**[FloodLight使用感受](http://www.cnblogs.com/mahuan2/p/4703260.html)**

一个使用java语言编写的基于Openflow协议的SDN控制器。

基本架构同webserver一样，有一个维护交换机连接信息的底层模块，当有交换机同控制器连接时，floodlight会将此连接保存到连接池中，并通过NIO的方式同交换机进行通信。

控制器还包含其他交换机数据包存储处理模块，比如网络拓扑模块（计算最短通信路径）、链路发现模块（使用LLDP协议发现网络的拓扑的变化）、设备管理模块（负责完成MAC地址同IP地址之间的映射）、存储模块（负载存储每台交换机的存储状态，比如保存的流表项等）、RESTAPI模块（负责通过RESTAPI的形式，完成流表项的管理）、webui模块（提供一个web界面，查询控制器状态）。

floodlight通过配置信息决定加载哪些模块，模块同时会注册其会理哪些消息。

Floodlight总体上有两个主要功能：

1. 处理交换机连接并将OpenFlow消息转换为事件（events）以使其他模块能够监听处理。比如发现交换机、链路失效、端口失效、接收到数据包等消息。
2. 决策某些特定OpenFlow消息（例如，PacketIn, FlowRemoved, PortStatus等），并将其按顺序转发到配置的监听模块。然后每个模块决定是否处理此消息并传递到下一个监听者或者停止处理消息。

能够处理的事件消息，需要处理OpenFlow消息的模块实现IOFMessageListener接口。

事件中有交换机的连接信息，可通过对交换机进行操作，同时还可通过全局方法获取其他交换机的信息，并对其进行操作。

总体感觉，其与webserver的模式很像。

1. 有维护客户端连接的模块，java中的servlet管理所有的用户连接（控制器维护交换机的连接，netty）
2. 有处理客户端请求的模块，客户端将request发送到服务端，服务端将response发送给客户端（控制器将交换机的变化、请求信息抽象成事件，发送给相应模块处理，模块处理时可以对交换机进行操作）

不足：目前控制器的自定义模块网络编程模式依然需要编写大量的代码，是一种面向计算机语言的程序编写模式，而不是面向网络的。导致实现自定义模块的编写工作量巨大，开发周期长，并且不易调试并发现流表项决策的错误。