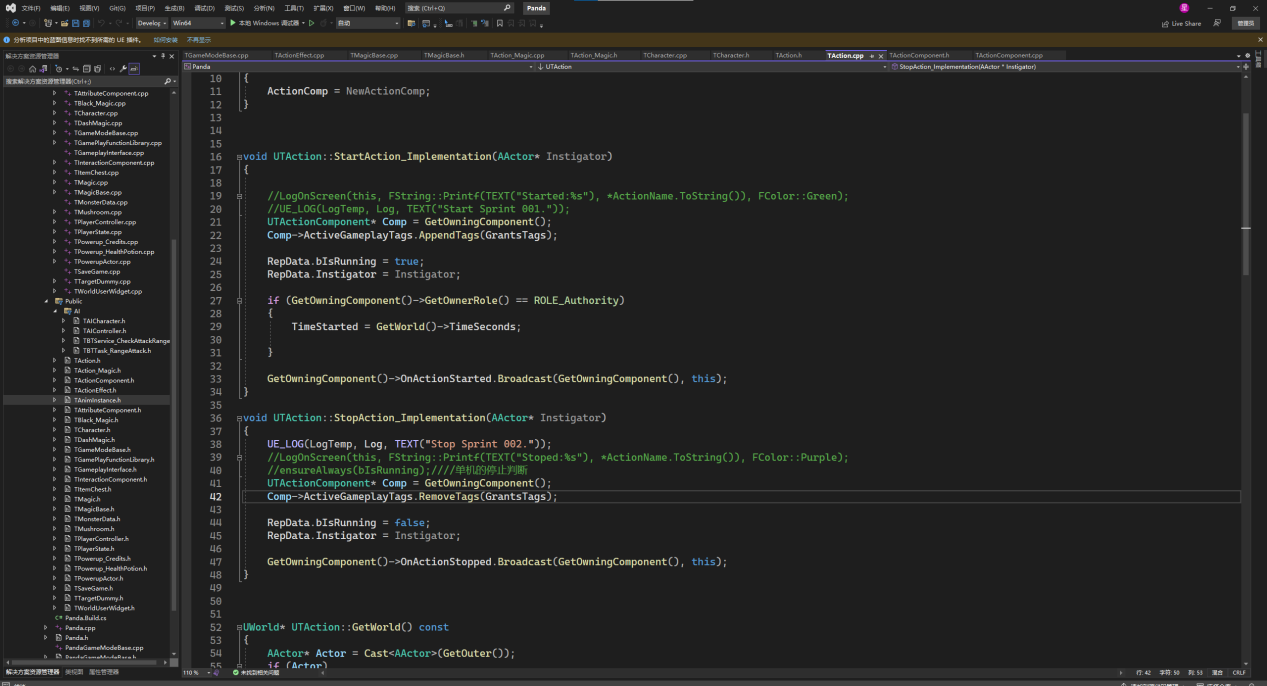
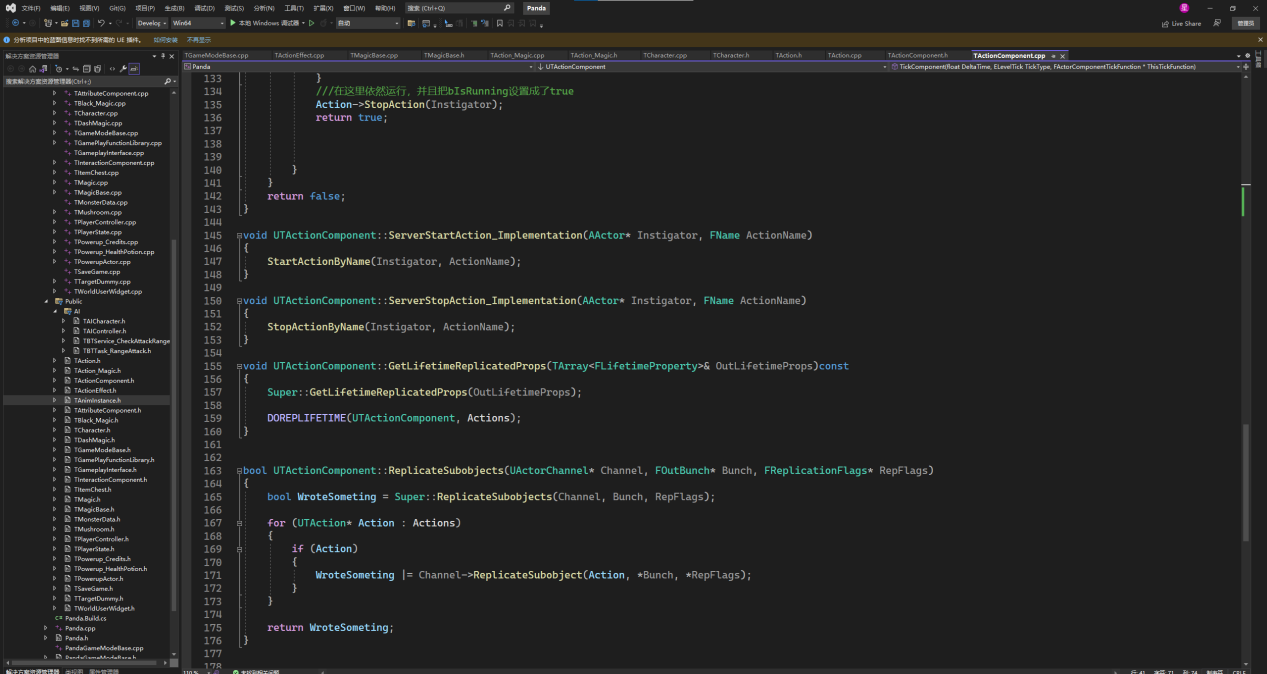
Ureal演示项目部分代码展示

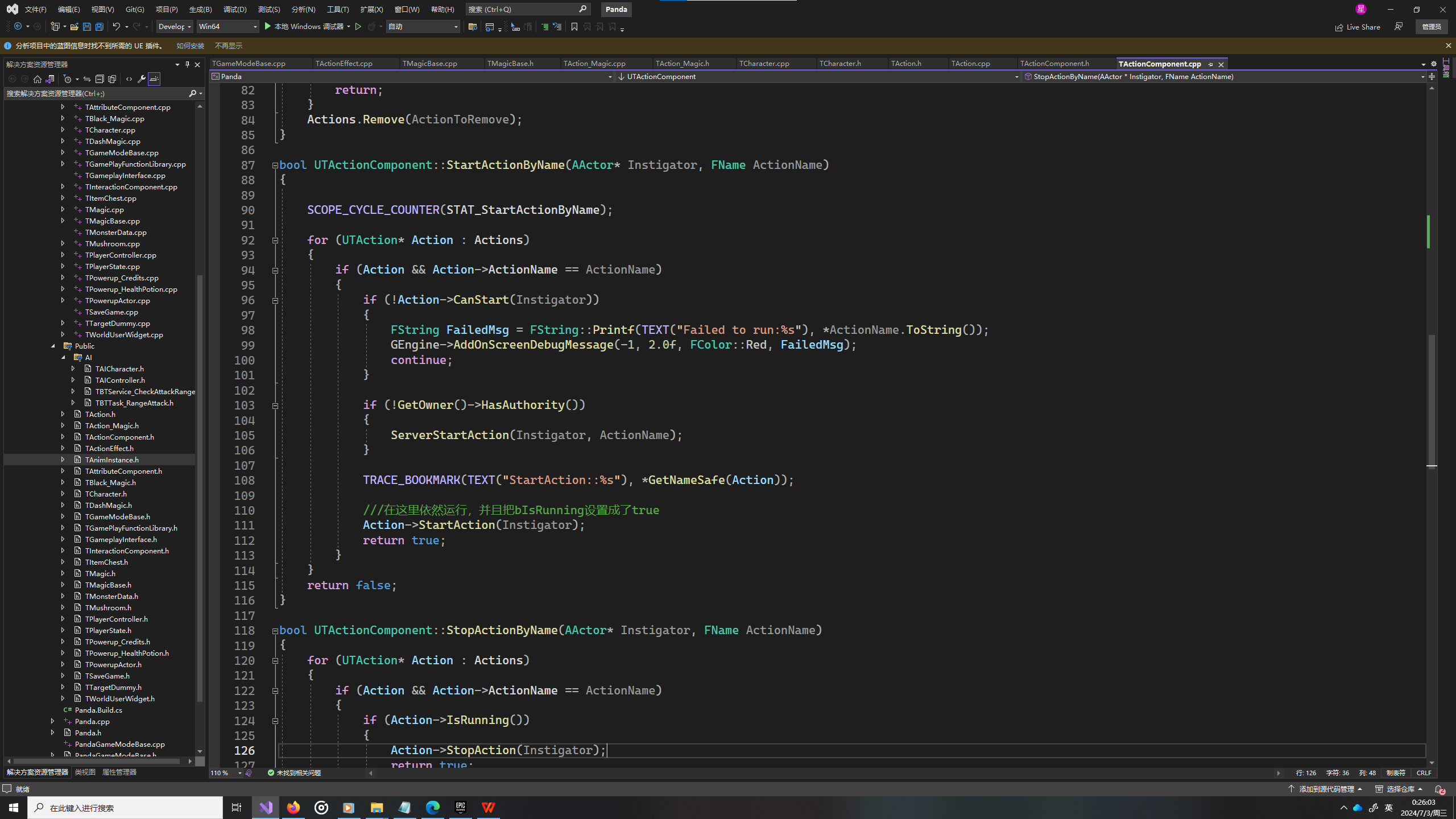
1.Gas系统，通过action component调用start action使用action方法，不同的action具体实现可以分别创建不同的action类重新实现，action的选择和互斥使用tag标签实现；用Gas的做法使得技能或增益可重复利用而且扩展相对灵活，可以定制不同逻辑的详细差别，GAS 设计用于多人游戏和网络游戏，具有强大的服务器端验证和同步功能，确保玩家之间的能力和效果同步正确，并防止作弊。

Action

上图是start Action的实现通过Tag标签判断action是否能运行，不同的action分别有不同的停止机制，如按钮松开，或时间到达等，这里的GetOwing Component重写了方法因为默认的Get Outer不能直接cast成Action Component。

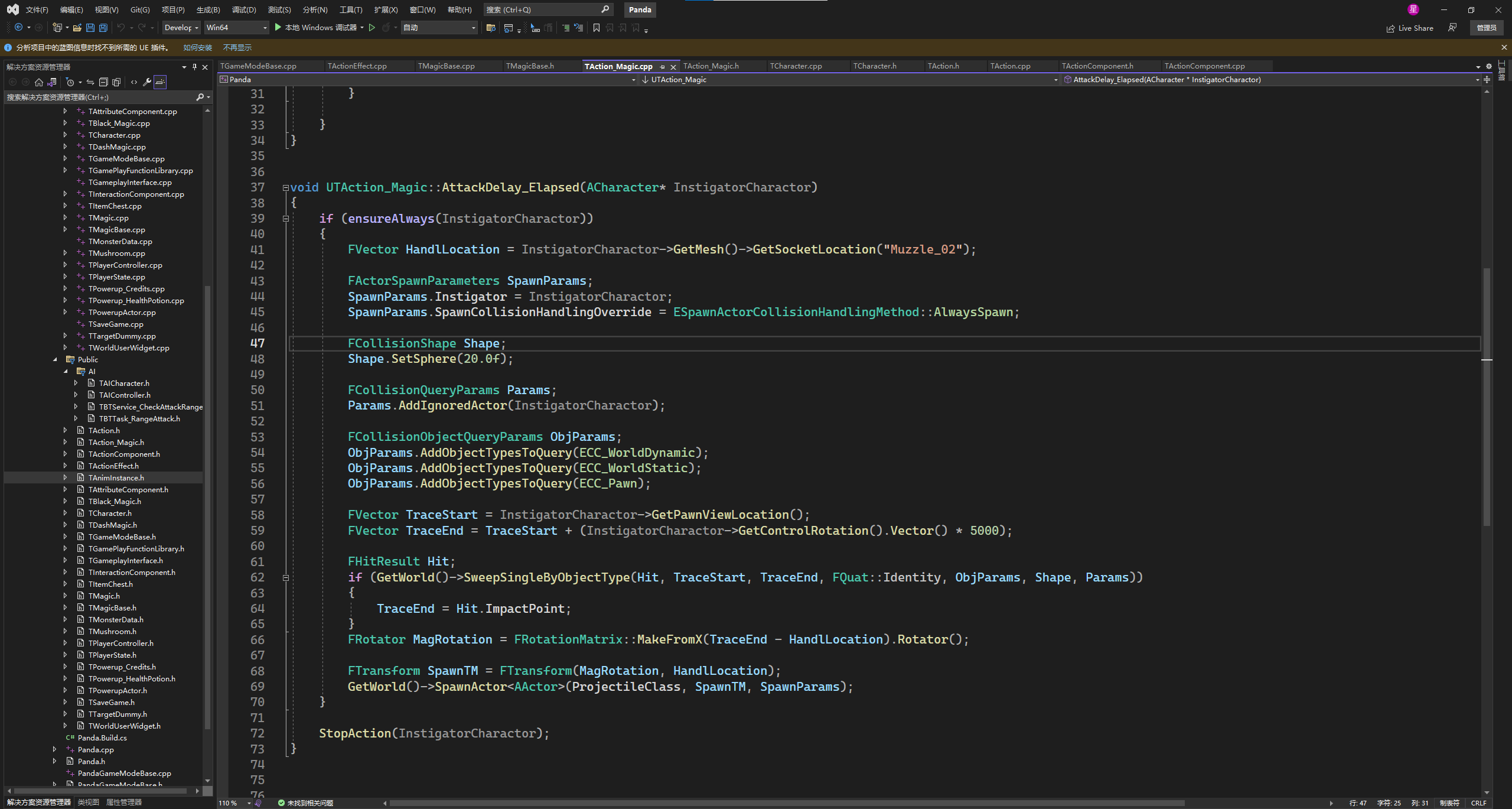


这里做了对服务器调用的判断，并且使用类似于UDP的逻辑进行网格同步，在另一个服务器没用接受到调用的时候先继续执行，接受到后继续调用，网络游戏非常容易在等待验收的时候造成大量延迟，这样可以避免部分延迟发生但牺牲了TCP的高可靠性。

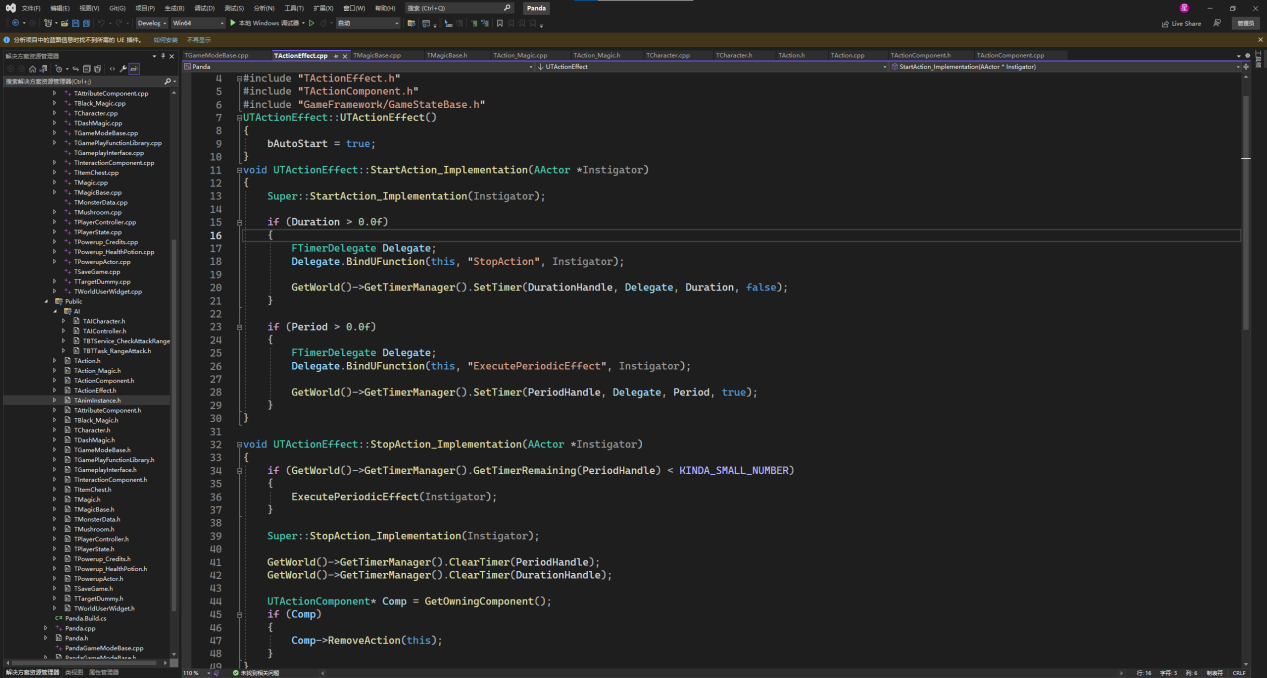


Action Component

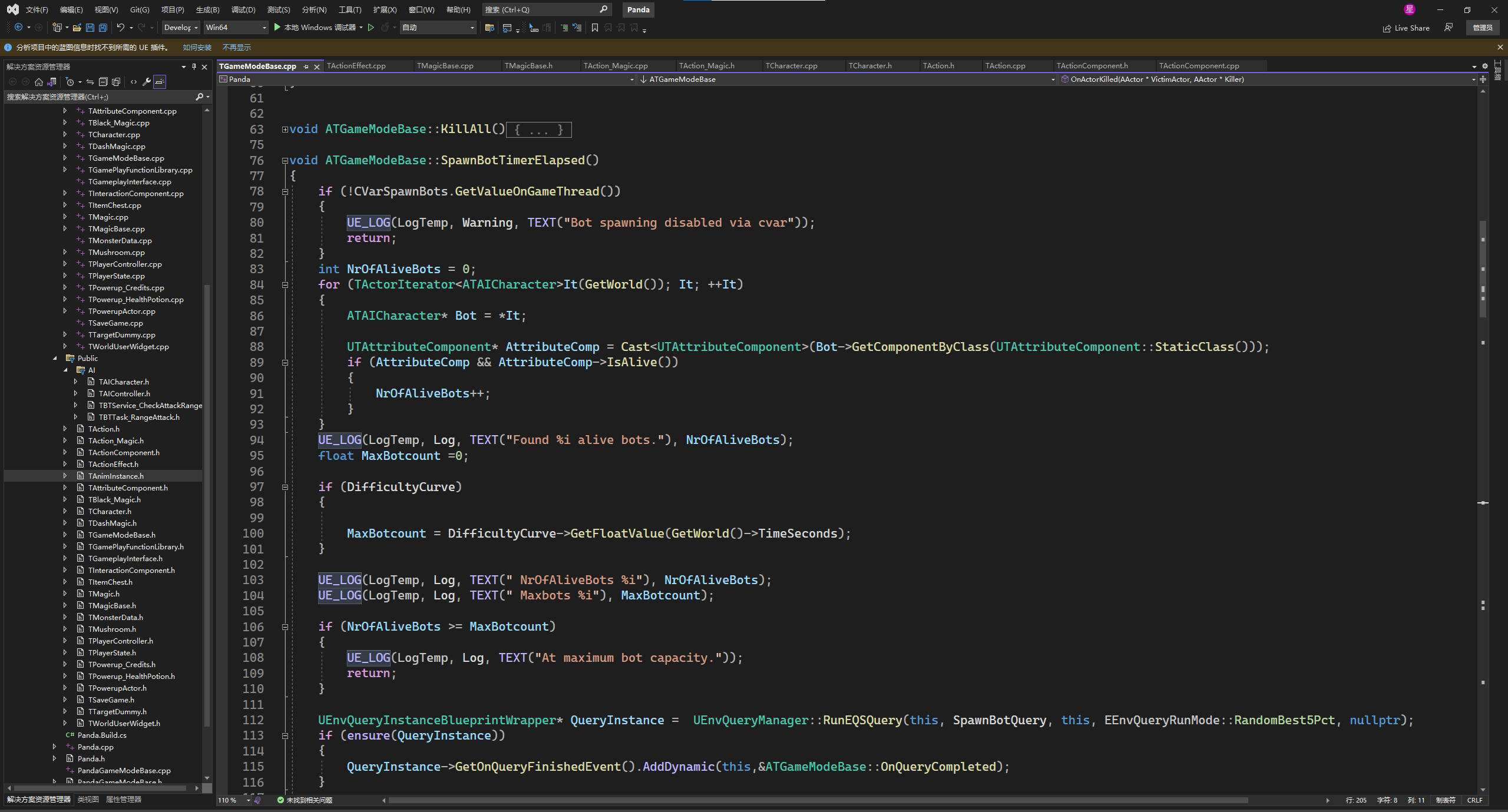
上图是Action Component使用action的实现，通过canstart等判断类确认character中的每一个添加的action是否在运行或者可以运行，再判断是否是服务器可调用，否则传递为Server版本；只需要将传入的FName改变写入character中就可以做到技能完整调用了。



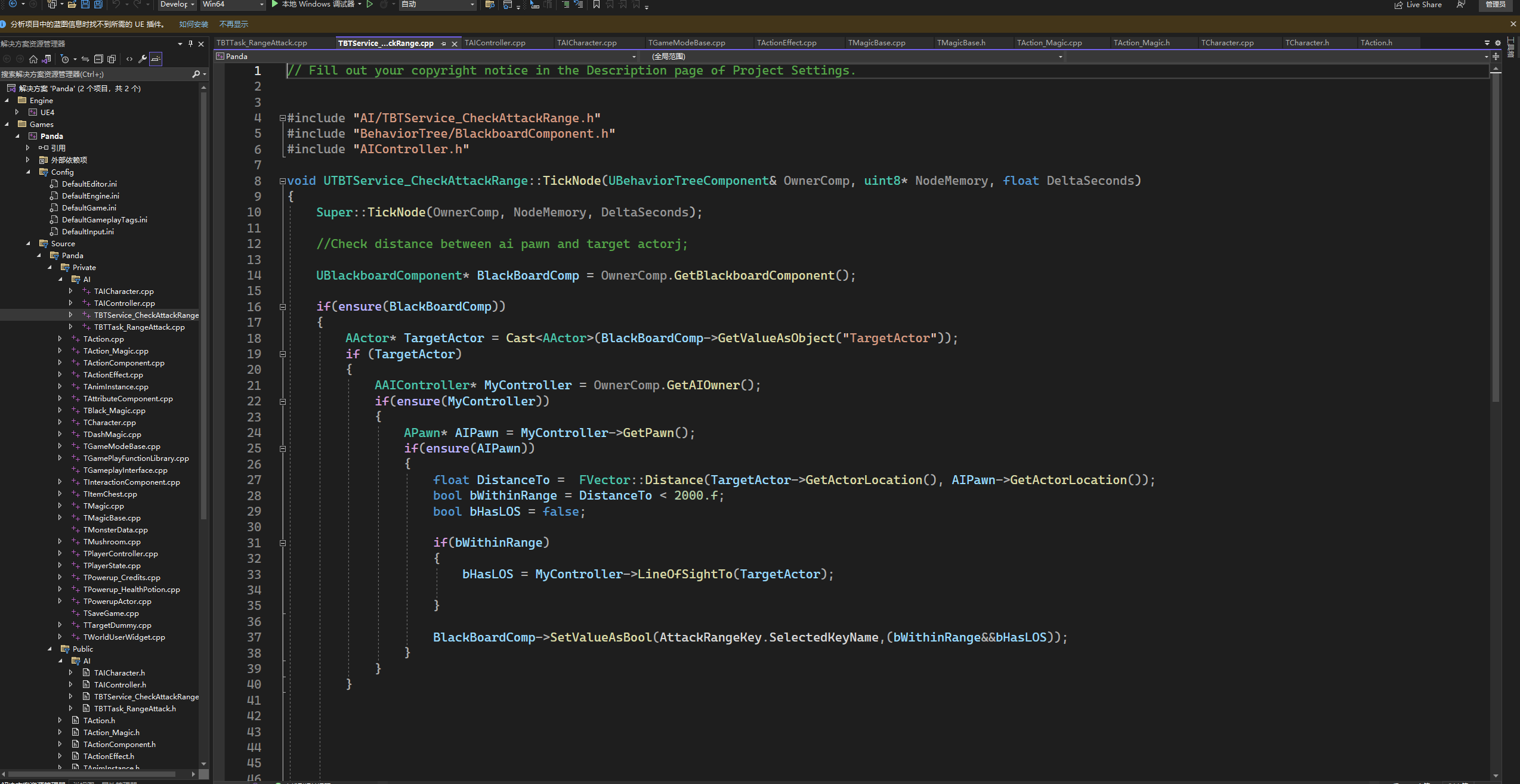
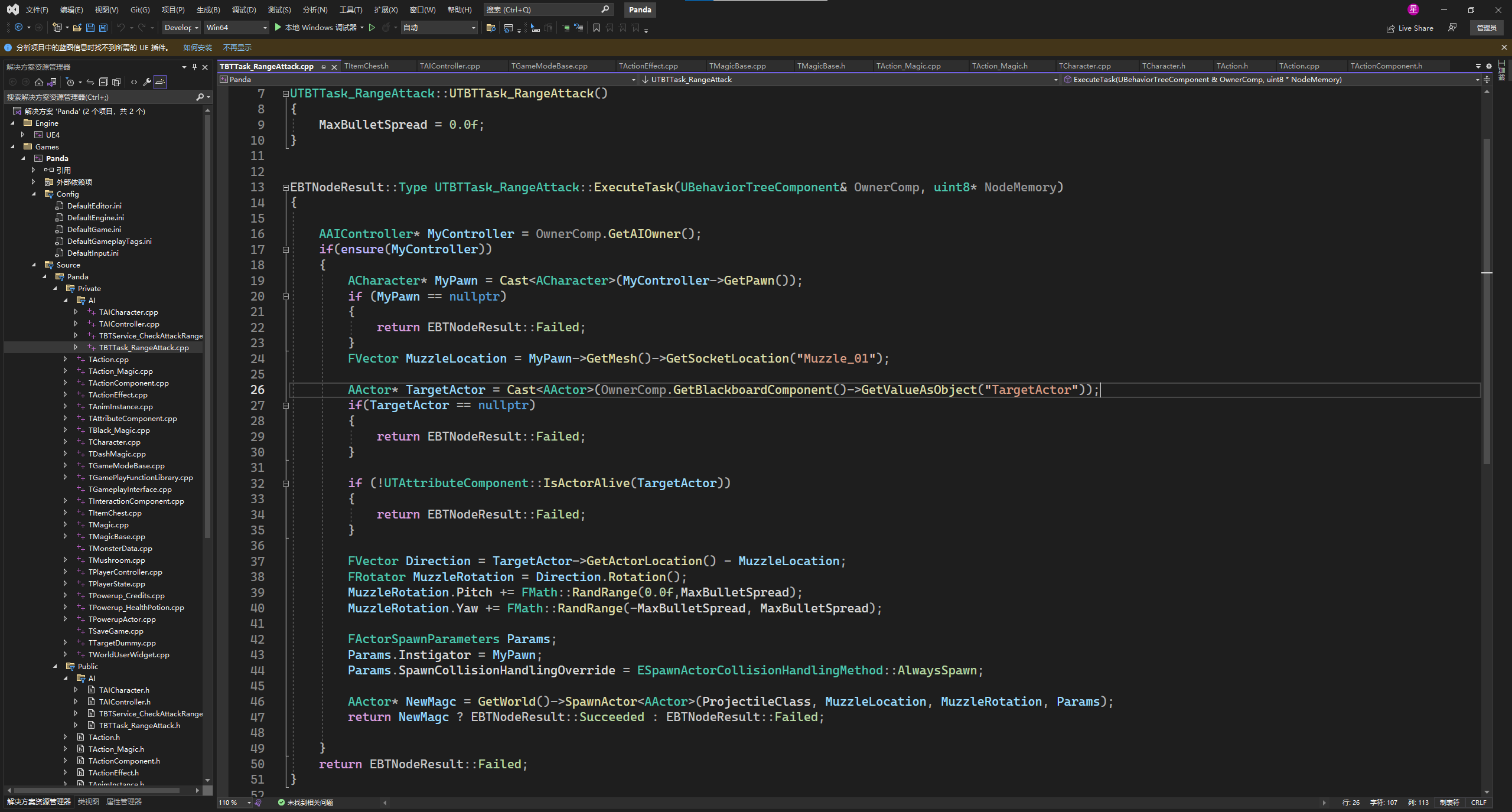
普通攻击的实现，用一个有半径的球体在射线上进行碰撞，找出第一个碰撞点，后续对碰撞点作查询，寻找是否存在attribute组件判断血量的加减和碰撞粒子到静态物体上还是pawn上的使用。



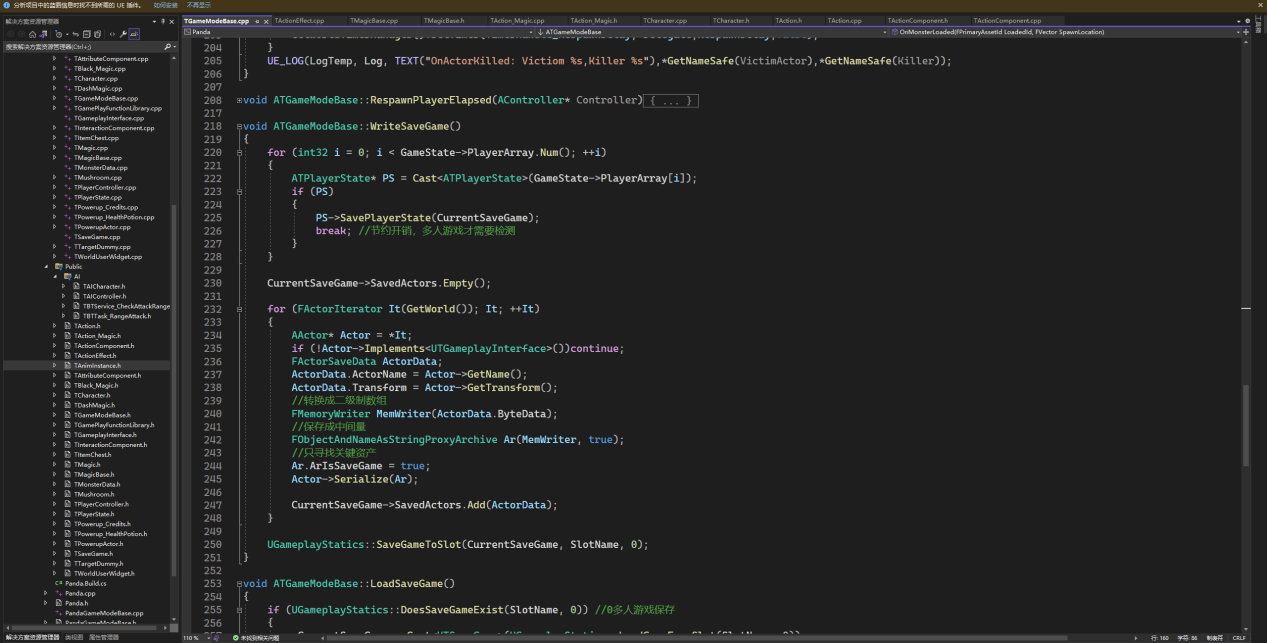
上图是燃烧buff的部分逻辑，只要period重复时间和duration持续时间判断有效继续buff，这里使用了 KINDA\_SMALL\_NUMBER这个虚幻内置变量，当Timer判断period时间小于这个非常小的值结束逻辑。



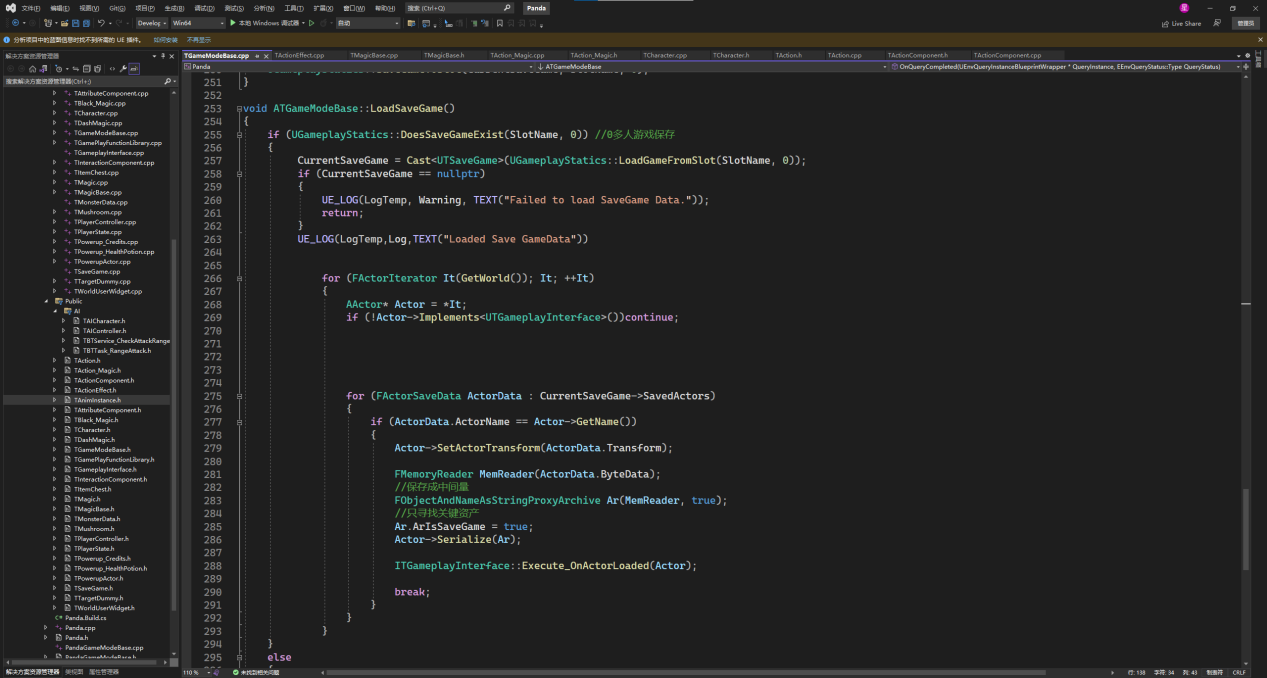
AI生成的逻辑，定义maxbot数量，当数量未达到max数量则持续生成，可在蓝图调整生成的曲线平滑生成数量；编辑器内定义了可生成的地形检查，在正确的地形上才能生成避免生成bug如卡墙，卡空中等



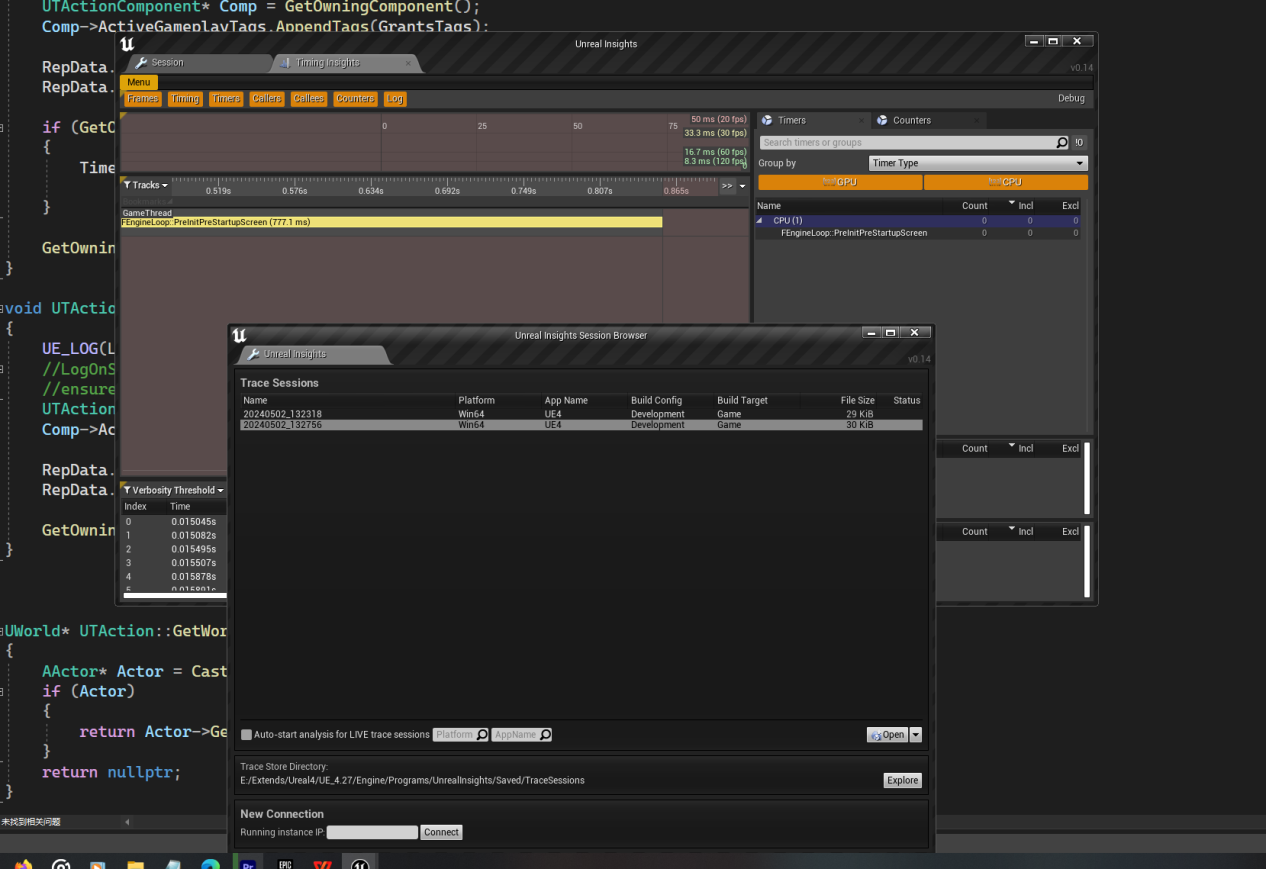
AI的行为逻辑，通过寻找目标后减掉旋转角攻击角色；可以感知攻击受到一定攻击后逃向障碍后。



保存游戏逻辑，将可交互等主要资源挨个保存到目录的一个文件夹中转换成排序好的序列



加载游戏逻辑，依次做同样的检查寻找重要资产，将保存的序列重新写入资产中完成关卡的加载和保存。



UE内置的性能调试