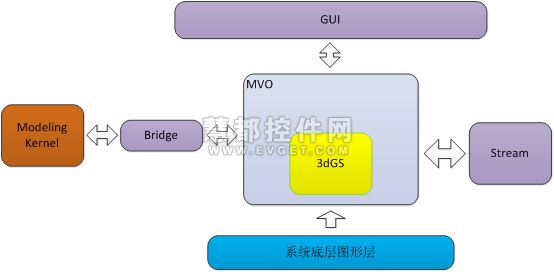
**HOOPS 3D入门教程二：HOOPS模块介绍**

HOOPS是一个模块儿化的开发套件，各个模块功能专一，模块之间的互操作也处理得不错。模块儿化是任何一个大型的开发工具的必由之路，好比国家大 了必须划分成多个省一样。在我这个系列文章中将会涉及到的模块主要有：3dGS，MVO，GUI，Stream还有 Modeling Kernel Bridge。它们之间的关系如下图所示。



3dGS是HOOPS最为核心的模块，其负责HOOPS的绘图及数据管理任务。它屏蔽了不同底层绘图库（OpenGL、DirectX等）接口的不 同，向上提供一套统一的、更加直观且便于调用的接口。3dGS使用纯C语言编写，HOOPS优秀的跨平台以及跨开发工具的特性也正是基于3dGS这一模 块。此外，3dGS还针对特定的底层绘图库有相应的优化，例如针对OpenGL的纹理渲染，3dGS就做了充分的算法优化，而这种优化如果要直接在 OpenGL上做，则需要编程人员具有非常老练的技术，可谓非斲轮老手不可为。我曾经亲自做过测试，拿一个用HOOPS编写的虚拟现实渲染程序和师兄用 OpenGL写的差不多的程序对比，其速度要快上4倍左右（fps，即每秒帧数）。而该师兄在我眼里已经是编程好手了。

MVO是HOOPS另一个重要模块，而且将是我们实际编程中接触最多的一个模块。MVO分别是Model、View、Operator的缩写，即文档、视图和操作算子。MVO有时候也称为MVC，这个C是Controler的缩写。相信用MFC开发过的同学对MVO或者MVC应该不会陌 生，CDocument和CView正是M和V的具体化身。至于O么，在MFC里面则分散在各种消息映射里，此处且不多提，在后续文章中用到的时候可顺便说明。

3dGS和MVO什么关系？它俩的关系跟Windows API和MFC的关系是一样的。做过Windows C/C++编程的人都知 道，Win API实际上可以实现Windows上编程的所有功能（这个不难理解，因为所有Windows上的程序最终都是转化成对若干个最底层的 Win API的调用），但是大家也知道用Win API直接编写大型程序是不现实的，因为这种C风格的程序接口在面对庞大的工程需求时完全是心有余而力 不从。所以后来微软推出了MFC，实际上是在面向对象和开发框架的基础上对底层Win API的封装。3dGS和MVO也是一样。实际上用3dGS可以实 现HOOPS上的所有功能，但是几乎没有人会完全用3dGS去做项目开发（搞研究自己折腾的人除外）。MVO是对3dGS的上层封装，提供了更加友好直观的编程接口，从而使得HOOPS能够适应大型程序开发的需要。

Stream主要负责HOOPS的I/O模块，处理数据流的读写和控制。HOOPS特有的文件类型主要有两种，分别是字符格式的hmf和二进制格式 的hsf。hsf具有非常优秀的特点，它具有高压缩比，支持流式加载，同样的模型，用hsf存储，文件大小仅为常见格式的五分之一左右。流式加载的能力使 得HOOPS能够很好地胜任B/S架构的程序。对于大型模型来说，流式加载是非常关键的，它使得观看者可以像收看流式视频那样，边下载边查看模型，而不必等到整个模型都下载完毕才能够打开浏览。

如果说3dGS、MVO、Stream等模块处理的是和绘图相关的逻辑命令，那么真正把HOOPS包装成一个可执行程序并且在Windows等窗口界面中绘制出图像来则是GUI模块的功劳。HOOPS可以寄宿于多种GUI开发库中，包括MFC、Qt、X11、WPF等等。支持多种GUI框架也是保证 HOOPS跨平台能力的重要条件。在最新的版本中，HOOPS更随潮流，添加了在iOS平台上的支持。不过作为一名业余Android开发者，我非常期待 HOOPS能够在不远的将来增加在该平台上的兼容。

本文最后介绍的一个模块是Bridge。上篇文章中我们也提到了，HOOPS本质上只是一个三维图形渲染框架，虽然它有一定的建模能力，但是估计没 有人会完全用它来进行复杂模型设计。而实体造型是几何建模引擎所擅长的，因此在用HOOPS进行CAD/CAE/CAM开发时常会搭配一个几何造型引擎， 所谓尺有所短寸有所长，在各领域选择合适的工具对于大型系统来说是非常必要的。HOOPS可以和多个实体造型内核相连接，例如Spatial的ACIS和 Siemens的Parasolid。