1. **ROS的定义**

ROS是一个适用于机器人的开源的元操作系统。它提供类似操作系统所提供的功能，包括硬件抽象、底层设备控制、常用函数的实现、进程间消息传递以及包管理。它也提供用于获取、编译、编写、跨计算机运行代码所需的工具和库函数。它主要采用的是松耦合点对点进程网络。

个人理解：ROS不是一个真正的操作系统，是类操作系统，是属于中间层的。ROS主要的目的是将硬件层抽象出来，留出接口方便开发。ROS可以分为框架、工具、功能和社区四大部分，其中框架主要是指分布式管理和进程管理，工具有仿真软件Gazebo、可视化工具Rviz和Rqt，功能有控制、规划、定位和建图等功能，社区指的是软件包管理，可以调用别人的软件包进行项目开发，一般可以在wiki或者github上找到资源。

ROS有四大优点：**1、松散耦合的机制方便机器人软件框架的组织；2、非常便利的数据记录、分析、仿真工具，方便调试；3、最丰富的机器人功能库，方便快速搭建原型；4、学界和产业界的标准，方便学习和交流。**使用一句话总结，使用ROS，能够方便迅速地搭建好机器人原型。其实，ROS这四大优点刚好一一对应框架、工具、功能和社区这四大部分。

1. **ROS安装（从底层的架构来分析为什么这样安装，需要时间）**

注：apt-get install 安装二进制，无法修改；

从github下载的软件包或者自己编写的需要编译才能安装。

1. **ROS名词解释**

1.package(功能包)：功能包是ROS中组织软件的主要形式，一个功能包包含多个可执行文件。

2.metapackage(功能包集)：功能包集是实现某种功能的多个功能包的集合。

3.node(节点)：节点是一个可执行文件，程序文件只有转成成可执行文件才可以运行。

4.topic(主题)：节点通信方式的一种，通过publish-subscribe来实现，异步和单向的通信。

5.message(消息)：topic通信的数据类型，可以理解为两者通信的协议。

6.service(服务)：节点通信方式的一种，通过client-server来实现，同步和双向的通信。

1. **ROS工程结构（文件系统级）**

ROS工程结构直观上是指图形化界面上的文件夹以及里面的内容，本质上是讲ROS的框架。

首先是创建一个新的Catkin工作空间（以前叫Rubuild），也叫Catkin Workspace，这个文件夹里面有三个文件夹，分别是src、bulid和devel，这三个文件夹分别存放pkg软件包、中间缓存文件以及目标文件，其中src里面的package文件是最重要的，也是最基本的组织形式，一个package里面包含两个必备的文件，分别是CMakeList.txt和package.xml。CMakeList.txt文件定义了编译的规则，直接规定了这个package要依赖哪些package，要编译生成哪些目标，如何编译等等流程；package.xml里面是package的自我属性描述，在以前的Rubuild里面叫manifest.xml，其实两者是一样的。除此之外，还有scripts(\*.py、\*.cpp)、include(\*.h)、src文件夹，这三个文件夹是与C++和Python有关的。有些情况下还会有config(param)、launch文件夹和一些自定义的通信格式（msg、srv、action）。这就是一个完整的ROS工程结构，其中是src、build和devel是必须有的，而src里面的CMakeList.txt和package.xml也是必须有的，其他的文件是看项目的需要而建立的。

整体的文件说明参考下图4-1所示：



图4-1 文件的说明

关于ROS的编译过程参考下图4-2所示：

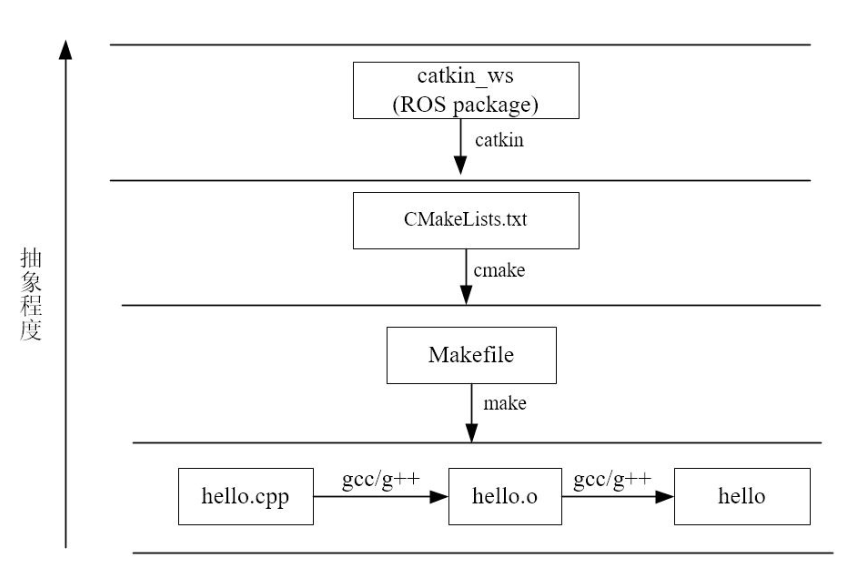


图4-2 编译的过程

那么对工程结构有了一个整体的认识后，我们需要如何创建和管理这些文件，这就需要用到我们的一些常用的指令来进行管理，这些指令跟LINUX里面的指令有点类似，大部分是在LINUX指令前面加上ros。

其中有六个指令比较常用，具体的使用格式可以查，大概有关印象。如下所示：

1.catkin\_creat\_pkg:创建一个package

2.roscd：跳到某个pkg 的路径下

3.rosed：编辑某个pkg下的某个file文件

4.rosls：罗列当前包的信息

5.rosdep：安装某个pkg的依赖包

6.rospack：rospack find package\_name是寻找某个包

rospack list 显示所有的包

对于ROS来说，package是最基本的组织形式，那么这些package这么多，应该怎么来组织呢，这个时候就有了一个功能包集的概念，以前叫Stack，现在叫Metapackage，其实说白了就是把功能相似的功能包组织起来放在一起，然后起个名字就成了Metapackage。

1. **ROS通信架构（计算图级）**

ROS的通信架构有一个总的管家，名叫Master(节点管理器)，所有节点之间的通信都要

通过向它注册才能进行，所以要先启动Master。启动Master需要用roscore命令，执行这个命令同时也启动rosout和parameter server。

在通信架构里面当然是通信最重要，那么就涉及到两个概念，一个是通信的对象——节点(node),另一个是通信的方式——4中方式(Topic、Service、Actionlib和Parameter Service)。

才能进行通信。

Topic和Service的区别如下图5-1：

图5-1 Topic和Service的对比

Parameter Service也是通信方式的一种，但是比较特殊，主要是因为参数服务器是节点存储参数的地方、用于配置参数，全局共享参数。参数服务器使用互联网传输，在节点管理器中运行，实现整个通信过程。它更加静态，有点类似字典的形式。对于参数服务器的维护有三种方法：1.命令行维护；2.launch文件维护；3.node源码。**个人理解：参数服务器是方便开发者在后续调参的时候用到的，这样就不需要每次都要改代码，可以提高开发的效率。**

Actionlib是ROS中一个很重要的库，类似service通信机制，actionlib也是一种请求响应机制的通信方式，actionlib主要弥补了service通信的一个不足，就是当机器人执行一个长时间的任务时，假如利用service通信方式，那么publisher会很长时间接受不到反馈的reply，致使通信受阻。当service通信不能很好的完成任务时候，actionlib则可以比较适合实现长时间的通信过程，actionlib通信过程可以随时被查看过程进度，也可以终止请求，这样的一个特性，使得它在一些特别的机制中拥有很高的效率。个人理解：action相比service主要是在于有一个实时的回馈过程，即feedback。这个在代码中的体现也是多了一段feedback的代码，其他的跟service类似，可对比来学习。

1. **ROS工具**
2. 仿真工具——Gazebo
3. 可视化工具——Rviz(3D可视化工具)、Rqt(集成图像交互界面)
4. 命令行工具——rosbag(记录和回放数据流)
5. 专用工具——moveit
6. **ROS编程库——roscpp、rospy**
7. **TF、URDF**
8. **参考链接**
9. [https://www.jianshu.com/p/0869eec39a3b](https://www.jianshu.com/p/0869eec39a3b（机器人操作系统ROS(一)-初识）) (机器人操作系统ROS(一)-初识)
10. <https://www.leiphone.com/news/201701/qKhqf4Rq4C6Bw0Gt.html> (机器人程序设计之如何正确入门ROS)
11. <https://sychaichangkun.gitbooks.io/ros-tutorial-icourse163/content/> (中国MOOC《机器人操作系统入门》课程讲义)