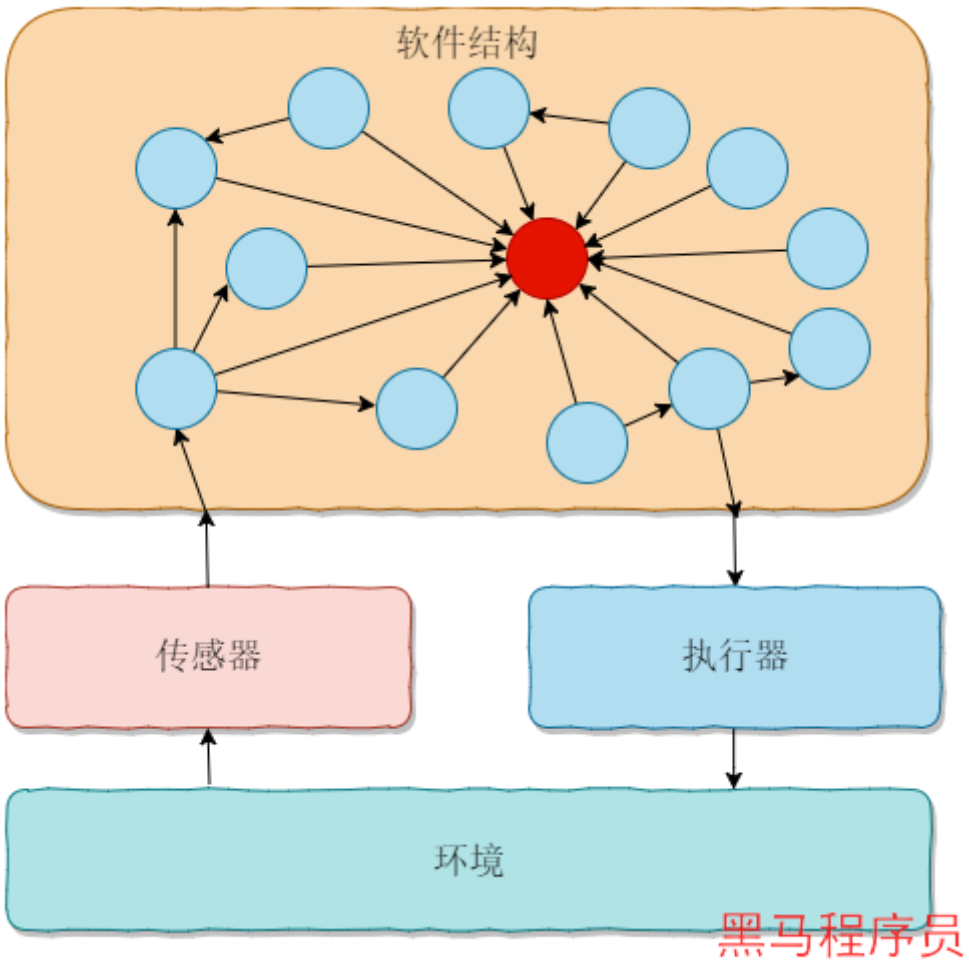
系统基本架构



总的来说，整个ROS的项目在部署运行过程中，分为几个部分：

- 环境
- 执行器
- 传感器
- 软件结构

环境

包含了软件环境和硬件环境。通常一个ROS的机器人，需要一个开发主板用于运算和控制操作。

例如，我们用树莓派作为开发板，用来作为这个机器人的运算和控制操作，那么这个就是软件所运行的硬件环境。开发板上通常需要有操作系统，所装的操作系统属于系统环境。当然开发板上可能会外接一些其他硬件，如摄像头。那么这些硬件也属于硬件环境。

执行器

执行器主要作用是给操作系统发送一些指令，通过指令取控制硬件操作。例如发送指令给开发板，让开发板外接的陀机转动，驱动物体运动。执行器更像是硬件驱动的输入。

传感器

在硬件环境中，可能会安装一些环境探测的硬件，例如温度感应器。这些硬件感应器会将感觉的数据发布出来，传感器其实扮演的就是这些硬件发布数据的传播者，更像是硬件驱动的输出。

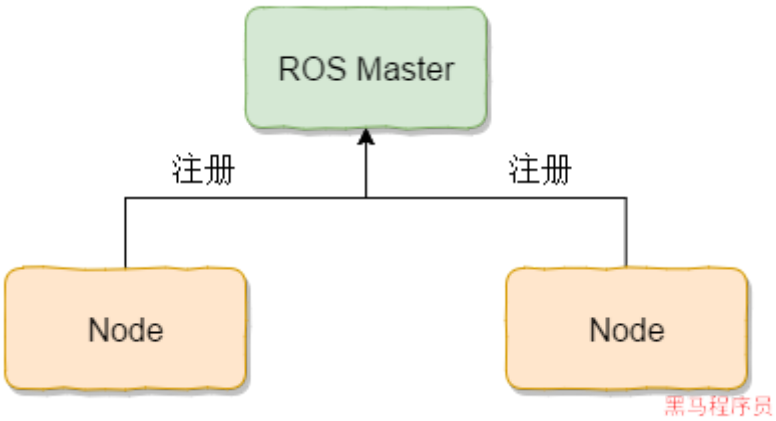
软件结构

这里所说的软件结构是指，ROS机器人运行时部署的软件及软件间的关系。

机器人运行起来后，内部会有很多单元程序运行，每个单元程序做很小的事情。有的小程序负责读取硬件驱动传递的数据，有的负责处理读取的数据，有的负责判断数据，有的负责发出指令....

总之，ros将复杂的程序分解成了很小的一部分，每部分干很少的活，每个部分还可以复用。

ROS 软件结构组成



ROS Master

- 管理Node节点间进行通讯的
- 每个Node节点都需要到Ros Master中进行通讯

通`roscore`命令可以启动ROS Master，启动节点前，必须启动ROS Master。

```
# 启动ROS Master
roscore
```

ROS Node

- 具备单一功能的可执行程序
- 可以单独编译，可执行，可管理
- 存放在package中

ROS 哲学

1. Peer to peer

点对点的设计。

- Node节点单元
- 采用了分布式网络结构
- 节点间通过RPC + TCP/UDP进行通讯

2. Distributed

分散协同式布局。可以将ROS同时部署到多台机器上，让多台机器进行通讯。

3. Multi-lingual

多编程语言的支持。

- 可以采用python，c++，lisp等语言进行开发。
- 遵循协议进行编码，和编程语言无关

4. Light-weight

组件工具包丰富。ros提供了丰富的开发工具包。

5. Free and open-source

免费并且开源。

- BSD许可。可修改，可复用，可商用。
- 开源使软件进步