_UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_leave);

GW\_UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_data\_query);

GW\_UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_logo);

GW\_UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_update\_level);

GW\_UNION\_CMD\_FUNCTION(cmd\_r2g\_union\_add\_score);

GW\_UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_update\_level);

GW\_UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_player\_levelup);

在RS服务端接收函数如下

//接收客户端消息

CMD\_FUNCTION(cmd\_c2r\_subsystem\_union)

{

GET\_PARAMS();

return UnionCmdProcess\_Svr(CmdGiver, cmd);

}

各消息执行函数

UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_create);

UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_dismiss\_other);

UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_change\_nickname);

UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_join);

UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_other\_join\_response);

UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_bulletin);

UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_appoint);

UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_invitewar);

UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_surrender);

UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_leave);

UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_data\_query);

UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_logo);

UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_query\_logo\_data);

UNION\_CMD\_FUNCTION(cmd\_c2r\_union\_update\_level);

//接受GWS返回的消息

CMD\_FUNCTION(cmd\_g2r\_union)

{

return UnionCmdProcess\_FromGW(cmd);

}

各消息执行函数

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_create\_result);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_dismiss\_other\_result);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_dismissed\_by\_other);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_change\_nickname\_result);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_other\_join\_apply);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_join\_response);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_appoint\_result);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_invitewar\_result);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_surrender\_other\_response);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_leave\_result);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_data\_refresh);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_citizen\_join);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_citizen\_leave);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_citizen\_online);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_citizen\_offline);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_citizen\_official\_change);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_disband);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_new\_bulletin);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_citizen\_nickname\_change);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_logo\_data);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_update\_level\_result);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_score\_change);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_levelup);

UNION\_CMD\_FUNCTION\_FROM\_GW(cmd\_r2c\_union\_player\_levelup);

具体每个消息流程请参照

GameWorldServer.chm

RegonServer.chm

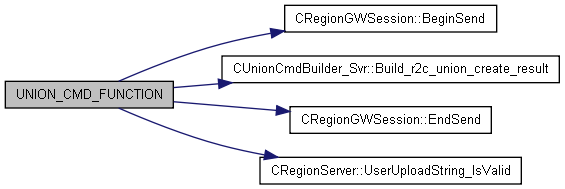
# 4 主要的交互流程

由于工会系统逻辑交互很多例如创建 解散 加入 退出 工会公告 工会邀请 工会放弃投降等等 在此只详细写出主要的

# 4.1 创建工会

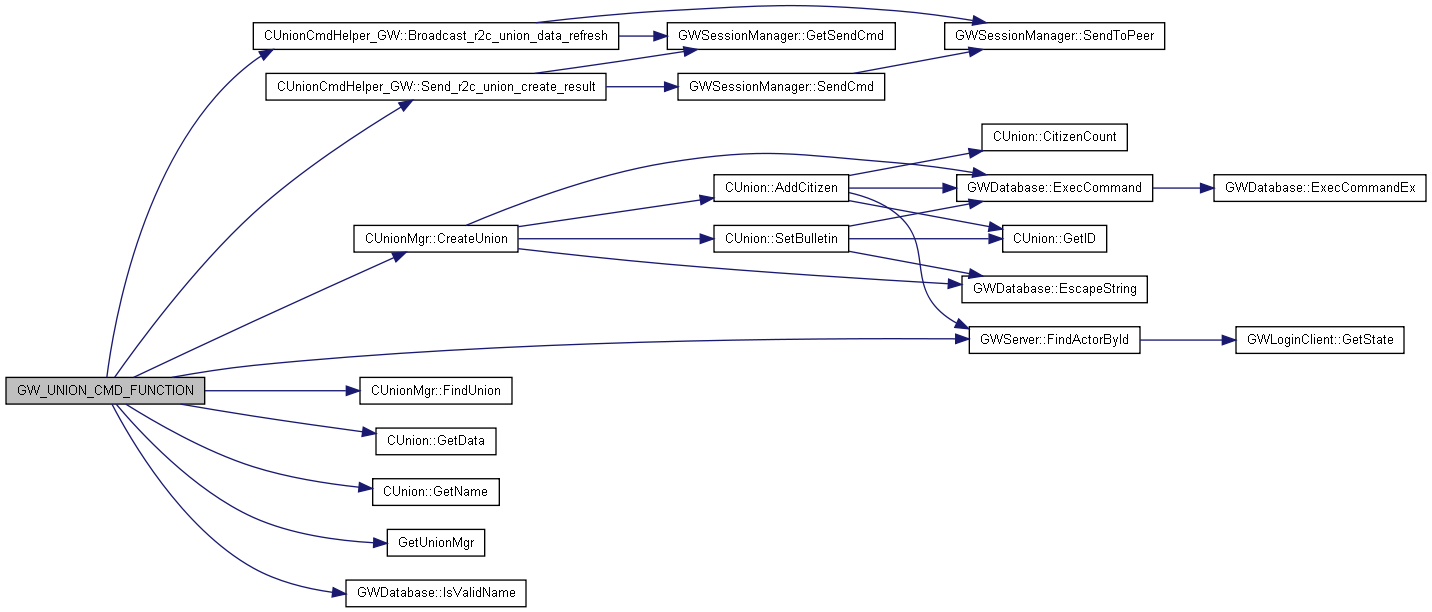
1.在函数CMD\_FUNCTION(cmd\_c2r\_subsystem\_union)接收到客户端消息cmd\_c2r\_subsystem\_union，执行函数接收到子消息是c2r\_union\_create，执行函数cmd\_c2r\_union\_create(CmdGiver, cmd)，经过所有逻辑处理发送消息c2r\_union\_create到GWS服务端。

2. 图如下：



3.在GWS服务端中函数GW\_BEGIN\_CMD(cmd\_r2g\_union)接收到服务端RS消息cmd\_c2r\_subsystem\_union，得到子消息c2r\_union\_create，执行函数CreateUnion创建工会，FindUnion函数查找到工会ID，GetData中得到工会数据。由Send\_r2c\_union\_create\_result函数将消息返回RS服务端，并且函数Broadcast\_r2c\_union\_data\_refresh广播到每个RS服务端。

4.消息图



5．函数CMD\_FUNCTION(cmd\_g2r\_union)接收到GWS服务端消息，执行函数UnionCmdProcess\_FromGW，并解析到子消息是r2c\_union\_create\_result，经过逻辑处理后由函数Build\_r2c\_union\_create\_result返回给客户端消息。

# 4.2 解散工会

1.在函数CMD\_FUNCTION(cmd\_c2r\_subsystem\_union)接收到客户端消息cmd\_c2r\_subsystem\_union，执行函数接收到子消息是c2r\_union\_dismiss\_other，执行函数cmd\_c2r\_union\_dismiss\_other，经过逻辑处理发送消息c2r\_union\_bulletin到GWS服务端。

2.GWS服务端中函数GW\_BEGIN\_CMD(cmd\_r2g\_union)接收到服务端RS消息，得到子消息c2r\_union\_dismiss\_other，执行函数cmd\_c2r\_union\_dismiss\_other，在函数FindUnion中根据ID找到工会，Union\_CanDismiss工会解散，RemoveCitizen移除掉工会成员。

3.发送和广播消息到RS服务端。

4. 函数CMD\_FUNCTION(cmd\_g2r\_union)接收到GWS服务端消息，执行函数UnionCmdProcess\_FromGW，并解析到子消息是r2c\_union\_dismiss\_other\_result，经过逻辑处理后由函数Build\_r2c\_union\_create\_result返回给客户端消息。