**ClientSimulator设计与实现说明**

ClientSimulator的设计与实现基于MeetingCore以及LoadTest，目标是想实现一个结构清晰，松耦合、易扩展的客户端底层框架，正好借此机会，相当于实现了一个Demo，以下就大的几个方面做简要描述，具体请参考源码。

1. **整体结构**

从解决方案中的工程来看



依赖关系如下：



ClientSimulator是命令行仿真测试客户端，ClientSimulatorUI是MFC界面仿真测试客户端。

VirtualClientMgr管理VirtualClient，VirtualClient的实现基于ClientCore。

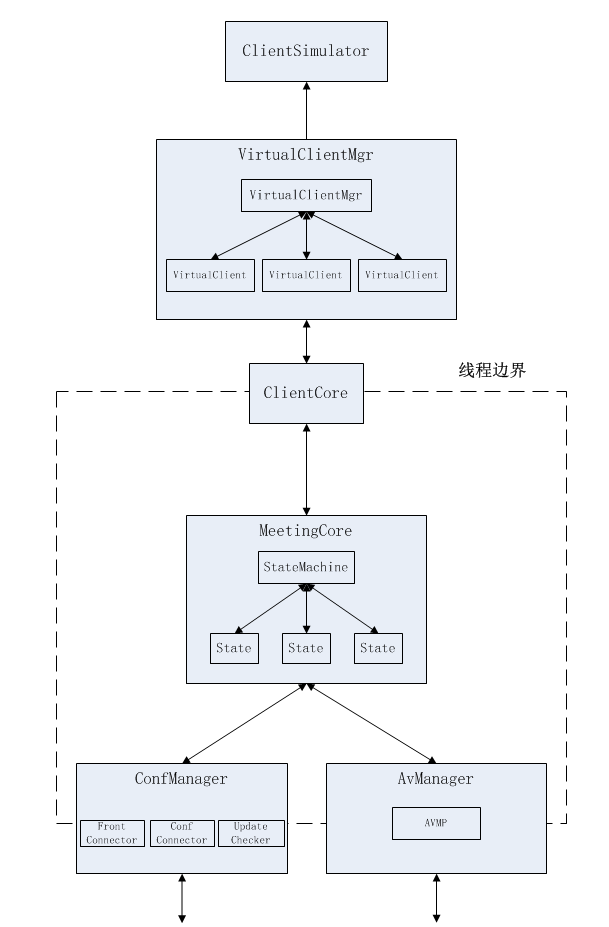
ClientCore相当于老的MeetingCore。

MeetingCore最核心的是状态机，包括状态机的生命周期、状态跳转及小部分会议逻辑。

ConfManager和AvManager分别封装了会议功能和音视频功能，向上屏蔽具体实现和协议。

1. **消息框架思想**

ClientSimulator整体是基于消息框架思想，但是，作为一个库，特别是一个客户端库，它只能属于客户端整体框架的一部分，在客户端框架与库交互界面，实现纯消息交互、异步处理不太现实，有时候反而会使得上层处理更复杂，特别是在需要同步处理的时候。因此，在客户端框架与库交互界面这一层，不是纯消息交互，严格来说是一半消息，一半调用，具体见下图：



图中箭头表示消息流向。实现层面，消息处理节点的消息处理，类似于MFC中MessageMap，只需要在对象中声明要处理的消息，并绑定消息处理函数即可，无需关心消息的转发。

1. **状态机实现**

状态机本身比较简单，但是这里主要考虑状态机的运转控制内置在状态机内部，状态机在接收到外部输入和激励后，产生确定的动作；同时，状态机的跳转方式可能有多种，里面使用策略模式，来定制状态机的跳转方式，使控制与运转分离，易于扩展。

