WBS字典

2.4 验证系统：

在开发之前，用实践的手段熟悉需要用到的工具/环境/引擎。

3.1 游戏设计：

规划游戏的游戏方式，交互方式，游戏场景，胜利条件等。

3.2 架构设计：

以Irrlicht引擎为核心，搭建外围组件。如网络，控制，音效，界面等。

3.3.1 3D动画模块：

每个节点有自己的动画列表，为节点添加动画，则场景树在遍历节点时会运行动画列表中的动画。

这里动画不是通常意义上的动画，只要希望游戏在运行时，某个节点不断执行某种操作时，就把这种操作封装成一个动画。

平移，旋转，缩放这些普通的动画。

所有恒星相对镜头静止也是一种动画。

导弹的自动追踪飞行是一种动画。

飞船的控制是动画。

武器发射是动画。

3.3.2 控制模块：

本质上是动画的一种。通过响应鼠标和按键消息来完成对模型的控制。

3.3.3 模型模块：

节点树管理场景中的所有节点，在场景中增加模型就是在节点树上挂载节点。通过设置节点的Visibility属性来选择显示哪些3D物体。

3.3.4 网络模块：

我们的项目为局域网对战游戏，游戏的过程中需要和其他主机通信。

3.3.5 Python模块：

游戏的开发过程中使用了Python脚本语言来辅助开发。主要是在布置场景和布置UI界面时，使用脚本语言不需要重新编译整个工程，节省了大量的时间。

Python模块和C++之间使用BoostPython库进行连接。需要在python里面使用的功能，要现在C++里面写好，然后导出到Python供Python使用。

3.3.6 分场景：

游戏中通过场景之间的跳转来实现游戏界面的跳转。

每个场景自己维护自己的一系列资源。

3.3.7 UI：

很可惜鬼火引擎并没有2D界面的引擎，为了使我们的游戏中具有2D界面，我们参照鬼火的引擎结构，自己实现了一个2D图像引擎。

其中有一个UIManager，对应鬼火的SceneManager；

IUIObject，对应鬼火的ISceneNode；

可以为IUIObject添加UIAnimator，对应鬼火的Animator。

IUIObject之间可以实现父子节点相互嵌套，动画的相互嵌套，变换都是自己写矩阵相乘实现的。

3.3.8 Shader：

我们使用GLSL作为Shader语言。

游戏中Shader进行了大部分的渲染工作。

游戏中没有光源，所有的光照都是在shader中计算的。

使用shader和动画我们做出了很多漂亮的效果，比如星球的大气层，星球的云彩，星环，恒星光晕，飞船尾焰，空间站灯光，水平仪和垂直仪等。