# **轩辕传奇引擎架构设计**

作者：韩宇飞

链接：<http://gad.qq.com/article/detail/42268>

**本文首发于知乎专栏：[MACK的游戏开发笔记](https://zhuanlan.zhihu.com/c_157429371)，欢迎各位关注。**

Mackhan 2013/1/8

1、 产品的特征说明和机器配置确定

2. 系统架构设计

2.1 进程线程结构

2.1.1 进程/线程的职责分配

2.1.2 进程/线程间通讯方式

2.2 整体框架

2.3 游戏逻辑流程

3、渲染框架

3.1 引擎的选择

3.2 轩辕传奇的渲染

4、性能优化

5、内部工具

6、第三方组件和工具

7、重难点问题以及解决方案

8、兼容性问题

9、防作弊

10、稳定性和测试

**1．产品的特征说明和机器配置确定**

《轩辕传奇》是腾讯游戏首款3D浅规则战斗网游。游戏是以上古战争为核心的中式玄幻网游，Gamebryo引擎，辅以丰富的动态效果和光影效果，带来出色的画面层次感。根据我们游戏需要达到的性能目标，以及对玩家硬件配置的调研我们制订了轩辕传奇的最低配置和推荐配置。《轩辕传奇》配置需求如下：

最低配置

【CPU】 单核1G

【内存】 1GB

【显卡】 Geforce5200系列或相同档次显卡 64M显存

【声卡】 支持Direct Sound的声卡

【操作系统】WINDOWSXP/2000/7

【DirectX】 9.0C以上

推荐配置

【CPU】 双核 Intel P4 2.8/Intel PE2140/AMD Athlon64x2 3600+

【内存】 2GB

【显卡】 GeForce7600GS/GT ATI  
X1600pro/XT或同档次显卡 256M 显存

【声卡】 支持Direct Sound的声卡

【DirectX】 9.0C以上

【操作系统】WINDOWSXP/2000/7

屏幕分辨率：  
1024\*768及以上

**2． 系统架构设计**

**2.1 进程线程结构**

客户端有五个进程：自动更新、统一登陆、客户端、Cross、IIPS进程。

客户端有六类线程：主线程、网络线程、异步更新线程、资源加载线程、声音线程若干、IIPS微端下载线程若干。

**2.1.1 进程/线程的职责分配**

进程职责分配：

自动更新：完成客户端发布版本补丁的下载和更新。

统一登陆：完成客户端帐号登录、服务器选择的登录过程。

客户端：游戏主进程。

IIPS进程：微端资源后台下载。

线程职责分配：

主线程：游戏主逻辑执行。

网络线程：接收游戏服务器发来的网络包，发送网络包到游戏服务器。

异步更新线程：利用多核的特性，把主逻辑中的一部分运算量放到另外线程中。

资源加载线程：对于异步加载的资源在此线程里排队加载。

声音线程：由fmod底层产生的声音数据加载和播放线程。

IIPS微端下载线程：在轩辕客户端中请求资源，由IIPSHost进程下载资源并和客户端中IIPS线程通信。

**2.1.2 进程/线程间通讯方式**

进程通讯方式：

自动更新：直接创建和调用统一登录进程，无数据通讯

统一登陆：直接创建和调用客户端进程，通过共享内存传递连接数据

客户端：接收统一登录的共享内存的连接数据，登录zone svr

线程通讯方式：

网络线程：通过Ring Buffer发送接收到的网络包到主线程，主线程发送网络包直接通过Windows消息发给网络线程隐藏窗口。

聊天网络线程：通过Ring Buffer发送接收到的网络包到主线程，主线程发送网络包直接通过Windows消息发给聊天线程隐藏窗口。

显示线程：主线程通过Ring Buffer传递显示命令 到显示线程。

资源加载线程：主线程和资源加载线程通过事件和互斥量的方式同步加载列表。

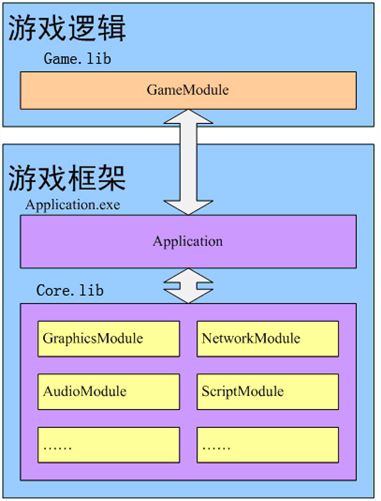
声音线程：由fmod内部管理声音线程间的数据同步。

IPS微端下载线程：IIPS和轩辕客户端通过Windows消息互相同步信息。

**2.2 整体框架**

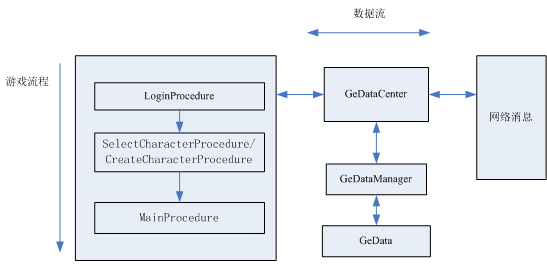
相关模块说明:

* Application: 应用程序接口,游戏程序的入口,主要功能是用来运行框架和管理模块.在这个模块里完成windows主游戏窗口的创建, 提供主循环入口,以EXE的方式实现;
* Game.LIB: 逻辑模块,游戏逻辑部分,包括游戏地图,角色,技能,网络消息处理,UI操作等;
* GraphicsModule: 图形模块，重写了GB的渲染部分;
* SceneModule: 场景和模型模块,包括加载/卸载地图场景,创建/删除Actor, 创建/删除特效.模块本身不提供管理的功能,对象的释放和管理交给外部的调用逻辑模块.
* NetModule: 网络模块,通过分装Tenio的网络组件,实现网络消息的收发功能;
* UIModule: UI控件模块,提供基本的UI控件,供游戏调用;
* ScriptBindingModule: 脚本模块,提供脚本访问功能;
* BaseModule:基本模块,提供诸如向量,矩阵等基本的数据结构和定义;
* ScreenModule: 屏幕空间渲染模块,为UI和屏幕空间的文本图片渲染提供支持;
* InputModule: 提供键盘和鼠标输入的模块;
* FileModule: 虚拟文件系统模块,提供虚拟文件支持;
* AudioModule: 音效模块,为逻辑模块提供音乐音效播放的功能;
* FCFModule: 提供FCF配置文件解释模块,FCF是自定义的配置文件系统;



**2.3 游戏逻辑流程**

a) 概述:



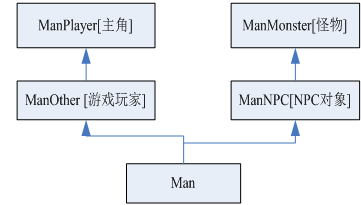
b) 流程

游戏流程分为Login,SelectCharacter/CreateCharacter,MainProcedure几个流程。流程划分的原则,是指该流程具备全窗口独占显示和独立的逻辑控制流程，通过状态机实现。

c) 其他核心模块的接口

GameProcedure类通过提供一系列的静态成员变量作为核心模块和逻辑模块的接口，管理和维护各个模块。

d) 游戏对象结构



游戏中的对象变化通过状态机来驱动，通过继承和组合实现不同的对象。

e) 逻辑地图和逻辑对象和渲染模块的组织关系

所有的逻辑对象派生自Man,组合了一个Actor的渲染对象。

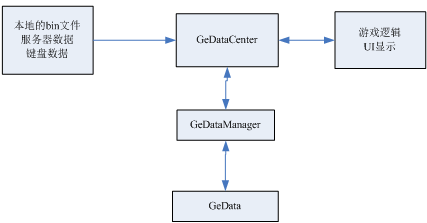
LogicMap(逻辑地图)，组合了一个Map的渲染对象

所有的Actor对象通过添加到SceneModule模块的Actor管理器中管理,交给场景模块管理和地形一起统一渲染。

f) UI模块结构:

g) 数据模块设计

提供一个数据中心的概念,将主要的数据管理器由一个数据中心作为统一的接口进行访问和刷新.将数据分块,而不是按照逻辑结构划分,这样具备更好的扩展特性。数据来源从本地的数据或网络消息获得。



h) 网络模块

所有的接收到的消息会压到一个队列里,消息轮询的时候,交给相应的注册处理函数来处理.所有的处理函数指针保存在一个map里.

**3． 渲染框架**

**3.1 引擎的选择**

《轩辕传奇》使用Gamebryo引擎的2.3版本。Gamebryo是一款主流的跨平台游戏引擎，该引擎使用简单比较成熟，有上百款商业游戏使用Gamebryo开发。轩辕选用Gamebryo主要考虑了以下几点：

1）稳定性：国内外有众多游戏都使用Gamebryo开发，包括许多MMORPG，在技术上是一款成熟稳定的游戏引擎。其中Gamebryo2.3是比较经典的一个版本，也是使用最多的一个版本。

2）灵活性。Gamebryo提供完善的数据管理框架。相对Unreal，CryEngine等引擎虽然没有统一的开发流水线但是可以很方便的扩展和修改，进行深度定制。例如轩辕传奇的就使用了自己开发的渲染渲染系统，此外还开发了场景编辑器，技能编辑器，UI编辑器等多套定制的工具并完美和Gamebryo进行了集成，极大的提高了游戏开发效率和实现了与众不同的游戏效果。

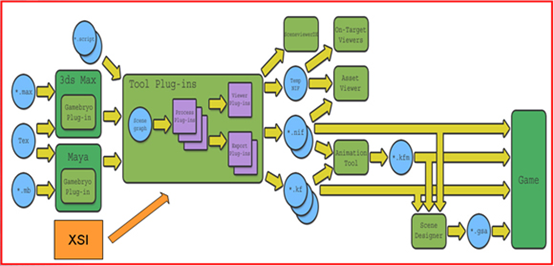
3）便捷性。Gamebryo在中国有着较完善的技术支持，此外Gamebryo有非常详细的文档，教程和社区，同时因为使用者众多可以方便快速的获得各种问题的解决方案。Gamebryo提供便捷的导出插件，美术可以快速预览和导出美术资源。

**3.2 轩辕传奇的渲染**

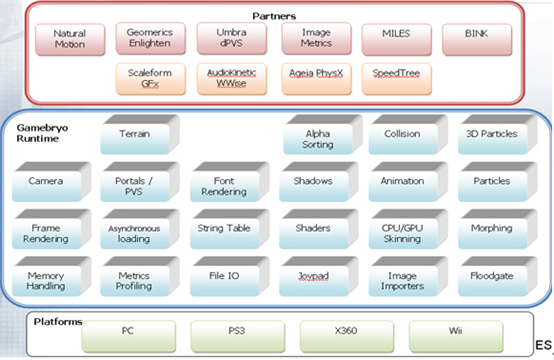
《轩辕传奇》渲染框架如下：



Gamebryo开发流水线：

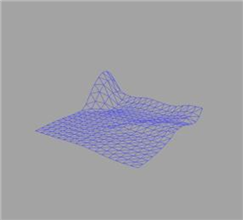
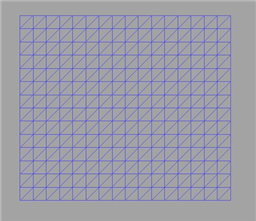


GameBryo的基本构成（其中轩辕传奇主要使用GameBryo的RunTime，并做出了一些修改和添加了一些定制组件，此外在RunTime的基础上又开发了大量编辑工具）：



**地形渲染：**

轩辕的地形渲染以Chunk为单位渲染，每个Chunk有16x16x2个三角形，17x17个顶点。为了共享顶点索引数据，地形数据使用了多流。地形渲染使用纹理混合，支持高光，支持动态和静态阴影，支持AO，支持Reflect，支持静态点光效果，支持雾效，支持半透明渐变。因为针对轩辕俯视角为主的游戏，同屏地形面数不会很高并且在目前显卡上渲染瓶颈也主要在DP数而非面数上了，同时轩辕的地形数据使用多流在GPU中解压生成显卡总线也不是瓶颈所在，所以地形的渲染只使用了材质LOD而没有使用面数的LOD，此外地形的渲染还使用了LightMap合并，材质排序等多种优化方式。具体结构图如下：



**地形的材质和材质LOD：**

轩辕的地形采用五层纹理混合，最多可以设置五张地形纹理。此外五层地形纹理中的四层还可以设置高光贴图（每个Chunk的高光贴图只有一张RGBA分别对应四层的高光）。五层纹理之间的混合系数又有单独的一张贴图保存（也是使用贴图的RGBA分别做每层纹理之间的半透明系数）。最后每个Chunk有两张DXT1的LightMap，一张LightMap的R通道保存编辑器烘焙的阴影，G通道保存编辑器烘焙AO，一张LightMap保存点光光照和点光阴影。  
轩辕地形效果如下：



为了提高渲染效率，轩辕传奇对较远的地形（距离根据用户配置调整）做了材质LOD优化。轩辕传奇的LOD方式类似于CryEngine的材质LOD，每个地形在编辑器编辑好后会烘焙一张俯视角的全局地形效果图（每个网格使用4个像素，每个chunk对应32x32的贴图，再以256x256为单位合并成dxt1的贴图，最大1024x1024的场景也不会超过2.6M），需要LOD的地形Chunk将使用这张全局地形贴图做基本纹理，这样一个Chunk只需要采样一张纹理就可以渲染完，而渲染的实际效果肉眼很难区分。LOD效果如下图，因为实际效果较难区分这里是故意设置很近距离的地形就开始LOD起到对比作用：



**角色和场景物件的渲染:**

轩辕的角色和场景物件渲染使用了多种材质，Shader有UberShader自动编译生成。Shader支持Glow，Reflect，高光，阴影，雾效，半透明渐变，三种顶点光照以及同时最多54盏点光。角色和场景物件的材质由角色本身的配置，游戏当前图形配置共同决定动态改变。角色一个DP最多支持40根骨骼硬件渲染。



**各种特效渲染：**

轩辕的特效包括使用自定义Shader的天空球，水，草，天气，贴花（后期处理投影贴花和几何体贴花），各种后期处理（例如景深，全屏泛光，Bump，，后期处理描边等等）已经使用GameBryo的标准材质渲染管线的特效，支持粒子，Glow，Glass，Bump，Dark，Decal，Shadow，Fog等等。



**4． 性能优化**

轩辕传奇在性能优化上做了大量工作也是轩辕传奇相对同类型游戏比较有优势的地方。轩辕传奇的优化主要在帧数、内存和容量方面。

帧数方面，在推荐配置上轩辕传奇实现了最高400人同屏放技能也能保证游戏流畅运行。在最低配置上轩辕传奇也保证50人同屏放技能能流畅运行。

内存方面，轩辕传奇实现了启动时内存只占用300M左右，任何极限环境（最高配置400人同屏放技能）下内存峰值在1G内存以内，长时间（1天以上）挂机内存无膨胀无泄漏。

容量方面，轩辕传奇经过多次优化在有120张完全不同的地图，2000件Avatar，上万种怪物物件，高质量静态阴影光照贴图等前提下完整客户端安装控制在1.6G，微端安装包更是只需要270M左右（极限微端只需要30M）。

**帧数方面的优化**（使用VTurn，Pix，GPA等工具分析和找出CPU、GPU的热点和瓶颈进行优化）：

1.动态调节配置和Shader分级：在游戏中我们针对不同的硬件实现了三套Shader，其中超低配置的Shader对渲染进行了大幅优化（地形一层纹理，角色一层纹理等）主要用来满足最低配置的电脑，中等配置和高等配置分别支持Shader2.0和3.0，此外同套Shader还有多种图形配置参数默认的动态配置调节功能可以根据当前帧数动态优化图形配置。玩家当前硬件对应哪套Shader和何种图形配置基本在游戏开始时由配置文件设定。

2.UI方面的优化：UI一直占渲染的很大比重，轩辕使用自主开发的UI框架，UI的渲染使用GB NiScreenElement渲染。在实际使用过程中我们发现GB的UI渲染存在Lock次数过多，DP过多问题。最终我们实现了自己的UI渲染，一方面通过画布和程序优化将UI尽可能合并到一个批次渲染，另一方面所有UI使用一个不变的IB，而VB也只有一个在需要改动时Lock一次进行设置，从而避免了Lock  
VB IB的等待和频繁设置，极大程度的提高了UI的渲染效率有30%以上的性能提升。



3.Shader的优化：在渲染效果的相关Shader开发完毕之后，我们会再次逐行分析和优化Shader，每次优化完后将编译成汇编指令和优化前进行对比，检查指令优化的个数和是否存在冗余指令，如果还有优化空间会再次反复迭代，在保证效果的前提下尽可能减少Shader指令。

4.角色优化：角色渲染几乎所有的角色都由多个 Avatar 部件组成，而多个蒙皮会形成多个批次，同时多张贴图也会形成多个批次，例如每个玩家m个部件，每个部件k个材质，同屏有n个玩家，渲染批次 = n\*m\*k。DP数是现代的显卡的瓶颈，当同屏显示大量玩家时高DP数会导致帧数大幅下降。针对角色的优化我们一方面合并蒙皮将模型数据合并成一个VB\IB，另一方面合并材质，将各个部件的贴图合并成一张。最终渲染一个角色只需要一个DP就可以完成，也保证了在400人同屏的情况下推荐配置也可以较流畅的运行游戏。



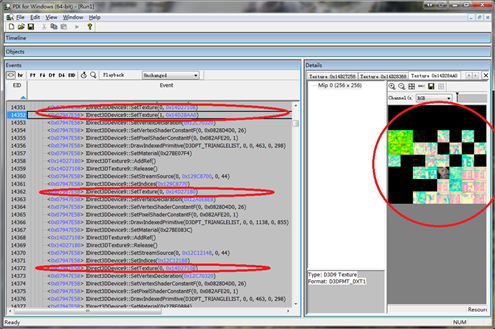
5.LOD优化：轩辕LOD优化方面采用了非常多的方式，例如场景中区分主次物件，不同大小重要程度的物件（自动设置结合美术编辑器中手工指定）加载显示的范围不同半透明渐显渐隐；骨骼更新LOD降频，较远的角色怪物等会降低骨骼更新的频率；粒子LOD，粒子显示的数量效果由和摄像机的距离觉得；材质LOD，地形，水，场景物件，角色的材质也加入LOD，距离较远的角色，地形等使用单一DP的材质。例如下图粒子LOD效果。




6. 减少渲染状态切换次数：一方面对渲染列表进行了材质排序和Shader排序，减少渲染状态切换次数。

另一方合并光照贴图，地形按Chunk为单位通过一套较复杂的规则对该Chunk上的场景物件贴图使用推箱算法合并光照贴图。每个物件只设置一次光照贴图，多个相邻的物件由于共享了LightMap纹理，大大减少了纹理切换率和显存碎片，提高了显存利用率。



最终平均减少了10%的状态切换次数。

7.其他一些逻辑CPU方面的优化：此外轩辕传奇还通过VTurn分析了CPU的各种热点，对一些算法进行了优化，例如拾取算法，碰撞检测等。

**内存方面的优化**（通过自主开发的Hook方式的内存工具找到内存的瓶颈进行优化）：

1.动态加载Shader：轩辕传奇一套配置的Shader最多可使用高达1200个Shader，在游戏初始化时加载如此多的Shader是非常占内存的，针对这个问题轩辕采用了动态加载Shader的方式，在每个Shader真正使用时才加载，可以减少大约120M左右的内存空间。

2.动态加载UI的Nif特效：UI的Nif特效比较多，优化后只有该特效渲染时才加载并加入Cache中，当超过一段时间不使用则释放；

3.字体优化：将字体作为缓存优化，不直接加载所有字体文件，减少了30M左右的内存空间。

4.网络包缓存优化：修改网络包缓存机制，平均减少30M左右内存空间，最大可减少120M左右内存空间。

5.ICON优化：因为ICON比较小，对ICON资源进行适当的压缩和缩小，减少13M左右内存空间。

6.DX内存释放：定期进行DX的内存回收。

7.bin表优化：修改了tagpathlist结构体，spelldata的加载方式，propertsets的map类型为NiTMap等，减少60M左右内存空间。

8.UI贴图优化：所有的UI贴图只在渲染的时候加载并加入缓存，一段时间不使用自动释放。

9.其他一些优化：例如技能的USpellProperty优化，组件的Log.dll优化去掉后减少10M左右内存空间，美术资源的清理和合并，修改一些导致内存膨胀和冗余的bug，对比Google，Windows和GB内存管理修改内存管理策略等等。

**容量优化：**

1.微端:集成IIPS实现数据动态下载后台加载，动态Lod逐级渐变等，最终实现了微端270M安装包（最小可达30M）。对场景中各种资源指定智能的优先级列表，完善的残缺体验，保证30K的下载速度玩家也可正常游戏，100K的下载速度和完整包一样体验。游戏中加入大量残缺体验和lod低精度资源保证即使本地资源不存在或者在下载过程中也不影响游戏体验。

2.地形挖空数据不导出：美术在编辑地形时会有很大的挖空面积，这部分数据可以不导出。

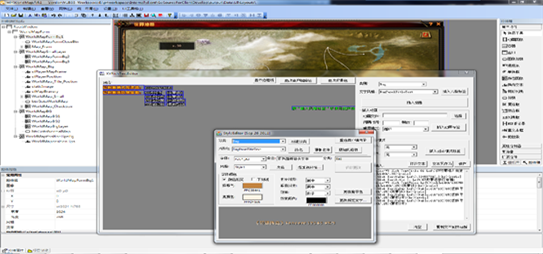
3.地形使用多流和压缩优化：地形每个Chunk顶点索引是相同的，只需要全局一份使用多流共享；每个Chunk中顶点只有z不同，在Shader中只需要顶点的z和网格索引就可以计算出顶点的xyz；地形的两套UV（基础纹理和LightMap）也可以是用索引计算出来，每个顶点可以节省16个byte；地形顶点法线和顶点色可以各使用一个DWORD压缩，在传输到显卡的时候利用D3D自动转成float，每个顶点共节省20个byte。最终地形容量节省了大约10%。



4.地形纹理优化：地形数据中容量最大就是地形的光照贴图。轩辕的地形的光照贴图总共有三张，一张是阴影贴图，一张是AO贴图还有一张是点光颜色和阴影贴图。优化光照贴图采用了三个方法，一个是编辑器导出时判断Chunk的光照贴图是否有效，如果一个Chunk没有AO效果或者不受点光影响，那么则不导出该Chunk的光照贴图。第二个方法是合并光照贴图，将阴影贴图和AO贴图分别存入一张DXT1的R和G通道，将点光贴图存入另一样DXT1中，这样只需要两张1/8压缩比的贴图就可以保存三张光照贴图了。最后一个方法是调整光照贴图的分辨率，我们测试发现占4%的512x512分辨率的大光照贴图容量占用所用光照贴图的50%，我们通过降低512x512贴图的分辨率保持或者提升小分辨率光照贴图的分辨率，最终实现了在保证了显示效果的前提下大幅降低了地形数据的容量。此外对地形的高光贴图也进行了优化，地形的高光贴图一开始是在编辑器导出时合并生成的，一个32x32的地图高光贴图就会占用256M，优化后在客户端加载地形时通过编辑器导出的高光贴图混合系数后台动态合并，大幅减少了高光贴图的容量。此外还进行了其他一些优化，最终优化后地形数据减少了80%的容量。

**5． 内部工具**

GameBryo引擎本身的编辑工具并不完善和易用，而使用Unreal等有强大编辑器的引擎开发又会面临效果单一，很难扩展的问题。轩辕传奇在既保证灵活性扩展性，又保证高效性和稳定性的前提下开发大量编辑工具，极大程度的提高了开发流水线的效率。其中主要的几个工具如下：

* 场景编辑器：场景编辑器是整个流水线中最重要的工具，主要由美术和策划使用，轩辕的场景编辑器实现了地形编辑（包括地形，水，草等），场景物件编辑，NPC和怪物放置，物件运行编辑，行走层编辑，天气雾效编辑，天空球编辑，静态阴影烘焙，AO效果烘焙，点光效果和点光阴影烘焙等等，使美术能在所见即所得的情况下快速高效的开发和修改场景。
* Shader编辑生成器：通过ubershader，利用脚本配置按照指定命名规则自动生成轩辕传奇所用的Shader，总计自动生成大约2700个左右的Shader在游戏中动态加载使用。
* 技能编辑器：主要由策划和美术使用，轩辕的技能编辑器主要提供各种技能的编辑，每个技能对应一个CMT，每个CMT可以包括声音层，特效层，动画层，逻辑层等，而CMT也不仅仅应用于技能，包括角色动画，换装等多种效果也都可以实现，使用者可以快速所见即所得的编辑出所需要的效果，而导出数据也可以动态在游戏中预览。
* UI编辑器：主要由策划和美术使用，使用简单易上手，可以脱离游戏编辑UI，同时支持编辑UINif特效，可以实现各种动态效果。
* AI编辑器：主要给策划使用，轩辕传奇中各种怪物，NPC，Boss的AI都通过AI编辑器以表格和图标的形式编辑。
* 内存分析器：主要由程序使用，通过Hook的方式在游戏运行时收集和分析内存信息，主要用于分析和解决内存泄漏和内存膨胀的问题以及内存优化。

**6． 第三方组件和工具**

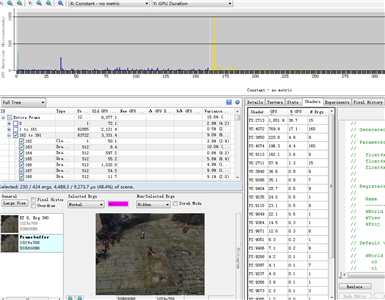
轩辕传奇主要使用的一些第三方组件和工具如下：

Fmod：声音组件。

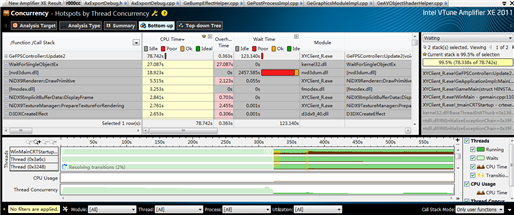
Bink：视频组件。

DirectX SDK Pix For Window：微软的渲染分析工具，主要分析渲染流程，找出瓶颈热点和调试渲染错误。

Intel GPA：渲染优化分析功能，用于定位渲染瓶颈和解决渲染问题。



IntelParallelStudio：是Vtune（有独立界面）和ThreadProfiler功能的合集，内嵌如VC中无独立界面。其中包括Composer，Inspector（线程错误等），Amplifier三个工具。主要用于CPU相关优化。



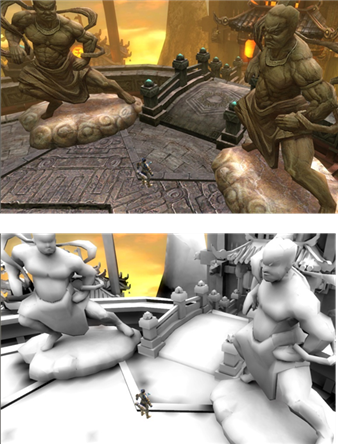
**7．重难点问题以及解决方案**

在轩辕传奇开发的过程中遇到很多难点问题，最终通过各种方法解决了这些问题，也实现了轩辕传奇在各种配置各种环境下，基本都没有卡顿现象保持较稳定的帧数。以下列出了一些比较重要和比较难解决的问题和解决方案：

* 资源加载优化的问题。轩辕传奇的地形和资源数据采用后台加载的方式实现，根据用户的硬件和配置加载主角周围一定范围的地形和资源。在开发过程中因为地形图资源的分布不均，玩家硬件配置的不同和一些逻辑方面的原因（某个区域集中大量玩家或者会触发剧情等），在玩家高速移动（坐骑，加速药水）的情况，在场景中跑动可能会遇到卡顿的现象影响用户体验，这是不能接受的。通过使用测试工具例如VTurn,分析代码，大量测试我们最终定位了很多影响加载的一些问题，其中比较主要的有线程的阻塞和空闲问题，加载任务分布和优先级，缓存和缓存命中问题以及一些逻辑算法上等。
* 针对线程问题，为了保证线程没有空闲和Lock问题提高多线程加载效率，我们将原有多个IO线程读取改为单一IO线程，串行读取磁盘数据。此外重新检查了多线程的线程锁尽可能减少锁的使用，多线程的任务投递改为ringbuf不使用锁提高多线程读取的性能。
* 为了防止加载任务分布不均的问题，我们修改了后台线程加载任务队列，限制每帧加载任务的吞吐量。同时又为了防止加载顺序不当影响玩家体验（例如玩家跑动地形没加载没显示出来），我们将加载队列改为LIFO方式，优先加载最近请求，对后台加载的资源划分优先级，例如地形和NPC优先加载，保证用户的视觉体验。
* 针对缓存问题，我们对游戏资源进行区分，实现了统一的Cache机制，每种类型的资源都有不同的Cache策略，并进行了大量测试，设置了比较合适的缓存策略（缓存资源数量、时间等）提高缓存命中率和效率，此外还对一些数据利用系统缓存进行了优化，通过预加载的方式让常用资源保持热度。
* 在逻辑算法上我们使用 Map 和 list 双缓冲检索资源，即具有列表的快速插入和删除的性能，又具有 Map 的快速查找性能。此外有修改了大量影响加载的代码和逻辑。
* Gamebryo多线程问题。轩辕传奇是一个多线程游戏，在游戏过程中有海量数据要通过后台加载和后台处理，但是Gamebyro2.3虽然是线程安全的，但是引擎内部存在很多致命的多线程bug，因此轩辕传奇一开始遇到很多很诡异的渲染错误和崩溃。通过大量测试和分析源码，我们发现并解决了Gamebryo2.3版本中的很多多线程bug，对引擎的源代码做出了一定修改后最终实现了轩辕传奇高效稳定运行。
* 地形优化中遇到一些难点。例如前面地形优化中描述的为了最大程度减少地形容量，压缩法线和顶点色每个顶点共节省20个byte同时又不降低效率，我们希望利用DX格式转化自动将DWORD转成float解压，但是GameBryo的常规格式不能指定DWORD作为顶点色（必须是float，GB最终会遍历每个顶点转换成DWORD），所以我们使用了多流的方式，将一个Chunk的顶点坐标放在一个流，顶点色和法线放在另一个流传入GPU，实现了优化。还有就是在地形合并LightMap时，我们合并时遇到了绿边黑边的问题，这是因为光照贴图采样方式都是线性采样,将光照贴图合成一张后，内部就会互相采样过界。比较明显的绿边是因为点光贴图采样到了阴影AO贴图，阴影AO贴图的R通道是阴影G通道是AO，大部分阴影AO贴图都是绿色。黑边则是因为阴影AO贴图采样到其他贴图。针对这个问题，我们在编辑器中烘焙每张光照贴图时缩小一个像素缩一圈（多余的一圈填充原始贴图的颜色），在客户端将UV缩小偏移的做法，这样每张光照贴图之间都有两个像素的空间，解决了边界的问题。



* 烘焙AO的问题。轩辕场景是有AO效果的，为了保证游戏效率没有使用实时SSAO而是在编辑器中烘焙静态AO效果。一开始我们在编辑器中集成了Beast引擎烘焙，测试后发现速度实在太慢一个比较大的场景可能需要多台电脑烘焙几天几夜无法满足快速开发的需要。因为我们在编辑器中开始自主开发烘焙AO系统，首先我们尝试了射线法烘焙AO，开发完毕后发现效率还是比较低。然后我们又实现里利用GPU使用SSAO的方式烘焙静态SSAO，实现后发现速度还可以但是精度效果无法满足。最后我们发明了一种了球形光照分布的方式来烘焙AO，通过计算大量球形分布的光源，利用叠加ShadowMap的方式烘焙AO。最终测试发现这种方式烘焙速度极快，一个物件烘焙使用不到1秒一个比较大的场景也只需要几分钟就可以烘焙完毕，同时AO的效果也比较好，最终我们采用了这种烘焙方式并加入了一些后期的实时Gamma修正以供美术修改，并申请了专利。



* Cuda加速的问题。在轩辕开发的过程中我们发现轩辕的场景编辑器导出数据的速度非常慢，分析后发现瓶颈主要在贴图格式的变换和DDS压缩上，为了提高效率我们使用了Nvidia公司的Cuda库利用GPU加速场景导出。加入后速度提高了几十倍非常满意，但是后续使用时发现在不同机器上导出的场景经常会有黑色块出现，而是是低概率随机出现。一开始测试发现如果导出时锁屏，设备丢失就会出现这种现象，我们又加入保证编辑器导出时无法锁屏的机制，但是最后使用时发现黑色块的现象只是有所减少但是还是出现，最终通过测试确定是Cuda库导致的，和Nvidia同事沟通后发现Cuda库还不稳定，在不同配置的机器特别是较老的显卡或者ATI的显卡上还有很多bug。最终我们放弃了使用Cuda库，而是通过烘焙时直接烘焙出需要格式以及修改一些导出算法的方式提高编辑器导出效率。
* 后期处理描边，投影贴花在个别电脑上出现斑点的问题。这个问题最终定位是因为在不同显卡上深度buf精度不同导致，我们通过修改后期处理流程不直接使用浮点深度的方式解决。

**8． 兼容性问题**

轩辕传奇软件方面兼容win98，win2000，xp，win7等主流windows操作系统，同时兼容各种语言版本，主要在开发过程中对游戏文字、编码方式等做出了兼容。硬件方面轩辕传奇兼容支持shader model 2.0及以上的显卡，兼容早期的使用CPU计算VS的集成显卡。因为阴影等效果的不同需求，轩辕传奇针对Shader2.0和3.0分别编译了两套Shader，此外针对早期的集成显卡例如G41，性能很低的独立显卡5200等又单独实现了一套极致优化的Shader，并将市面上各种显卡配置信息生成了配置文件，在游戏启动时会自动选取相应的Shader和配置。此外轩辕还实现了特有的动态自动配置调节系统，在游戏中可以根据玩家的硬件配置和游戏环境无缝动态调整游戏配置，例如玩家在空旷的野外系统配置可能会很高，但是当玩家在百人千人城战时系统又会动态调整到比较低的配置，始终保证玩家游戏流程和帧数稳定。

**9． 防作弊**

在防作弊方面一方面依赖研发中心的反外挂系统方案保护，采用安全组提供Antibot系统，增强程序的防调试能力。另一方面轩辕还在游戏内集成了自主研发的反外挂插件，在玩家游戏时可以动态检测和收集玩家是否作弊的信息并发送到服务器上并自动分析出外挂和作弊情况。

**10． 稳定性和测试**

轩辕传奇在游戏稳定性上花了很多功夫。在代码层面定期重构，每日构建，每日review（保证每行提交的代码都经过检查）。此外在游戏内加入了大量的自动化测试、报警、日志和上报功能，争取在开发阶段尽可能减少问题。在测试方面根据开发任务指定测试用例和测试环境，和测试同事制定各种测试用例。此外轩辕传奇也开发了一些自动测试和数据代码检查的功能，最大程度保证客户端的稳定性。