文本复制检测报告单(全文标明引文)

№:ADBD2018R_2018053015312720180530154823440173994786　　检测时间:2018-05-30 15:48:23

检测文献:　53140133_初芯宇_计算机科学与技术_基于cocos2d的一项RPG网络游戏软件设计与实现

作者:　初芯宇

检测范围:　中国学术期刊网络出版总库

中国博士学位论文全文数据库/中国优秀硕士学位论文全文数据库

中国重要会议论文全文数据库

中国重要报纸全文数据库

中国专利全文数据库

图书资源

优先出版文献库

大学生论文联合比对库

互联网资源(包含贴吧等论坛资源)

英文数据库(涵盖期刊、博硕、会议的英文数据以及德国Springer、英国Taylor&Francis 期刊数据库等)

港澳台学术文献库

互联网文档资源

CNKI大成编客-原创作品库

个人比对库

时间范围:　1900-01-01至2018-05-30

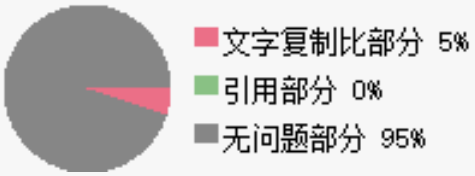
检测结果

总文字复制比：5%　　跨语言检测结果：0%

去除引用文献复制比：5%　　去除本人已发表文献复制比：5%

单篇最大文字复制比：1.3%（基于.NET的网络五子棋论文 - 豆丁网）

重复字数：	[1530]	总段落数：	[8]
总字数：	[30430]	疑似段落数：	[4]
单篇最大重复字数：	[389]	前部重合字数：	[626]
疑似段落最大重合字数：	[684]	后部重合字数：	[904]
疑似段落最小重合字数：	[32]		



指　标：☐疑似剽窃观点　☑疑似剽窃文字表述　☐疑似自我剽窃　☐疑似整体剽窃　☐过度引用

表格：0　　公式：0　　疑似文字的图片：0　　脚注与尾注：3

（注释： 无问题部分 文字复制比部分 引用部分）

1. 53140133_初芯宇_计算机科学与技术_基于cocos2d的一项RPG网络游戏软件设计与实现_第1部分

总字数：2702

相似文献列表 文字复制比：0%(0) 疑似剽窃观点：(0)

原文内容 红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容

摘要

基于cocos2d的一项RPG网络游戏软件设计与实现

随着游戏行业的发展，网络游戏的普及率也有了快速的提升。相较于其他的游戏类型，网络游戏的开发有着独特的技术和过程。那么在一次网络游戏开发中到底会遇到怎样的问题？通过一次完整的回合制RPG游戏开发发现，网络游戏开发的主要难点在于将游戏内容拆分为服务端和客户端两个部分，并且需要通过网络通信来连接两个部分，让服务端在服务器上运行并计算所有玩家的数据，客户端在客户机上运行，展示玩家的画面。这样拆分的同时，让更多的玩家有了交互的机会，但是同时也还存在很多的问题。

一方面是游戏开发本身就有的，诸如大型游戏的逻辑结构复杂，并且要面向随时可能提出的游戏策划变动，所以开发时对于系统的维护性和扩展性要求都极高。由于游戏的显示要求通常较高，所以对于显示的优化也有较高的要求。另一方面是网络通信所带来的，由于服务端只能通过网络被动接收消息而不能主动感知客户端的情况，所以游戏作弊的可能性被放大了，除此之外，网络通信还会导致两部分的交互变得更加困难，情况更加复杂多变。

本次项目的主要包括登录、移动、战斗、聊天四大功能，在此基础上再对游戏进行了部分优化和数据安全性校验，尽可能以一个商业游戏的要求来进行开发。

关键词：回合制RPG游戏，网络通信，C/S架构

Abstract

With the development of the game industry, the popularity of online games has also rapidly increased. Compared to other games, online game development has unique technologies and processes. So what problems will be encountered in the development of an online game? Through a complete turn-based RPG game development,it is discovered that the main difficulty of online game development is to split the game content into two parts of the server and the client, and need to connect two parts through the network communication, so that the server runs and calculates data for all players and the client shows for players. This split allows more players to have the opportunity to interact, but at the same time there are many problems.

On the one hand, game development has much trouble. For example, the logic structure of large-scale games is complex, and it is necessary to face changes in game planning that may be proposed at any time. Therefore, development requires high maintenance and scalability of the system. And since the display requirements of the game are generally high, there is also a high requirement for the optimization of the display.

On the other hand, because the server can only passively receive messages over the network and cannot actively perceive the situation of the client, the possibility of cheating in games is magnified. In addition, network communication can also make the interaction between the two parts more difficult. The situation is more complicated and changeable.

The project mainly includes four major functions: login, mobile, battle, and chat. Based on this, the game is partially optimized and the data security is verified. This is developed as a commercial game as far as possibly.

Key words: turn-based RPG,Telecommunication;,C/S architecture

目录

2. 53140133_初芯宇_计算机科学与技术_基于cocos2d的一项RPG网络游戏软件设计与实现_第2部分

总字数：688

相似文献列表　文字复制比：0%(0)　疑似剽窃观点：(0)

原文内容　红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容

3. 53140133_初芯宇_计算机科学与技术_基于cocos2d的一项RPG网络游戏软件设计与实现_第3部分

总字数：3477

相似文献列表　文字复制比：18%(626)　疑似剽窃观点：(0)

1	基于.NET的网络五子棋论文 - 豆丁网 -《互联网文档资源（http://www.docin.com）》- 2016	11.2%（389） 是否引证：否
2	网络游戏管理系统设计与实现 王贞(导师：沈刚) -《华中科技大学博士论文》- 2009	11.1%（386） 是否引证：否
3	论网络游戏的发展与利弊,文秘范文,东星资源网:http://www.dxf5.com/ -《网络（http://www.dxf5.com/）》- 2012	10.6%（369） 是否引证：否
4	基于JAVA的棋牌游戏系统只网络五子棋 吴雨泽 -《大学生论文联合比对库》- 2015	10.2%（353） 是否引证：否
5	基于JAVA的棋牌游戏系统之网络五子棋的开发 吴雨泽 -《大学生论文联合比对库》- 2015	10.2%（353） 是否引证：否
6	网络游戏_互动百科 -《网络（http://www.hudong.co）》-	10.1%（350） 是否引证：否
7	★论网络游戏的发展与利弊 -言小范文网 -《网络（http://www.yxtvg.com）》- 2012	10.1%（350） 是否引证：否
8	别让网游成为你生活的全部-百度文库 -《互联网文档资源（http://wenku.baidu.c）》- 2012	10.0%（347） 是否引证：否
9	三维网络游戏服务器关键技术研究与原型系统实现 陈杨(导师：刘艳) -《天津大学硕士论文》- 2009	9.4%（327） 是否引证：否
10	游戏_百度百科 -《网络（http://baike.baidu.c）》- 2010	9.3%（325） 是否引证：否
11	游戏_百度百科 -《网络（http://baike.baidu.c）》- 2010	9.3%（325） 是否引证：否
12	本周教育课_礼泉一中吧_贴吧 -《网络（http://tieba.baidu.c）》- 2010	9.3%（325） 是否引证：否

13	世界IT发展历史年表大事记 - 南三方---网站设计开发录 - CSDN博客 - 《网络（http://blog.csdn.net）》- 2017	9.3%（325） 是否引证：否
14	按电子游戏内容目的进行分类： - 《互联网文档资源（http://wenku.baidu.c）》- 2017	9.3%（325） 是否引证：否
15	网络游戏-网络游戏- 计算机百科网 计算机科学，电脑，软件，硬件，网络，数码，IT百科 -jsjbk.cn - 《网络（http://www.jsjbk.cn/）》-	9.2%（320） 是否引证：否
16	数字媒体艺术之网络游戏的发展-百度文库 - 《互联网文档资源（http://wenku.baidu.c）》- 2012	9.2%（320） 是否引证：否
17	IT发展历史-李兵的博客-科学网 - 《网络（http://www.sciencene）》-	9.2%（319） 是否引证：否
18	中国网络游戏发展对策研究 朱壮文(导师：田胜立) - 《北京印刷学院硕士论文》- 2005	8.6%（300） 是否引证：否
19	分布式网络游戏的设计与开发及相关技术研究 周宇辉(导师：申铉京) - 《吉林大学硕士论文》- 2006	8.6%（300） 是否引证：否
20	浅析游戏女性角色服饰设计 邓菁; - 《艺苑》- 2009	7.2%（252） 是否引证：否
21	基于JAVA的棋牌游戏系统之网络五子棋的开发 吴雨泽 - 《大学生论文联合比对库》- 2015	7.2%（249） 是否引证：否
22	浅析yy网页直播平台目前的现状和问题 施懿铭 - 《大学生论文联合比对库》- 2017	7.2%（249） 是否引证：否
23	游戏网络技术研究与设计 刘勇(导师：徐远纯) - 《景德镇陶瓷学院硕士论文》- 2009	7.1%（247） 是否引证：否
24	CNNIC：网络娱乐应用进一步转向移动端 行业不断正规化_国内 - 《网络（http://politics.gmw.）》- 2017	5.3%（186） 是否引证：否
25	浅谈游戏测试1-百度文库 - 《互联网文档资源（http://wenku.baidu.c）》- 2012	4.9%（171） 是否引证：否
26	从外部性效应分析中国的网络游戏市场 张杰; - 《社科纵横(新理论版)》- 2009	1.2%（41） 是否引证：否

原文内容　红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容

第1章绪论

网络游戏开发概述

网络游戏，即“在线游戏”，是“网游”的简称。指玩家利用网络上的虚拟人物与多人或单个玩家进行游戏，从而达到娱乐的目的。

根据CNNIC第40次调查报告的数据，2017年上半年网络游戏用户数量稳步增长。到2017年6月，中国网络游戏用户达到4.22 亿，较去年底增长460 万，占网民总数的56.1%。手机网络游戏用户规模为3.85 亿，较去年底增长3380 万，占手机用户总数的53.3%，手机网络游戏用户规模增长率达到9.6%。

网络游戏发展现状表明，2017 年上半年，国内网络游戏产业稳定发展，收入规模显著扩大，游戏与 IP 产业链其他环节的联系日益加深。从游戏本身的发展来看，竞技与社交仍是促使重度游戏保持极高营收能力的核心要素，而随着游戏用户群体的不断垂直细分，作为小众市场的单机游戏有望成为新的行业增长点。从行业发展上看，营收增长与产业联动加深是 2017 年上半年网络游戏行业两大发展特点。

网络游戏简单发展历史回顾

从1969网络游戏兴起，发展迄今共经历了约三个时代：

第一代网络游戏：1969年至1977年

起初由于计算机硬件和软件没有统一的技术标准，所以第一代网络游戏的平台和语言各不相同。这个时期的游戏大多数是试验品，运行在高等院校的大型主机上，属于非持续性的游戏，机器重启之后所有数据都会丢失，而且游戏只能在同一服务器系统的内部执行，无法跨系统运行。

第二代网络游戏：1978年至1995年

在这一阶段，开始有专业的游戏开发商和发行商涉足网络游戏，并且推出了第一批具有普及意义的网络游戏。这时的网络游戏有了“可持续性”的概念，即玩家操纵的角色可以在游戏世界中长期发展，而不是每一次开启游戏都会重新生成数据。游戏也开始可以跨系统运行，玩家只要拥有电脑和调制解调器，并且硬件兼容，就可以连入当时的任何一款网络游戏。

第三代网络游戏：1996年至今

越来越多的游戏开发商和发行商进入这个行业，一个规模庞大的产业生态环境开始成型。人们开始更多的思考网络游戏的设计方法和经营方法，希望归纳出一套系统的理论。同时，大型网络游戏（MMOG）的概念浮出水面，使得网络游戏不再

依托于单一的服务商和服务平台，而是直接接入互联网平台，在全球范围内形成了一个大一统的市场。

网络游戏开发的关键问题

开发网络游戏有两大块技术必须掌握，即使在开发中进行严格的分工合作，这两方面也都需要有所了解。

网络编程

游戏编程

在一款网络游戏中，各个客户端之间有频繁的数据传输，为了处理这种大量的传输需求，网络编程是必需的技能。

网络编程实际上是一个很广泛的概念，比如外置的存储设备通过USB接口与PC机相连也需要网络编程，蓝牙和红外线等通信也属于网络编程范畴。而网络游戏中只需要用到以互联网编程为主的技术。

其次是游戏编程，这一部分的主要目的，简单来说就是将画面绘制给玩家。因为游戏编程是只在玩家所使用的客户机上运行的，所以实际上的逻辑处理流程并没有统一的标准，不像网络编程有套接字API这种标准库来支撑。通常各种类型的游戏都会有各种不同的实现方式，甚至于连开发工具都参差不齐。不过游戏编程也有几乎不变的基本处理逻辑：

在不考虑各方面的优化情况下，游戏编程的主要目的就是将点绘制在屏幕上，在正式的游戏中，这通常包括两步。首先进行初始化，对一些数据做出初始的赋值，代表着游戏还未开始时的画面数据，然后是无限循环，在游戏时间内，使用一个无限进行的循环来反复进行接收用户输入，计算当前画面，将当前画面绘制在屏幕上三个步骤，以此来重复刷新用户所能看到的画面，让整个游戏的画面动起来。

游戏主要功能点介绍

本次毕业设计旨在开发一款回合制的MMORPG网络游戏，使用的美术资源均为二维图形，用户视角为固定的45°俯视。其中包含游戏的客户端和服务端。玩家通过游戏客户端登录到游戏服务器，在游戏服务器中进行游戏。

基础设定

角色模型

8个朝向，每45°一个朝向区域，向不同朝向区域移动的操作均使用对应朝向的模型及动作动画。

可针对不同操作和状态播放指定动作动画。

NPC模型

4个朝向，每90°一个朝向区域，向不同朝向区域移动的操作均使用对应朝向的模型及动作动画。

NPC首次刷出为默认配置朝向（需求可配置）。不同朝向区域的玩家对NPC进行点击时，NPC将旋转并保持至相应朝向（是否旋转需求可配置）。

可针对不同操作和状态播放指定动作动画。

可点击，点击后可设定触发事件。

主场景元素：

坐标：角色在场景内位置表示。

障碍：表示场景的不可移动区域，玩家无法通过操作移动到该区域。

地表：地表由一张整图构成，它处于整个场景的最底层。

NPC：放置在场景内的角色，角色禁止不动，但可以设置角色的朝向（4个朝向即可），玩家可以点击与之对话，对话形式以UI表现。

登录系统

游戏启动后显示启动界面，在启动界面里会展示游戏的载入进度，载入完成后关闭，进入游戏的登录界面。登录界面提供玩家输入用户名和密码的UI用户界面，玩家输入后即可验证登录。在玩家登录校验完毕后进入角色选择界面，在这个界面下可以选择该账号已经创建过的角色开始游戏，或者进入新的角色创建界面为该账号创建其他角色进行游戏。创建新的角色共分为两步，即为新角色选择职业和为新角色输入姓名。

移动系统

移动功能依赖于游戏的主界面，主界面是包含预设的主场景元素的、在非战斗场景内始终开启的一级操作界面，具有大量的图标和人物基本信息展示。

玩家对游戏内的场景进行点击，角色将进行自动寻路并绕过障碍，移动到玩家点击的位置。移动过程中镜头以玩家角色模型为中心点跟随移动。在场景内，玩家可以同步看到其他玩家的移动操作。当遇到玩家接近场景边缘的特殊情况时，为使镜头不会超出场景边缘，此时在保证镜头不超出边界的情况尽量跟随玩家移动。除此之外，角色还可以切换到其他场景。

场景中的角色和NPC严格符合其基础设定。

战斗系统

战斗以敌我双方若干个角色为主体进行团队战，每个角色依次行动，可以进行攻击、技能、防御、援护、使用道具等指令，当其中一方所有全部角色死亡，则战斗结束。战斗界面UI有各种角色操作指令可以供点击操作。

战斗场景：战斗场景为一张静态图，敌我双方角色分别处于场景左上角和右下角进行对峙。

敌我双方：每一方最多有10名角色。

站位：每一方有4*5的方格表示站位，每次战斗会根据玩家的情况来决定角色的站位，可配置默认情况的站位。

战斗流程：每个回合会有30秒的操作阶段，当所有玩家完成技能选择或操作时间结束后进行回合演示。演示阶段按照所

有角色的速度进行排序。速度快的先行动，处于死亡状态的角色不能行动，处于特殊状态的角色限制部分行动指令。所有角色演示完毕后进入新一个回合的操作阶段。

角色行为：

普攻：角色进行一次普通攻击。

技能：包含但不限于以下技能类型，群体攻击技能、单体技能、群体恢复、单体恢复、复活等。

防御：受到攻击的伤害减少50%

援护：对己方角色进行援护，在队友被攻击时概率触发援护行为，减少其受到的伤害。

战斗角色基本属性：

速度：决定角色的出手顺序，速度越高，回合内优先出手。

血量：表示角色的生命状态，血量有最大血量和当前血量，当前血量降为0则角色死亡

物攻、物防：当角色进行普通攻击或物理技能进行攻击，攻击力取决于技能属性和物攻值，被攻击角色的防御力由物防值决定。根据攻击力和防御力，确定最终被物理攻击的角色损失多少血量。

法攻、法防：类似物攻物防

Buff：角色身上附带的增益效果，增益效果能提高角色战斗能力，增益效果包括但不限于以下类型：增加角色某些基础属性、攻击可附加对方特殊状态、攻击吸血等。

DeBuff：角色身上附带的负面效果，减弱角色战斗能力。负面效果包含但不限于以下类型：降低角色某些基础属性、使角色处于某种特殊状态。

特殊状态：眩晕、封技能、封普攻、混乱（敌我不分）

聊天

客户端之间通过聊天系统互相传递消息。聊天系统分为多个频道：世界、队伍、附近、私聊。聊天信息可以包含文字信息，表情信息，自定义链接信息（如装备链接、角色链接等）或者三者的任意组合。可以为服务端配置发言限制（如间隔时间，发言条数）。

指　标
疑似剽窃文字表述

1. 达到 4.22 亿，较去年底增长460 万，占网民总数的56.1%。手机网络游戏用户规模为3.85 亿，较去年底增长3380 万，占手机用户总数的53.
2. 竞技与社交仍是促使重度游戏保持极高营收能力的核心要素，而随着游戏用户群体的不断垂直细分，作为小众市场的单机游戏有望成为新的行业增长点。
3. 行业发展上看，营收增长与产业联动加深是 2017 年上半年网络游戏行业两大发展特点。
4. 网络游戏兴起，发展迄今共经历了约三个时代：
   第一代网络游戏：1969年至1977年
   起初由于计算机硬件和软件没有统一的技术标准，所以第一代网络游戏的平台
5. 游戏只能在同一服务器系统的内部执行，无法跨系统运行。
   第二代网络游戏：1978年至1995年
   在这一阶段，开始有专业的游戏开发商和发行商涉足网络游戏，并且推出了第一批具有普及意义的网络游戏。这时的网络游戏有了“可持续性”的概念，即玩家操纵的角色可以
6. 游戏也开始可以跨系统运行，玩家只要拥有电脑和调制解调器，并且硬件兼容，就可以连入当时的任何一款网络游戏。
   第三代网络游戏：1996年至今
   越来越多的游戏开发商和发行商进入这个行业，一个规模庞大的产业生态环境开始成型。人们开始更多的思考网络游戏的设计方法和经营方法，希望归纳出一套系统的理论。

4. 53140133_初芯宇_计算机科学与技术_基于cocos2d的一项RPG网络游戏软件设计与实现_第4部分

总字数：4073

相似文献列表　文字复制比：16.8%(684)　疑似剽窃观点：(0)

1	做游戏开发怎能不知道这个跨平台游戏开发框架——Cocos2D-X_CPlusPlus -《网络（http://chuansong.me/）》- 2017	6.2%（253） 是否引证：否
2	游戏开发 吴佳耀 -《大学生论文联合比对库》- 2017-03-23	6.1%（248） 是否引证：否
3	计算13-1班1306010118史金龙 -《大学生论文联合比对库》- 2017-06-24	5.4%（220） 是否引证：否
4	e4_孙_手机游戏	5.1%（209）

	孙 - 《大学生论文联合比对库》- 2017-03-23	是否引证：否
5	基于Cocos2d-x的游戏设计与实现	5.1%（209）
	韩子健 - 《大学生论文联合比对库》- 2017-04-27	是否引证：否
6	白钢_2013081101_基于Cocos2d的棋牌游戏设计与实现	5.1%（209）
	白钢 - 《大学生论文联合比对库》- 2017-06-05	是否引证：否
7	基于Cocos2d-X游戏引擎的棋牌类游戏	5.1%（209）
	王斌 - 《大学生论文联合比对库》- 2017-05-18	是否引证：否
8	一种Slot类游戏的主体逻辑及骨骼动画模块的设计与实现	5.1%（209）
	刘岸 - 《大学生论文联合比对库》- 2017-05-23	是否引证：否
9	基于Cocos2d-X游戏引擎的棋牌类游戏	5.1%（209）
	王斌 - 《大学生论文联合比对库》- 2017-05-28	是否引证：否
10	Android植物大战僵尸游戏设计与实现	5.1%（209）
	袁仁俊 - 《大学生论文联合比对库》- 2017-04-19	是否引证：否
11	植物大战僵尸的游戏与设计	5.1%（209）
	袁仁俊 - 《大学生论文联合比对库》- 2017-04-24	是否引证：否
12	亓云浩-1310750031-基于Android的幼儿手指画游戏的设计	5.1%（209）
	亓云浩 - 《大学生论文联合比对库》- 2017-05-17	是否引证：否
13	传统文化元素在CG角色原画设定中的应用	4.1%（165）
	李羿臻 - 《大学生论文联合比对库》- 2017-04-29	是否引证：否
14	刘桂元_计算机科学与技术_20133221_基于Android系统的手机游戏的设计与实现	3.8%（156）
	计算机科学与技术 - 《大学生论文联合比对库》- 2017-05-19	是否引证：否
15	基于安卓的纸牌游戏	2.7%（108）
	熊师怡 - 《大学生论文联合比对库》- 2017-05-10	是否引证：否
16	基于cocos2D-X引擎《梦幻九阴》的开发	2.4%（99）
	王波 - 《大学生论文联合比对库》- 2015-05-22	是否引证：否
17	回合制游戏-其它- 鞭牛士网络编辑百科知识库 百科，鞭牛士百科，网编百科	2.1%（85）
	-《网络（http://baike.bianews）》-	是否引证：否
18	免费回合制网络游戏排行榜 -网络游戏排行榜-数据排行-企业管理世界网	2.1%（85）
	-《网络（http://www.jakj.com.）》- 2011	是否引证：否
19	回合制网游 人类的进化史_网游基地_娱乐	2.1%（85）
	-《网络（http://www.xici.net/）》- 2013	是否引证：否
20	常州动漫节探索“炮炮兵”的世界_动漫	2.1%（85）
	-《网络（http://comic.qq.com/）》- 2013	是否引证：否
21	基于Cocos2d-x引擎的游戏的寻路算法研究与实现	2.1%（84）
	朱京晶(导师：魏慧琴) -《北京交通大学博士论文》- 2017-06-01	是否引证：否
22	基于 HTML5 的拼图类游戏的设计与开发	1.6%（67）
	王迪 - 《大学生论文联合比对库》- 2017-04-30	是否引证：否
23	基于HTML5的拼图类游戏的设计与开发	1.6%（67）
	王迪 - 《大学生论文联合比对库》- 2017-05-05	是否引证：否
24	20132230_王迪_基于HTML5的拼图类游戏的设计与开发	1.6%（67）
	王迪 - 《大学生论文联合比对库》- 2017-05-26	是否引证：否
25	基于安卓的纸牌游戏	0.8%（34）
	熊师怡 - 《大学生论文联合比对库》- 2017-05-19	是否引证：否

原文内容　红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容

第2章开发工具与核心理论

与其他软件比较，网络游戏通常包含了大量的计算逻辑和变现逻辑。在开发过程中，通常也会有一些核心理论和工具作为支撑。本章将对开发过程中使用的工具、核心理论基础和网络游戏的几种架构模型进行详细介绍。

开发语言与环境

本次使用C++编程语言开发游戏的服务端和客户端。

C++语言继承自C语言，在程序开发时可以根据实际的情况使用C语言风格的过程化开发，也可以使用以数据结构为核心的对象化程序设计。C++由于其在计算机上高效运行的特点而在游戏开发中被广泛运用。

本项目所有的程序开发时使用Microsoft Visual Studio工具，均运行于Windows操作系统。

Microsoft Visual Studio简称VS，是美国微软公司的开发工具包系列产品，是一个基本完整的开发工具集，包括了整个开发流程中所需要的大部分工具。

Microsoft Windows是美国微软公司研发的一套操作系统，采用图形化模式GUI。

网络游戏架构模型

网络游戏的开发模式由物理和逻辑两部分组成，根据其特性的不同，相应的物理架构和逻辑架构的选择也有所区分。物理架构和逻辑架构分别有两种不同的选择，组合起来共有4种形式。

网络游戏常用架构

物理架构就是指“实际进行通信的设备之间存在怎样的关系”，它的两种形式分别是C/S模型和P2P模型。物理架构的选择将会决定游戏开发在四个方面的特性，分别是延迟、服务器设备成本、带宽成本和非法行为（见表2.1）。

表2.1 物理结构上的分类及其性质上的差异

C/S P2P

延迟（所需时间、通信延迟） 大小

服务器设备成本大小

带宽成本根据游戏内容而定

非法行为困难容易

可以看出两种结构各有优劣，C/S模型的优势在于效果，由于其将大部分的计算放在服务端进行，所以可以屏蔽大部分的非法行为，但是同时，由于服务器需要承载大量的计算与通信任务，所以较之P2P模型，所需的成本大大增加了。

逻辑结构指的是“实际进行游戏的玩家之间存在怎样的关系”，其两种模式分别为MMO和MO，它们拥有截然不同的游戏体验。

MMO（Massively Multiplayer Online）指大型多人在线游戏，其目的是要大量玩家长期累积游戏。因为要同时处理大量玩家，所以响应时间必然会延长，然后又要供玩家长期进行游戏，所以会有大量的数据。

MO（Multi-player Online）指多人在线游戏，这类游戏的目的通常是让少数玩家在短时间内进行实时的对战，所以这类游戏追求的是高速的操作，需要能与之相匹配的响应时间。由于游戏是在短时间内实时进行的，所以要处理的数据与MMO游戏相比微不足道。

这两种游戏在内容上通常有很大的差异，同时导致了其特性的不同（见表2.2）：

表2.2 逻辑结构上的分类及其性质上的差异

MO MMO

游戏核心少量玩家聚集在一起竞赛大量玩家进行社交活动

同时玩家数 20左右 200-1000000

延迟 50毫秒 300毫秒

游戏时间（累积） 几分钟几年

RMT（真实货币交易） 很少活跃

平台游戏机 PC、移动设备

物理架构与逻辑架构各自独立，所以组合起来总共有4种形式。每种形式具体的应用见表2.3：

表2.3 网络游戏的4种形式及游戏类型

MO MMO

C/S 休闲游戏 MMORPG虚拟世界、大战

P2P ARPG（动作类RPG）、对战格斗游戏、FPS、竞速游戏、RTS ×

回合制RPG网络游戏

角色扮演游戏（Role-playing game）简称为RPG，是游戏类型的一种。在游戏中，玩家负责扮演某个角色在一个写实或虚构世界中活动，并在一个结构化的规则下，通过一些行动指令，令其所扮演的角色发展。

回合制是一种游戏打怪的形式，所有游戏内玩家轮流拥有自己的回合。只有轮到自己的回合，才能够进行操作。早期由于硬件的运算能力有限，在考量游戏乐趣与操作简易的情况下，多半采取这种模式。回合制游戏节奏较慢，玩家可以有大把的时间用来聊天。回合制游戏操作简单，可以加入各种复杂的系统，玩家可以同时操作数个角色。回合制游戏的PK较为轻松，玩家把战斗当做是一种娱乐。

对于一款回合制RPG网络游戏来说，首先它应当是一款以MMO为主，附带有少量MO内容的网络游戏，以MMO为主是因为大量的玩家都在同一个游戏世界中进行游戏，所以符合MMO游戏的特定，但是战斗却是少数玩家存在于独立的游戏副本中进行的，与其他玩家均没有交互，所以战斗系统是符合MO游戏特点的，那么在选择物理架构时，就应当将战斗系统与游戏的其他模块分开来看。

游戏的主要物理架构，毋庸置疑应使用与MMO游戏特性相符的C/S架构，但是战斗模块应该独立使用P2P模型还是与其他模块一起使用C/S模型还需要仔细考虑一下。对比两种物理模型的特性差异，可以进行如下的分析：

首先是延迟方面，战斗的方式为回合制，玩家的操作都集中在回合开始的时候，不需要在整个战斗中都有大量的高速操作，所以C/S模型和P2P模型都能满足游戏要求。

然后是服务器的成本，包括计算成本和带宽成本，C/S架构将所有的计算都集中在服务器上，这样增加了服务器的计算量，并且与客户端之间会有大量的通信，而P2P模型将计算都交由客户端来处理，通信主要存在于客户端之间，与服务器只需

要传输少量的结果数据即可，所以要略优于C/S模型。

最后在非法行为方面，对于网络游戏来说，这是最为关键的一点，如果游戏不再公平，那么这个游戏的玩家必定会越来越少。战斗时要防止玩家作弊，关键的地方在于，不能信任任何来自于客户端的数据。在这样的思考下，如果使用P2P模型，将所有的计算交由客户端来进行，服务器只被动接收结果数据的话，几乎没有办法校验客户端的计算是否合法，玩家是否擅自改动了客户端的数据，即使是使用同时战斗的多名玩家相互校验的方法，也会存在一定的隐患，并且会使得开发难度成倍增加，在没有对战斗计算和通信的比重进行估算的情况下，这种优化很可能是得不偿失的。

综合以上几方面优缺点的比较，对于回合制MMORPG游戏来说，应该尽量采用C/S架构模型来实现。

对于一个C/S架构模型的网络游戏来说，功能的实现将根据实际情况分布在客户端和服务端上。一般来说，客户端将负责所有的表现逻辑，服务端将负责所有的计算逻辑。但是这一点并不绝对，由于某些功能的计算量较大，并且不需要过多的关注其中可能存在的非法行为，所以开发时通常会选择将这些计算任务交给客户端来进行，以减少服务器的计算量。

本次开发中也会按照这个原则，将所有的表现逻辑放在客户端中，大部分的计算任务由服务器来完成，客户端只负责少部分的逻辑计算，如角色的自动寻路，聊天系统的文本转换等。

Cocos2d-x游戏引擎

所谓游戏引擎是指一些已经开发好的提供编写游戏的工具的集合，他们可以让游戏开发者能够快速和容易地做出游戏程序，而不用从零开始。目前市面上主流的游戏引擎有Unity，OGRE，Irrlicht，Panda3D，Crystal Space，JME，Cocos2d-x等，其中部分是收费的商业引擎。

本次开发的是一款纯2D游戏，使用2D美术资源，所以选择了主要用于2D游戏开发的Cocos2d-x游戏引擎。

Cocos2d-x游戏引擎是一个基于MIT协议的开源框架，用于构建游戏、应用程序和其他图形界面交互应用。引擎核心采用C++编写，提供C++、Lua、JavaScript三种编程语言接口。引擎中提供了图形渲染、GUI、音频、网络、物理、用户输入等丰富的功能。Cocos2d-x 适配 iOS, Android, HTML5， PC Windows 和 macOS X 系统，功能侧重在手机原生和HTML5 两大领域，并积极向 3D 领域延伸扩展。截止 2017 年底，Cocos2d-x 在全球拥有超过100万注册开发者，在中国市场占有率45%，全球市场占有率 18%，是中国第一、全球第二的手机游戏引擎。

除了开源的游戏引擎外，Cocos2d还附带有Cocos Creator编辑器，用于游戏场景的搭建与UI的设计。

在本次游戏客户端的开发中，将使用Cocos Creator搭建所有的场景，使用Cocos2d-x引擎来进行场景内的所有逻辑处理。

Cocos2d-x游戏引擎采用节点树来管理游戏对象。它把整个游戏划分为多个不同的场景，每个不同的场景再分为不同的层，每一个层可以拥有任意数量的节点，每个节点可以是任意的游戏物体。引擎中有以下几个关键概念：

导演（Director）：Cocos2d-x中把统筹游戏的类抽象为导演，是整个游戏引擎的核心，游戏中系统的控制都是由导演来完成。

场景（Scene）：场景是Cocos2d-x引擎中不可缺少的一部分，通常游戏中都会构建多个游戏场景来用于页面的切换，比如启动页面到登录界面，再到游戏场景。通常可以通过切换不同的场景来控制游戏的流程。

层（Layer）：层，类似于图像处理中的图层，一个游戏场景中一般会使用多个层来显示不同的东西，比如背景为一层，人物为一层，UI为一层。通过分层来对场景内的游戏物体进行分类，然后对同一类的物体可以统一添加某些相同的逻辑，比如某一层可以优先接收到用户的点击并进行处理。

节点（Node）：节点是Cocos2d-x引擎中的基本单元，在游戏中任何一个对象都是节点，包括按钮，人物，窗口甚至场景，层等等，有些是可视的物体，有些是抽象的逻辑概念。通过控制可视节点的移动，旋转，缩放等可以在游戏中实现复杂的显示效果，通过逻辑节点来实现比较复杂的表现逻辑。

本章小结

本章介绍了游戏正式开发之前所需要的一些准备工作。包括平台和编程语言的选择，几个常用网络游戏架构的介绍，以及本次开发所使用的架构模式和游戏引擎的选择。

指　标
疑似剽窃文字表述

1. Microsoft Visual Studio简称VS，是美国微软公司的开发工具包系列产品，
2. 在游戏中，玩家负责扮演某个角色在一个写实或虚构世界中活动，并在一个结构化的规则下，通过一些行动指令，令其所扮演的角色发展。
3. 游戏打怪的形式，所有游戏内玩家轮流拥有自己的回合。只有轮到自己的回合，才能够进行操作。早期由于硬件的运算能力有限，在考量游戏乐趣与操作简易的情况下，多半采取这种模式。回合制游戏节奏较慢，玩家可以有大把的时间用来聊天。回合制游戏操作简单，可以加入各种复杂的系统，玩家可以同时操作数个角色。回合制游戏的PK较为轻松，玩家把战斗当做是一种娱乐。
4. Cocos2d-x游戏引擎是一个基于MIT协议的开源框架，用于构建游戏、应用程序和其他图形界面交互应用。引擎核心采

用C++编写，提供C++、Lua、JavaScript三种编程语言接口。引擎中提供了图形渲染、GUI、音频、网络、物理、用户输入等丰富的功能。Cocos2d-x 适配 iOS, Android, HTML5， PC Windows 和 macOS X 系统，功能侧重在手机原生和 HTML5 两大领域，并积极向 3D 领域延伸扩展。截止 2017 年底，Cocos2d-x 在全球拥有超过100万注册开发者，在中国市场占有率 45%，全球市场占有率 18%，是中国第一、全球第二的手机游戏引擎。

除了开源的游戏引擎外，Cocos2

5. Cocos2d-x游戏引擎采用节点树来管理游戏对象。它把整个游戏划分为多个不同的场景，每个不同的场景再分为不同的层，每一个层可以拥有任意数量的节点，

脚注和尾注

1. C/S模型，即客户/服务器模型，由服务器进行计算，客户机用于交互和展示计算的结果。.
2. P2P模型，即对等式网络模型，在游戏开发中，指没有专门的服务器进行计算，所有的逻辑运算都由客户机来完成，然后客户机之间再同步计算结果同步显示。.

5. 53140133_初芯宇_计算机科学与技术_基于cocos2d的一项RPG网络游戏软件设计与实现_第5部分

总字数：9464

相似文献列表　文字复制比：2%(188)　疑似剽窃观点：(0)

1	liyg -《大学生论文联合比对库》- 2014-06-12	1.8%（173） 是否引证：否
2	张瑞鹏+2012236654+吴勇翀+基于TCPIP通信聊天室的设计与开发 -《大学生论文联合比对库》- 2016-04-01	1.8%（173） 是否引证：否
3	基于Hadoop平台的网络数据的VoIP流量统计系统设计与实现 刘慧芳 -《大学生论文联合比对库》- 2015-06-03	1.6%（154） 是否引证：否
4	02240110113-刘慧芳-基于Hadoop平台的网络数据的VoIP流量统计系统设计与实现 刘慧芳 -《大学生论文联合比对库》- 2015-06-17	1.6%（154） 是否引证：否
5	基于Socket的局域网络通信软件开发 郭锋;刘建伟; -《电子科技》- 2009-05-15	1.5%（144） 是否引证：否
6	IP城域网综合接入认证系统的分析与设计 赵岩; -《计算机光盘软件与应用》- 2012-06-23	1.5%（142） 是否引证：否
7	计算机网络连接监控系统 - 豆丁网 -《互联网文档资源（http://www.docin.com）》- 2016	1.5%（141） 是否引证：否
8	电子数据类型化及其真实性判断 何文燕;张庆霖; -《湘潭大学学报(哲学社会科学版)》- 2013-03-15	1.5%（141） 是否引证：否
9	运用WANem测试微机保护的通信协议栈及网络性能测试 管金酉;胡艳茹; -《电子技术与软件工程》- 2013-12-15	1.5%（141） 是否引证：否
10	卫星数据接收站中网络协议的应用 王毳; -《无线电工程》- 2009-04-05	1.5%（141） 是否引证：否
11	10kV开关柜无源无线温度监测系统研究与应用 苏乐平(导师：余涛;彭刚) -《华南理工大学博士论文》- 2015-04-24	1.5%（141） 是否引证：否
12	DCS远程专家指导系统中图像传输的研究与实现 刘伟娜(导师：潘卫华) -《华北电力大学博士论文》- 2015-03-01	1.5%（141） 是否引证：否
13	10031501 崔世航 王克朝 崔世航 -《大学生论文联合比对库》- 2014-05-29	1.5%（141） 是否引证：否
14	崔世航_10031501_王克朝_Linux局域网即时通讯软件设计与实现 崔世航 -《大学生论文联合比对库》- 2014-06-10	1.5%（141） 是否引证：否
15	王双喜毕业论文-----Linux平台下的聊天软件的设计与开发 -《大学生论文联合比对库》- 2014-05-30	1.5%（141） 是否引证：否
16	王双喜毕业论文-----Linux平台下的聊天软件的设计与开发 388 王双喜 -《大学生论文联合比对库》- 2014-06-04	1.5%（141） 是否引证：否
17	基于linux的QQ聊天系统及其软件设计 王双喜 -《大学生论文联合比对库》- 2014-06-05	1.5%（141） 是否引证：否
18	局域网监控系统的设计与实现 浦海霞 -《大学生论文联合比对库》- 2015-05-22	1.5%（141） 是否引证：否
19	基于zigbee和以太网传输功能的主单元设计 王昭阳 -《大学生论文联合比对库》- 2016-05-15	1.5%（140） 是否引证：否
20	（连载）华为笔试试题(1)ip，tcp,udp_小金三 -《网络（http://blog.sina.com）》- 2012	1.5%（139） 是否引证：否

第3章关键技术

除了整体方案的选择，在游戏开发中还使用了一些关键的技术。

通信中间件

相较于单机游戏，网络游戏最大的特点就在于实现功能的逻辑中嵌入了网络通信，这使得客户端和服务端之间的交互变得十分繁琐与复杂。

为了使这种复杂的模型简单一点，所以在网络游戏的开发中，往往需要选择一个通信中间件来提供通信功能，一方面屏蔽网络通信的底层操作，在正式开发中更加方便快捷。另一方面可以对不同系统的通信进行解耦，使得不同系统的开发耦合度降低，减小开发难度。在本次开发中，通信中间件分为两个部分，一个用于客户端，一个用于服务端，均实现了网络连接与数据包的收发功能。两个部分分别被打包成lib文件（静态链接库），然后在对应项目中使用。

TCP提供IP环境下的数据可靠传输，它提供的服务包括数据流传送、可靠性、有效流控、全双工操作和多路复用。通过面向连接、端到端和可靠的数据包发送。而UDP则不为IP提供可靠性、流控或差错恢复功能。一般来说，TCP对应的是可靠性要求高的应用，而UDP对应的则是可靠性要求低、传输经济的应用。因网络游戏通常对于数据的准确性有相当高的要求，所以大部分的游戏开发都会选择TCP而不是UDP。

客户端和服务器两部分的通信逻辑相似，均从配置文件中读取信息，然后使用TCP进行数据收发，只是在初始化和连接数上有一定的区别。

服务端通信组件

服务器的通信初始化，需要IP和端口号才能开始接收客户端的连接请求。为了限制服务器的吞吐量，还需要设置服务端的最大监听数、最大连接数和数据包大小限制。所以服务端的通信配置文件如下：

```
[Config]
LocalIP=localhost
ListenPort=8080
MaxListen=100
MaxConnect=1024
MaxRecvPackSize=1024
```

当完成通信组件的初始化之后，就使用该IP和端口号开始监听客户端的连接请求，每连接一个新的客户端就会创建一个新的Socket放入队列中进行通信。

由于服务端的连接数有限，所以会对所有的Socket进行管理。当连接数达到配置的上限时，服务器就达到了负载的上限，不会再接入其他的客户端，所以对于每个Socket都设有一个值保存最近的通信时间，当一个Socket很长时间没有进行通信时，就会判断其已经失活，并进行相应的处理。

客户端通信组件

客户端的通信要比服务端简单一点，只需要保持与单一服务器的连接即可。所以在配置中我们需要服务器的IP和端口号来进行连接，然后同样地设置数据包的大小上限，额外还需要配置连接的超时时间以防止无法连接上服务器的情况。所以客户端的配置文件如下：

```
[Config]
ServerIP= localhost
ServerPort=8080
ConnectTimeout=3000
MaxRecvPackSize=1024
```

客户端的初始化就是使用该IP和端口号连接上服务端。除了基本的数据收发外，为了防止长时间没有进行通信而连接失活，每隔一段时间都会向服务端发送一个固定的请求来更新最近通信时间。

通信机制

通信层使用不同的处理器来处理接收到的数据包，处理逻辑由功能模块各自提供并在通信层中注册，注册时需要为这个

处理器分配一个独特的ID作为辨识标准。

其他模块可以使用通信层传输一个结构体对象，这个结构体中需要包含一个ID来指明对应的处理器。当通信层收到一个结构体对象时，会根据这个ID来查找到对应的处理器进行处理。这样的设计使得所有的通信相互独立，互不干扰。

通信协议

网络游戏开发中，最核心的工作即是设计客户端和服务端之间的通信协议，通信协议决定了客户端和服务端的交互内容和交互时机，因而也确定了客户端和服务端程序各自的运行流程，只有确定了通信协议之后才能进行客户端和服务端的系统设计。因此首先设计了登录通信协议（如表3.1所示）、移动通信协议（如表3.2所示）、战斗通信协议（如表3.3所示）、聊天通信协议（如表3.4所示）。

表3.1 登录通信协议

接口名发送方发送时机功能说明

LoginMsg 客户端用户选择登录的时候向服务端发送用户名和密码进行验证登录

rLoginMsg 服务端验证完用户登录之后返回用户的登录结果，并返回角色列表

LogoutMsg 客户端用户正常登出的时候通知服务端某个用户登出

OfflineMsg 服务端有用户下线的时候通知相关客户端该玩家从视野中消失

RoleMsg 客户端客户端选择角色或者创建角色时客户端请求选择角色或者创建角色

rRoleMsg 服务端客户端选择完角色之后返回选择角色的位置信息

表3.2 移动通信协议

接口名发送方发送时机功能说明

LocationMsg 服务端当服务器需要客户端同步位置的时候发送某个角色的位置信息，客户端无条件同步该位置

MoveMsg 客户端当角色开始移动时发送角色将要行走的路线给服务器

MoveInfoMsg 服务端当有角色开始移动的时候通知相关的客户端某个角色的移动路线

SyncLocationMsg 客户端当角色需要同步位置时向服务器请求一次位置同步

TPMsg 客户端切换地图时指明要切换到的地图以及要切换到的坐标

表3.3 战斗通信协议

接口名发送方发送时机功能说明

BattleData 服务端战斗开始时本场战斗中所有角色的信息

RoundStart 客户端当新回合开始的时候通知服务端开始下一回合

PlayerAction 客户端当玩家选择了操作之后发送玩家选择的操作

BattleMsgBase 服务端当战斗中发生任意情况的时候战斗事件基类，具体内容根据情况而定

RoundEnd 服务端当一回合的所有操作执行完之后通知客户端本回合已经结束，并附带战斗是否结束的标识。

表3.4 聊天通信协议

接口名发送方发送时机功能说明

ChatMsg 客户端发言时包含发言内容、发言频道和发言人的信息

ChatMsg 服务端有玩家发言时包含发言内容、发言频道和发言人的信息

ChatPrivateMsg 客户端私聊发言时包含发言内容、私聊对象名和发言人的信息

ChatPrivateMsg 服务端有玩家私聊时包含发言内容和发言人的信息

ChatReturnMsg 服务端当发言失败时失败的原因

可复用ID

游戏中很多地方都需要临时分配Id，比如玩家登陆之后为其分配的临时Id，或者生成装备和道具时要为其分配一个独一无二的Id，这个Id的分配会比较频繁，并且在整个游戏的运行中会一直进行，所以如果Id不可复用的话，那么迟早会将所有Id都分配出去而导致系统崩溃。为了解决这个问题，可以使用一套可回收Id的系统来对已经不再使用的Id进行复用，尽可能保证Id不会用尽。

这套系统类似于内存的管理机制，使用位图的方式来对Id进行映射管理。申请好一段连续的内存，然后用这段内存中的每一位来映射一个不同Id，这样就可以用少量的内存来管理大量的Id。整个系统中可以有很多段这样的内存，并且可以动态申请以无限扩展Id的数量。当需要一个新的Id时，从所有的内存中去查找一个空闲的位，找到之后通过计算找到对应的数字Id，然后将该位置为不可用即可。当一个Id使用完毕之后，也找到对应的位，将该位置为空闲状态等待下一次分配。该系统满足了ID复用的基本要求，可以快速取出一个空闲ID，快速归还一个使用完毕的ID。

寻路算法

寻路是游戏开发中一个很经典的问题，为了使用数学模型来进行寻路，通常会将不规则的地图划分为一个一个细小的格子，在最基础的方格地图上使用寻路算法进行计算。

目前比较常用的寻路算法有深度优先搜索、广度优先搜索、Dijkstra 算法、贪婪最佳优先搜索和A*算法。

深度优先搜索（DFS）用遍历的方式查找一条从起点通往终点的路径，并不能找到去往终点的最短路径，所以不考虑。

广度优先搜索（BFS），是一种盲目搜寻法，它从起点系统地展开整个地图，直到找到终点。由于是逐渐展开的，所以

可以保证找到的路径一定的最优的，但是需要将所有的顶点都遍历一遍，效率十分低下。

Dijkstra 算法基于广度优先算法，但是在遍历时可以代入每条边的权值，每次遍历时都可以计算从起点到每个点的代价，然后从代价最小的点开始遍历，在寻路时带有一定的方向性，但是依然是在盲目搜索，没有解决效率低下的问题。

贪婪最佳优先算法类似于Dijkstra算法，但它排序的基础不是从起点到每个点的代价，而是估算出每个点到终点的代价，使用该值来进行下一个点的选择，但是这个算法有着和贪心算法一样的缺点，只能最快地找到一个较优解，不能得到最优解。

A*汲取了上述寻路算法的优点，它在每次遍历时，计算出起点到该点的代价，并且估算出该点到终点的代价，将两者相加，以此作为选择下一个点的依据，这样既有方向性地快速寻找出一条路径，并且可以保证该路线是最优的。对点的评估算法叫做启发函数，所以AStar寻路是一种启发式的算法。通常来说，启发函数的计算过程，都是从起始点到该点的距离加上该点到终点的代价，这个代价的计算方式有很多，并且会影响AStar找到最优路径的速度。在本次开发中使用欧几里德距离作为点到终点的代价。

在寻路过程中，希望得到的路径点是连续且平滑的，故在对路径评估的函数中加入转弯干扰值，以减少所得路径过多的转弯。

移动方案

移动是由玩家触发的行为，当玩家点击地图上某个点时，客户端会计算出相应的路线并且移动。

这里首先涉及一个问题，移动交由谁来计算？这个问题涉及两个方面的因素：通信频率和安全性。对于这个问题，有如下两种方案，即客户端和服务器之一进行计算。

如果由客户端来计算，那么数据的传输就应该是客户端实时地向服务端发送自己当前的坐标，服务端改变了坐标之后再同步给其他的客户端。这种方案下，由于客户端的位置一直在改变，所以需要每一帧都向服务端发送一次当前的位置，那么通信的频率将会是每帧一次，即使进行一定的优化处理，也需要每隔几帧就同步一次位置信息，而且这种优化还很可能会导致其他的客户端看到的角色时常闪烁。在安全性上，服务端几乎无法判断客户端发来的位置是否正确，只能被动地改变，那么这个数据将极不安全。所以这种方案几乎不可行。

如果由服务端来计算的情况，客户端向服务端发送将要行走的路线，服务端根据路线计算位置，并通知所有的客户端更新人物位置。这种方案，客户端只需要向服务端发送路线即可，但是服务端还是需要每帧或者每隔几帧就向所有客户端发送一次位置信息，通信量依然很大。在安全性上，由于客户端通知服务端的只是路线，速度和是否可以行走都是由服务端来判断，而路线是可以进行校验的，所以在安全性上有了大大的提升。

综上所述，将计算交给服务端似乎要比在客户端上进行好得多，在安全性上有了很大的提升。但是在通信上似乎没有多大的改善，因为都是需要一端计算然后每一帧都通知另一端。那么，可不可以两端同时进行计算呢？

于是，提出了第三种方案：

只要服务端和客户端各自计算就可以减少同步所需的通信量。那么，所需要传输的就只是移动的路线，以及一些特殊状态的信息，比如路线改变，或者遇到了障碍物等。

虽然基本的移动操作实现并不复杂，但是游戏的复杂性通常在于需要处理很多特殊的情况，比如玩家改变了正在移动的路线，又或者客户端传来的移动路径上存在障碍物。当移动路线发生改变时，客户端首先需要重新计算路线，然后将重新计算后的路线发给服务器以完成整个路线的变更，而后服务器再将路线的变更通知给其他所有客户端。

首先，当客户端重新计算完路线，将路线发给服务器时，这时候服务器中的角色位置几乎不可能与客户端重新寻路时的位置相同，所以这个时候需要做一些特殊处理来修改角色的移动路线。

在每一条移动线段中增加了一个标志位，表明该线段是否是一次寻路的第一条线段，以此来区分重新寻路的情况。

对一次移动路线的每条线段都进行了编号，重新的寻路的时候可以通过编号来快速查找到重新寻路的起始点。

在每次重新寻路时，都在新的路线前加上从当前位置到重新寻路点的路径，以此来同步客户端和服务器的位置。

其余的客户端也执行了和服务端差不多的操作，以保证在重新寻路时路线更加平滑而不是直接将人物位置设置到重新寻路的起始点上。

其次，正常情况下，由于客户端和服务器地图相同，所以客户端寻找出来的路线不会遇到障碍物。但是，有可能在寻路完成之后地图发生了改变，本来没有障碍物的地方生成了障碍物，又或者客户端发送的路线本身就存在问题，导致服务器中角色行走时遇到了障碍物，这个时候会选择将角色停顿在上一个位置，并且通知所有客户端将角色拉回该位置。

战斗AI

回合制游戏的战斗AI内容简单，因为战场形式很容易数据化，只需要根据设计好的职业AI内容进行判断和决策即可。但是回合制游戏中玩家操纵的角色应当由AI来决定动作还是由玩家来指定操作是不确定的，所以为了统一代码接口，使用一个抽象类Decisioner来为每个Battler决定本回合要执行的动作。Decisioner有两个派生类，一个是PlayerDecisioner使用玩家选择的操作，一个是AutoDecisioner使用AI来自动选择操作。然后AutoDecisioner下面再派生出各个门派各自的AI决策器（如图3.1）。

图3.1 战斗AI类图

初始的时候，所有角色的决策器均是AutoDecisioner，当玩家选择了操作，就把该Battler的决策器替换为PlayerDecisioner，这样的话在战斗计算前就只需要统一调用Decisioner的Execute接口了，不需要再去关心玩家是否指定了角

色的战斗指令。然后在每一回合结束时，再把所有玩家角色的决策器替换为相应的AutoDecisioner。

采用这种工厂模式搭配策略模式的方案，好处在于，战场的回合计算过程中，将关注点全部放在了流程的进行上，完全忽略了角色的门派或者玩家是否操作所带来的差异，只需要调用统一的流程接口往下执行即可。除了在AI模块，在后面的技能执行过程中也用了类似的方式来削弱角色之间的差异对整体流程带来的影响。

对象池

在游戏开发中，有的模块会涉及到大量对象的创建和删除，比如射击游戏中的子弹，战争游戏中的小兵等等，这样大量对象的创建和删除会导致性能的降低，这个问题通常会使用对象池来解决。

对象池，顾名思义，将所有的对象存储在一个池子里，当需要的时候取一个出来使用，不需要的时候就放回去，而不是每次使用都实例化一个新的对象，每次使用完都将其析构掉，这样子一方面减少了new和delete操作带来的损耗，一方面更加容易发现内存泄漏的问题。

批渲染

要解释批渲染首先得说明DrawCall这个概念。

图像引擎生成一帧画面的处理过程，简单来说有几步：

1. 经过简单的可见性测试，确定哪些物体在摄像机的视野内，只需要渲染这部分在视野内的物体。

2. 将这些物体的顶点以及法线、变换等数据处理准备好。

3. 使用图形API，通知GPU使用上述数据进行绘制。

DrawCall即是通知一次GPU的过程。如果不同的物体使用不同的纹理和材质就需要调用两次DrawCall，而每次调用DrawCall都需要很多准备工作，比如检测渲染状态、提交渲染数据等等，所以会有很多额外的开销。那么将大量的精灵在DrawCall中进行渲染就可以很大程度上减少渲染的开销。

所以对于大量不同物体的渲染，最好能将所有的图像绘制在同一张纹理上，然后使用这一张纹理来同时渲染这些物体，这样子就可以在同一次DrawCall中绘制完成，可以节省很多不必要的开销。

视野划分

在一款RPG游戏中，通常大地图中的人都是很多的，如果要每两个人都互相知道对方的信息的话，那需要的传输量是相当大的，所以通常我们都会将地图按照视野进行划分来优化，以减少数据的传输。

首先把一个屏幕大小的区域称为一块地图，假设大地图很大，约有100块地图，每块地图平均有10人，那么整个大地图里就有1000人，当一个角色移动的时候，需要向所有1000人发送移动的信息，也就是发送1000次。但是，真的有必要向所有的1000人都发通知吗？实际上并没有这个必要，因为能看见这个角色的，也就只有这个角色所在周围共9块地图的角色，那么，我们实际上只需要向这9块地图里的角色发送该角色的移动信息就可以了，发送信息的次数就从1000次减少到了90次，到了100：9的比例，当地图的大小扩大时，这个比例还会继续增大。

所以，在服务端，地图被划分为多个视野块，每个视野块的大小略大于一个屏幕所能显示的区域大小（这是为了容许正常的网络延迟所带来的误差），然后所有角色的信息都只会向其周围共9个视野块的角色发送，当玩家跨越视野块的时候又会获取到新视野块中的角色信息。

技能配置表

一款商业游戏与一款独立游戏，其中一个很重要的区别就在于，商业游戏的开发过程中分工明确，策划者与程序开发人员独立工作，那么开发人员就要考虑到如何能让游戏策划人员在后期轻松地反复修改游戏内容和数据以进行尝试。

而如果要让非程序开发人员能够轻松地修改游戏内容，那么最常用的方式就是通过修改配置表来实现修改游戏内容和数据的目的。游戏数据的配置表可以很容易地设计出来，但是游戏内容的配置表相对来说就比较抽象，要以配置表的形式修改游戏内容就较为困难。比如各个门派的角色技能，在开发后期可能由于设计原因需要经常修改，反复尝试，所以不能以硬编码的形式写在代码中。

由于客户端和服务端对技能的关注点不同，所以使用了不同的配置表，服务端的配置表如图3.2和图3.3，客户端的配置表如图3.4。

图3.2 服务端技能配置表

图3.3 服务端技能配置表

图3.4 客户端技能配置表

服务端配置表记录的是技能的一些基础数据：ID、名字、消耗、可以作用于己方还是敌方、作用者是否需要存活、是否可以作用于自身、作用的人数。这些是技能的基础属性，用于技能的映射和施法校验。

然后记录的是技能的效果列表：效果数以及每个效果的类型、作用概率、作用人数、数值属性和数值。效果的类型包括Buff类型、状态类型、攻击、治疗、Buff或状态的清除，通过编号段来划分；数值属性对于不同的效果类型有不同的意义，对于攻击来说表示攻击属性（真实伤害、物理、魔法），对于Buff和状态表示回合数，对于清除Buff或状态，表示Buff或状态的ID。

目前技能效果总共有5种：添加Buff、设置状态、攻击、治疗和清除Buff或状态。每种效果都有各自的初始化和执行函数，以便产生不同的效果。

客户端配置表主要包含技能的三部分：

基本数据，包括名字和ID。主要是用于设置UI的展示内容，显示给玩家以便查看技能。

施法属性，包括是否选择目标和选择目标的限制，主要是用于玩家操作阶段，协助设置状态机的跳转情况，以在玩家选择技能时进行限制。

演示属性，该技能的基本演示序列，是客户端战斗流程演示的核心。演示属性我使用的是微动作序列，类似PowerPoint的动画系统，通过一系列的微动作连接演示以达到流畅展示整个动作的目的，每个微动作我使用四个属性来设定：

第一个属性指定微动作执行的时机，和上一个动作同时发生或者在上一个动作执行完成后进行。

第二个属性指定微动作类型，分别有移动、施法、攻击、收到伤害、收到治疗、复活、设置Buff或状态共7种类型。

第三个属性指定微动作的执行者，因为在一个技能的施放通常都会涉及到施法者和被施法者，这个属性用来指定该微动作由施法者来执行还是由被施法者来执行。

第四个属性是辅助位，在某些动作需要一些额外信息的时候通过辅助位来设置，比如移动时要移动到的位置。

两个配置表中相同技能的配置项共同决定了一个技能设计，包括其基础数据、施法效果和动画表现。

技能动画

技能这种复杂多变设计的表现逻辑对于游戏来说一直是一个难点，因为在开发时不知道之后的策划会是什么样的，还会新增哪些技能，甚至是新的设定，所以在技能的表现上，需要有充足的可扩展性。

对于完整的战斗流程演示，我选择的方法是：先根据服务端通知的动作ID查找到对应的动作，然后获取动作相应的展示序列，再根据动作的展示序列来生成相应的微动作对象，并按照每个微动作的顺序和执行的时机安置好整个微动作序列，后面即可直接按照序列演示。

微动作序列是战斗演示的核心，但是难点却在于额外动作的发生。比如援护、暴击、闪避等概率事件的发生，又或者是二次攻击、吸血等附加效果的触发，都是需要由服务端来统一计算，然后客户端根据服务器的通知来进行演示。困难之处在于，客户端不知道什么时候会触发哪一个或者哪一些事件，并且未来还可能添加很多类似的事件，那么，如何来应付这种随机的触发，并且能提供足够的扩展性，就成了解决问题的关键。

思考这个问题时，首先是，在基于已经确定好的微动作序列方案中，怎么样才能展示出这些突发的效果呢？就以普通攻击时触发了援护动作为例吧。正常的普通攻击动作序列应该是：

1. 施法者移动到被施法者面前。
2. 施法者攻击。
3. 被施法者受伤。
4. 施法者回到原位。

总共四个步骤，然后在有援护触发的时候，动作的序列应当如下：

1. 施法者移动到被施法者面前。
2. 援护者移动到被施法者面前。
3. 施法者攻击。
4. 被施法者受伤。
5. 援护者受伤。
6. 援护者回到原位。
7. 攻击者回到原位。

可以看出，触发援护时的序列和原本的序列很相似，仅仅是在施法者攻击微动作的前后添加了援护者的三个动作：移动到被施法者面前、受伤、回到原位。类似于援护，闪避的动作序列也仅仅是把被施法者的受伤动作替换为闪避动作即可，除此之外，还有暴击、二次攻击等等，也都可以通过按照一定的规则来修改原有的序列以得到新的演示序列。那么一个解决方案就基本成型了：首先生成出动作的基本序列，然后按照顺序，让额外的动作依次来对已有的序列进行调整以生成新的演示序列即可。在有新的额外动作的时候，只需要添加新额外动作的序列调整规则即可，然后就可以按照既定的流程进行演示。

根据上述的方案，设计出了如图3.4的方案：

使用一个微动作队列来存放并行执行的所有微动作，然后再用一个队列来存放生成的所有并行执行的微动作队列。这个队列由微动作代理来生成、管理、执行。

整个程序的执行过程如下：

首先查找动作列表里是否有行为包，如果没有则直接结束。

将行为包之前的所有额外信息全部保存下来。

使用行为包生成一个微动作序列。

再用之前保存的额外信息来挨个调整动作序列以展示所有事件。

删除已使用的信息。

然后开始循环执行微动作序列中每一组并行执行的微动作。

执行完之后再执行下一组，直到队列为空。

图3.4 客户端技能配置表

本章小结

本章介绍了本次开发中所使用到的一些关键技术点。包括一些实现问题和一些优化问题的解决方案。

指　标
疑似剽窃文字表述

1. TCP提供IP环境下的数据可靠传输，它提供的服务包括数据流传送、可靠性、有效流控、全双工操作和多路复用。通过面向连接、端到端和可靠的数据包发送。而UDP则不为IP提供可靠性、流控或差错恢复功能。一般来说，TCP对应的是可靠性要求高的应用，而UDP对应的则是可靠性要求低、传输经济的应用。因网络游戏通常对于数据的

脚注和尾注

1. 欧几里德距离即是使用勾股定理算出的两点间的直线距离。.

6. 53140133_初芯宇_计算机科学与技术_基于cocos2d的一项RPG网络游戏软件设计与实现_第6部分

总字数：4650

相似文献列表　文字复制比：0%(0)　疑似剽窃观点：(0)

原文内容　红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容

第4章服务端

游戏的开发分为服务端和客户端两个部分，两部分相互协作共同实现功能。首先介绍游戏服务端的开发。

服务端分为通信层、逻辑层、数据层三层来开发，通信层负责与客户端的通信以及数据的打包和解包；逻辑层进行整个游戏的逻辑处理，包括每帧都会运行的游戏本身的逻辑，以及接收到客户端数据之后的处理逻辑；数据层由两块构成，一块管理从配置文件中读取到的数据，一块负责数据库的读写任务。

通信层

通信层的主要目的是将通信与游戏逻辑隔离开，作为通信中间件与游戏服务端之间的桥接工具，接受客户端传来的数据包，进行校验之后调用逻辑层中相应的处理逻辑进行处理。需要向客户端发送数据时，接受逻辑层的调用，将相应的数据打包成规定好的数据包，再发送给指定的客户端。

所以通信层总共有以下几个行为：

1. 初始化通信中间件。
2. 定义所有的数据包与数据包结构Id。
3. 注册所有的处理器，建立起处理器与数据包Id的映射关系。
4. 接受客户端发来的数据调用相应的处理逻辑。
5. 将逻辑层的数据打包发送给指定客户端。

行为1、2、3都由通信层的管理器执行。行为4是通过定义不同的通信处理器，并且在处理器中实现对应的处理函数来实现的，当有数据传来时，通信中间件会取出数据包中的Id，根据之前建立起的处理器与Id的映射关系来找到对应的处理器，并执行其中的处理逻辑。行为5是通过专门的数据发送器，来将相应的数据打包成特定的结构，以起到隔离逻辑和通信结构的作用。

逻辑层

根据游戏的功能点，服务端的逻辑共分为四个部分，分别是玩家管理，移动处理，战斗计算和聊天转发。

玩家管理

这一部分主要负责玩家的登录和登出，为每个玩家进行数据的读取与保存，维护所有在线玩家的列表，建立起玩家Id，角色名与玩家数据的映射关系，以便在需要的时候能快速查找到对应的玩家实例进行处理操作。

移动处理

这一部分主要负责玩家的移动逻辑处理。本次游戏开发的移动方案是客户端在寻路之后，就开始移动角色，同时将角色的移动路线发送到服务端，服务端再将移动路线转发给其他视野范围内存在该玩家的客户端，然后服务端和其他客户端也各自计算该角色的位置信息。这样就不必在移动时传输大量的数据以同步角色的位置信息了。如果服务端发现移动到障碍物时会要求客户端将角色停下来（以防止客户端使用非法的手段快速移动角色）。如果角色移动过程中，玩家改变了行走路线，就将新的路径发送到服务端，这时由于网络延迟的原因，服务端的位置几乎不可能在新路径的起点，所以需要先校验新的路径起点是否合法，如果合法再将角色移动到新路径的起点，之后再沿着新的路径进行移动。

战斗计算

整个服务端的战斗系统计算流程如下：

当战斗开始时，随机为双方分配NPC对象构建战场，并且将角色信息发送给客户端以开启一场战斗。

将构建好的战场添加到战斗管理器中以便每帧都进行更新计算。

战斗的每回合运行

首先，服务端等待客户端，当收到客户端准备完毕的通知之后开始倒计时。

当收到玩家的操作信息时，设置角色操作，并且开始回合。

如果没有收到操作信息，倒计时已经结束，那么直接开始回合。

首先进行战场回合前的处理：

对所有角色按速度进行排序。

对自动战斗的角色进行AI决策，计算出战斗指令。

要进行防御的角色添加防御状态。

然后是战场回合的计算，对每个角色按照顺序进行回合：

首先是单名角色回合前的处理。

执行所有Buff的效果。

更新Buff的回合数。

由于回合前的处理可能会导致角色死亡，所以进行一次战斗结束检测，如果结束直接返回。

然后执行单名角色的操作，操作过程比较复杂，之后在技能模块会详细介绍。

由于角色的操作有可能导致角色死亡，所以此处再次检测战斗是否结束，如果结束直接返回。

然后进行单名角色回合后的处理：

对角色的状态回合数进行更新。

每名角色的回合结束后，检测一次战斗是否结束，如果结束直接返回，如果没有结束则继续下一名角色的回合，直到所有角色都执行完毕。

然后进行站场回合后的处理

撤销所有角色的防御状态。

每次战场回合结束之后判断战斗是否结束，然后给客户端发送回合结束的消息，如果没有结束则返回等待客户端准备完毕，如果结束则直接关闭战场。

战斗的流程较长，但是因为几乎没有变化，所以很简单，依次进行相应的操作即可，比较复杂的是角色技能的执行。

在服务器初始化时会将每个门派的技能（如表4.1）数据读取到内存中，包括技能的ID、消耗、目标限制和技能效果等等，在角色施放技能时，查找到对应的数据，然后对目标使用相应的效果，通过客户端与服务器的通信，同步每一回合的战斗数据，具体协议如表4.2。

表4.1 全职业技能表

技能ID 技能效果

职业1-技能1 物理群体攻击

职业1-技能2 物理单体攻击

职业1-技能3 一定回合内逐渐提升自身攻击力，并且攻击时附带吸血效果

职业1-被动技能提升攻击力

职业2-技能1 法术群体攻击

职业2-技能2 法术单体攻击

职业2-技能3 解除一名角色的控制状态并提升其速度

职业2-被动技能提升暴击几率

职业3-技能1 法术群体攻击

职业3-技能2 单体封印（70%概率）

职业3-技能3 使指定角色一定回合内，进行攻击时可以额外造成一次真实伤害

职业3-被动技能每回合第一次受到攻击时有一定概率进行闪避

职业4-技能1 法术群体攻击

职业4-技能2 最多使三名角色在一定回合内每回合恢复一定生命值

职业4-技能3 使一名角色复活并恢复一定生命值

职业4-被动技能场上存活己方人员生命值百分比越低，技能恢复生命值越高

职业5-技能1 法术群体攻击

职业5-技能2 增加己方一名角色的速度和攻击力

职业5-技能3 增加己方所有角色的防御力

职业5-被动技能和职业6同时在场时提升防御力

职业6-技能1 法术群体攻击

职业6-技能2 降低对方所有人防御力

职业6-技能3 使一名角色进入混乱状态（70%概率），并降低速度

职业6-被动技能和职业5同时在场时提升防御力

表4.2 战斗详细数据的通信协议

接口名发送方发送时机功能说明
Action 服务端某个角色执行完操作之后通知客户端该角色执行的动作与执行的结果
Help 服务端触发援护的时候通知客户端一次援护事件
Dodge 服务端触发闪避的时候通知客户端一次闪避事件
Crit 服务端触发暴击的时候通知客户端一次暴击事件
Defend 服务端开始防御的时候通知客户端某个角色开始防御
BloodSuck 服务端触发吸血的时候通知客户端一次吸血事件
ExtraAttack 服务端触发额外攻击的时候通知客户端一次额外攻击事件
BuffEffect 服务端 Buff生效时通知客户端某个Buff生效
BuffChange 服务端 Buff添加或者清除的时候通知客户端某个角色添加或者清除了某个Buff
StateChange 服务端状态添加或者清除的时候通知客户端某个角色添加或者清除了某个状态
BuffDead 服务端 Buff自然消亡时通知客户端某个Buff过期
StateDead 服务端状态自然消亡时通知客户端某个状态过期
聊天转发
服务端的聊天系统只需要对玩家的发言进行简单限制判断，然后转发即可。
由于每个玩家的发言状态不同，所以为了能够分开进行管理，为每个在线的玩家都分配了一个发言代理，用于角色自己的消息转发和发言限制管理。
所以当服务器收到发言消息的时候，查找到对应的玩家，然后使用发言代理进行转发。发言代理首先根据频道判断是否能进行发言，如果能再获取发言对象列表进行转发，并更新发言状态。
数据层
数据层有两部分功能，分别是存储配置文件中的数据和数据库的读写。
模板数据管理
在游戏中存在很多的模板数据，比如一个职业的角色在创建时的属性点应该为多少，同一件装备在生成时的属性加成是多少，这些数据是由整个游戏所有玩家共用的，并且是永远不可变的，这部分数据一般都被存储在配置文件中，在游戏服务器启动时便读取进内存，在需要的时候就拷贝出一个副本供外部使用。这个功能就在数据层中实现。
数据库读写
游戏中的数据分为两部分，一部分是开发者所配置的游戏基本数据，一部分是玩家逐渐累积的游戏角色数据，基本数据是从配置表中读取入内存中，而游戏角色数据由于玩家数量的庞大，所以需要存储在数据库中。数据层便需要负责这一部分数据的读写。
这次游戏开发使用的是MySQL数据库来保存游戏角色数据，总共使用了两张表，分别是user_data和role_data：
表4.3 用户表（user_data）结构
字段名称数据类型是否可为空约束条件字段说明
user_id int 不可为空 primary key 用户id，自增字段
username varchar(20) 不可为空 primary key 用户名，不可重复
password varchar(35) 不可为空 null 密码
表4.4 角色表（role_data）结构
字段名称数据类型是否可为空约束条件字段说明
rolekeynum int 不可为空 primary key 数据id，自增字段
roleid int 可为空 null 数据归属角色id
user_id int 可为空 null 数据归属用户id
dataname varchar(20) 可为空 null 数据类别
datacontent longblob 可为空 null 数据的二进制表现
在user_data表的数据存储中，为了防止由于特殊情况导致玩家数据泄漏，所以其中的密码部分（password）存储的是对于玩家密码的MD5散列值。在玩家登陆的时候，我们会对玩家输入的密码进行MD5散列，然后判断散列值是否相同。
role_data表是游戏中所有持久化数据的存储表，其中dataname代表存储数据的类别，然后之后的datacontent存储的是该类型对应的数据。这样设计数据表的好处在于，能够很好地应付游戏更新时新增玩家数据或者修改玩家数据结构的情况。当游戏更新时，可能会把某些模块的数据替换成新的结构，这样就会出现更新之前的数据还是使用老结构，而更新之后的数据使用新结构的情况，而这样存储就可以通过数据的类型来区分新老结构，分别进行处理转化。而在新增模块时，更是只需要添加一个新的数据类型即可。
由于数据库的读写是一个阻塞的过程，在高速处理大量玩家数据的服务器中，这种阻塞接口几乎是不可能使用的，所以只能选择使用一个单独的线程来进行数据库操作，然后使用生产者-消费者模式，通过两个队列来进行两个队列之间的交互。
两个队列分别为任务队列和结果队列，主线程在任务队列中插入任务，每个任务包含要执行的SQL操作和要操作的数据，以及执行完成之后的回调函数，在执行完成之后子线程将任务数据和回调函数再插入到结果队列中，由主线程去处理。

本章小结

本章介绍了服务端的基本结构和部分实现细节。

服务端分为通信层、逻辑层和数据层，逻辑层再分为登录、移动、战斗、聊天四大功能。

7. 53140133_初芯宇_计算机科学与技术_基于cocos2d的一项RPG网络游戏软件设计与实现_第7部分

总字数：3353

相似文献列表　文字复制比：1%(32)　疑似剽窃观点：(0)

1	基于cocos2d-x的横版过关游戏的设计与实现 殷曄 - 《大学生论文联合比对库》- 2015-05-19	1.0%（32） 是否引证：否

原文内容　红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容

第5章客户端

客户端最重要的任务就是将计算结果展示给玩家。

UI搭建

对于客户端显示部分的UI，我们采用了Cocos目前的可视化UI编辑软件Cocos Studio完成。

我们先在Cocos Studio中搭建好UI的显示，以及层级关系，并导出相应的文件。之后，我们在代码中构建对应的Layer，把搭建好的UI从导出的文件中读取出来，让其显示到Layer上。完成了构建之后我们就可以在代码中给每一个UI加上其触发的事件。

客户端的登录界面，选角的部分层，以及主场景跟战斗场景的部分层，均采用这种方式来进行。

游戏主场景

游戏主场景分为三层，分别为UI层，人物层，地图层。其顺序为UI层在最上方，然后人物层在UI层之下，地图层之上。这样在显示上的顺序就为UI层，人物层，地图层，让各个层级之间有遮挡关系。

角色模块

角色是游戏中很重要的一部分，其不光包括玩家控制的角色而且包括NPC等存在。

对于角色。我们确定了一个角色层来承载所有的角色并且，设置角色之间的遮挡关系为距离近的角色遮挡之后的角色。

当服务器发过来其他玩家登陆时，在我们本地的显示上也会显示出来对应的登录玩家。这时因为客户端跟服务器二者分布在不同的地方，二者相互之间的执行顺序并非线性结构，所以我们本地接收到其他玩家的登录不一定处于可以构建其他玩家的状态，比如我们的角色层在这时可能未构建完成。这时我们就需要一个添加和删除角色的任务队列来使两个程序同步。在接收到服务端的消息时，客户端不会立即行动，而是把这个消息先缓存下来，等本地可以执行添加或者删除任务的时候，再按顺序执行。

接下来，在角色动画显示的时候我们发现，Cocos2d-x的动画系统是不可以按照时间需求从指定关键帧进行播放动画的，这会导致人物在行走或者战斗的时候，每一次的动画播放必须从头开始播放。譬如，玩家在控制人物行走过程中，如果频繁点击一个地方会导致，人物一直保持在动画的第一个关键帧。发现了这个问题之后，根据该游戏的要求我们自己写了一套人物的动画系统来保证，人物播放动画的连续性。

在资源文件中，我们把所有动画资源打包成正图和plist文件，在程序开始时根据plist中的配置读取对应的整图文件，然后将整图文件解析成一个个对应的分图。然后在动画的配置文件中，将每一个角色的动画要求都写出，如该角色的动画关键帧数，每个关键帧所占用的时间等。接下来我们在为角色建立三个类：XRoleSpriteState，XRoleLogicSprite，XRoleGraphicsSprite分别为角色的状态管理对象，角色的逻辑对象以及角色的渲染对象。我们在角色的状态管理对象中保存该角色的部分信息，如该角色名，玩家名，动作类型等等。然后我们根据这部分信息初始化出角色的逻辑部分和渲染部分。其中逻辑对象中根据角色逻辑上的三个行为：静止，行走，战斗设置出一个有限状态机，明确该角色在不同的状态，逻辑上应该完成什么样的行为。并且在角色的逻辑对象中保存该角色关于逻辑方面的部分信息，如角色目前的坐标位置，角色的血量，蓝量等数据。接下来，角色的渲染对象会根据角色的状态对象和逻辑对象中包含的信息来初始化，由状态对象决定该渲染对象当前显示的角色名，玩家名等等信息，由逻辑对象决定该渲染对象应处于的坐标等信息。

接下来，在渲染对象中会存在从该对象的创建到目前为止所经历的毫秒数，渲染对象会根据上次切换关键帧后到现在所经历的时间决定是否切换关键帧。如果该时间大于我们给该角色定义的帧间隔，就会切换关键帧，从而达到播放动画的效果。

这样的话即使多次需要播放同一个动作，角色在更新的时候检测到并没有状态发生改变，所有并不会从头开始播放动画。而且在角色状态发生改变的时候，如玩家点击角色右边区域后又点击了角色上方区域，角色播放向上行走的动画时，并非从头播放，而是继承了向右行走的上次关键帧切换时间，从而在向上行走的动画的中间部分开始播放。

地图模块

地图模块分为三个部分，分别为，逻辑地图，渲染地图和渲染地图对应的地图块。

在逻辑地图中，我们把地图图片对应的矩形栅格化，取每十个像素为分界，把地图分为若干个10*10像素的逻辑方格。接

下来我们在逻辑地图的类中构建两个属性，一个是表示地图上障碍与非障碍的信息数组（这里虽然构建一个二维数组比较方便理解，但是考虑到人物在行走时候需要经常判断逻辑格是否是障碍，为了提高程序的运行速度，我们决定使用一个一位数组来表示，然后在应用的时候通过计算得出数组的下标，这样可以充分地利用数据的局部性，并且一位数组可以让我们在读取地图障碍配置文件，即map文件时，可以直接把文件中的内容按长度复制到一整块内存中，加快了加载速度），一个是表示地图上NPC的编号与所在位置的映射（该映射用std::map<int, int[4]>来表示，map的主键用来表示该NPC的编号，之后的数组用来表示该NPC占据的块位置和大小）。

拥有了逻辑地图后，玩家点击不同的位置时，我们可以把点击的像素点转化成对应所在的逻辑格，然后根据该逻辑格的种类：障碍，非障碍，NPC所在位置等情况，对该点击进行不同的处理。

在渲染地图中，我们为了避免一次性加载整个地图情况，采用按需动态加载的模式。事前我们先将地图进行切块，然后根据游戏显示范围的大小，决定每次加载3*5个地图块，用于构建当前玩家视野内的地图。对于每个地图构成的地图块的路径和排列方式，我们保存在一个tab配置文件中，玩家在进入该地图时，渲染地图会读取对应的地图块，然后以屏幕中心点的位置判断需要加载哪些地图块。而且每当玩家走出最中心的地图块时，整个渲染地图会根据新的视野中心进行地图块刷新。

在渲染地图块中，我们为了保持地图块的图片质量，采用tga文件格式的图片，该格式可以保持图片压缩后不失真。

摄像机跟随移动

在玩家游戏过程中，我们要保证，玩家操纵的角色始终要在屏幕中显示。为了达到这样的效果，我们在玩家操纵角色移动时，应该保证摄像机的移动。

在一般情况下，摄像机是以玩家操纵的角色为中心进行移动的，但是当玩家移动到地图的边缘时，即以玩家为中心时，摄像机会显示地图外的景象时。摄像机会保持x轴或y轴锁定不变，当然也有可能是x，y轴同时锁定不变。

当摄像机移动时，需要保证的一点是，移动是UI层之外的时候，位于顶层的UI层，应该无论什么时候都在同一个位置，保证玩家可以点击到。这样就要划分主场景上的不同层的摄像机掩码值，保证摄像机移动时，部分层移动，部分层静止。

战斗系统

客户端的战斗由四个子模块共同实现：战斗管理器、流程控制、数据管理、显示。

战斗管理器负责战斗部分的对外接口，以及整个系统的初始化，具体的逻辑都在其余三个模块中处理。

流程控制是通过一个状态机来完成的，分别有战斗开始、等待玩家操作、等待服务器、立即动作执行、演示、回合结束、战斗结束七个状态，通过状态之间的切换来实现完整的战斗动画展现，在特定的状态中处理服务端消息，并改变数据和显示。

数据模块管理整个战斗的所有数据，在战斗开始时使用服务端传来的数据初始化，主要是双方角色的数据，以方便演示的时候显示。所有数据的改变均由流程模块来发起，然后再改变画面中的数据显示。

显示模块用以整个战斗场景的显示，分为三部分，背景、人物和UI。背景指战斗的背景图片，根据角色所在地图不同而不同，角色指双方的战斗角色及其相应的血条、提示信息等，UI指技能面板、操作倒计时等。

聊天系统

聊天系统使用的是Cocos2d-x提供的TableView组件，用以显示列表数据，只需要绘制出每一句发言的消息块，然后将所有的消息块插入TableView中即可。由于文字和表情需要混插在屏幕中，并且有的文字是可以点击的，比如装备链接，所以需要将每个字都分开，以便排版。

除了消息展示以外，输入框的实现也较为复杂，需要手动添加光标，并且实时计算光标的位置，根据用户点击的位置来锁定附近的光标位置并将光标移动过去，在输入或者删除信息时要从光标的位置开始计算。

本章小结

本章介绍了客户端的基本实现，包括场景搭建、移动、战斗、聊天等。

8. 53140133_初芯宇_计算机科学与技术_基于cocos2d的一项RPG网络游戏软件设计与实现_第8部分

总字数：2023

相似文献列表　文字复制比：0%(0)　疑似剽窃观点：(0)

原文内容　红色文字表示存在文字复制现象的内容; 绿色文字表示其中标明了引用的内容

第6章成果汇报

成果展示

玩家登陆界面

在服务器启动后，玩家运行客户端进入登陆页面，玩家可以在登陆页面中输入账号和密码进行登陆（如图6.1）。

图6.1 登陆界面图

加载界面

图6.2 加载界面图

加载界面用于加载进入下一个场景的时候所需要的资源，玩家登陆时和进入主场景时都需要加载界面来加载资源（如图

6.2）。

选择角色界面

玩家在登陆之后会看到自己建立的角色和该角色所处的位置信息。点击你要选择的角色后进入游戏主场景（如图6.3）。

图6.3 选择角色界面图

创建角色界面

图6.4 选择建立角色的类型界面图

玩家在已建立角色界面点击创建角色按钮，可以跳转到创建角色界面去创建新的角色。创建角色界面分为两个部分，一个是选择建立角色的类型的界面（如图6.4）。另一个是查看即将建立的角色的详细信息的界面（如图6.5）。

图6.5 查看将建立角色的详细信息界面图

主界面

图6.6 游戏主界面

当玩家创建好角色或者选择了自己之前创建的一个角色之后，就会进入到游戏的主界面中。玩家对游戏的大部分交互都是在主界面完成（如图6.6）。

在主界面中玩家可以与NPC对话，使用NPC提供的不同功能（如图6.7）。

图6.7 NPC对话界面

图6.8 聊天系统界面

当然，玩家在主界面也可以和别的玩家通过聊天功能进行对话（如图6.8）。

工作总结

网络游戏的开发总体来说分为服务端和客户端两大模块的开发，使用网络通信将两部分结合起来共同实现功能。

在本次游戏开发中，我完成了登录、移动、战斗、聊天四大功能模块。在开发时我切身体会到了完整的网络游戏开发过程，从登录到游戏到战斗，除了整体的结构设计和大体的功能开发，由于模块之间的相互耦合和数据交互，还带了不少的细节问题，诸如角色的数据存放位置等等。很多模块都经历了不止一次重构过程才有了现在这样基本完成要求的程序结构。

除了体会完整的网络游戏开发过程之外，还在尽量学习商业游戏的开发过程，因为不再是只有程序的游戏开发了，还需要学会怎么和策划、美术协同合作，不能把一切的设定都放在程序代码里，大部分的数据都需要以配置表的形式读取，以便策划在调整游戏设定时自行进行尝试。

并且针对网络游戏的特殊要求还进行了必要的优化和安全处理，尽可能减少服务端的通信量和计算量，并且在客户端的计算和渲染效率上进行改进。整个游戏的开发都坚持着以服务器为主的思想，对客户端传来的一切数据保持怀疑加以校验，以保证服务器不会因为数据问题而崩溃。

未来展望

这次虽然完成了基本功能的开发，但是对于一个商业网络游戏来说还远远不够，无论从功能的完成度上还是从功能的完整性上都还有很多的工作要做。

登录的流程还不够完善，因为涉及到数据库的交互和网络通信，所以会涉及到一些等待状态，这些等待状态需要能屏蔽一些操作，这需要一个完整的状态机来维护。

然后在移动上，在快速切换行走路线时还不够完善，并且当有玩家进入战斗时，其他客户端的显示还有部分问题存在，这些都需要在后续的开发中解决。

除此之外，对于一个完整的游戏来说还有很多的功能需要开发，诸如装备、道具、任务等复杂系统，所以一个完整的网络游戏开发的工作量远不是这两个月的尝试能够见识到的，将来的开发还有很多路要走。

参考文献

[1] 中嶋谦互.网络游戏核心技术与实战[M].毛姝雯/田剑.北京：人民邮电出版社.2014.1-443.

[2] 郁大威.网络游戏服务器架构技术研究与优化[D]：［硕士学位论文］.上海:上海交通大学计算机科学与工程系,2014.

[3] Ignasi Barri．Distributing game instances in a hybrid client-server/P2P system to support MMORPG playability [J]．Multimedia Tools and Applications，2016，75(4)：pp.2005-2029.

[4] 李勇．基于Cocos2d-x引擎的游戏架构设计与实现[D]：［硕士学位论文］．北京：北京邮电大学信息与通信工程学院，2015.

[5] Holte R C,Perez M B,Zimmer R M,et al.Hierarchical a*[C]//Symposium on Abstraction,Reformulation,and Approximation.1995.

[6] 何国辉,陈家琪.游戏开发中智能路径搜索算法的研究[J].计算机工程与设计,2006(13):2334-2337.

[7] 韩宇泽.网络游戏中的寻路算法[J].电子技术与软件工程,2017(24):16.

[8] 张士铎,刘克.网络游戏后台管理系统的关键技术研究[J].中国传媒大学学报(自然科学版),2016,23(05):62-66.

[9] 刘子豪.MMORPG游戏服务器端的设计与实现[J].信息与电脑(理论版),2018(01):61-63.

[10] 孙启.网络游戏服务器面临的技术挑战及对策[J].电子技术与软件工程,2016(21):20.

[11] 王瑞彪,李凤岐,施玉勋,张宪超.基于IOCP机制的网络游戏服务器通信层的实现[J].计算机工程与应用,2009,45(07):75-78+81.

[12] 李大鹏. 基于P2P架构的多人在线游戏的安全性研究[D].吉林大学,2017.
[13] JP Thomas,Mohamed Essaaidi,Youssef Lyhyaoui,Souad Alaoui,Abdelouahid Lyhyaoui,Sté,phane Natkin.Problems of Security in Online Games[J].NATO Security Through Science Series,D: Information and Communication Security,2006,6.
[14] 杨大全,王海军,赵士青,杨佳宁.网络游戏中数据加密与解密技术的研究[J].沈阳工业大学学报,2007(05):578-581.

致谢

在完成“基于Cocos2d的一项RPG网络游戏软件设计与实现”的毕业设计中，出现了很多我个人不容易完成和达到的工作点。在此，非常感谢在这个过程中，给予我帮助的老师和同学们。

首先，我觉得最需要感谢的是我的指导老师韩冬冰，感谢他在忙碌的教学任务中抽出时间来指导和监督我完成此次论文任务。倘若没有了指导老师的帮助，我相信我不会这么顺利地完成此次论文任务中的开题报告，中期检查，后期完善等等一系列步骤。

然后我需要感谢我的企业导师郑宇华。游戏开发对我而言是一个全新的领域，在我刚刚进入这个领域的时候，总感觉无处下手，这时是郑导师告诉我游戏开发应该从什么地方开始，又应该从什么地方改进。在他的悉心指导中，我慢慢明白了游戏开发的核心和要点是什么，并在自己的努力下完成了一款属于自己的游戏。

感谢母校四年来给予帮助的老师和同学们。毕业设计是一个厚积薄发的过程，它是对于以前所学内容的检验，整理与升华。倘若没有以前在计算机学科方面积累的知识，我相信我是无法完成此次毕业设计的。并且在母校的这四年时光中，与老师和同学们的相处不光教会了我关于专业的知识，还教会了我一种终身学习，永不言败的精神，这种精神让我在毕业设计每次遇到困境的时候，不会暗自言败，而是鼓起信心，想方设法越过困难。我相信这种精神，是可以让我受益一生的财富。

最后，再次对在我毕设设计和整个大学生涯中，所有帮助过我，鼓励过我，支持过我的人们致以真挚的谢意。同时我还要特别感谢百忙之中抽出时间的众多答辩老师，感谢他们再次对我的论文进行审阅和评议。谢谢！