重庆大学
CHONGQING UNIVERSITY

机器人操作系统
Robot Operating System

——第二章：ROS基础

江 涛、熊祖名、张博文

重庆大学 自动化学院

2023年5月5日

本章目标

1、深入理解ROS操作系统的工作空间，学习基本操作

2、掌握ROS操作系统通讯机制，掌握ROS通讯程序编写，完成话题、服务、参数间的收发传递

3、深入理解roscore，rosrun的指令逻辑

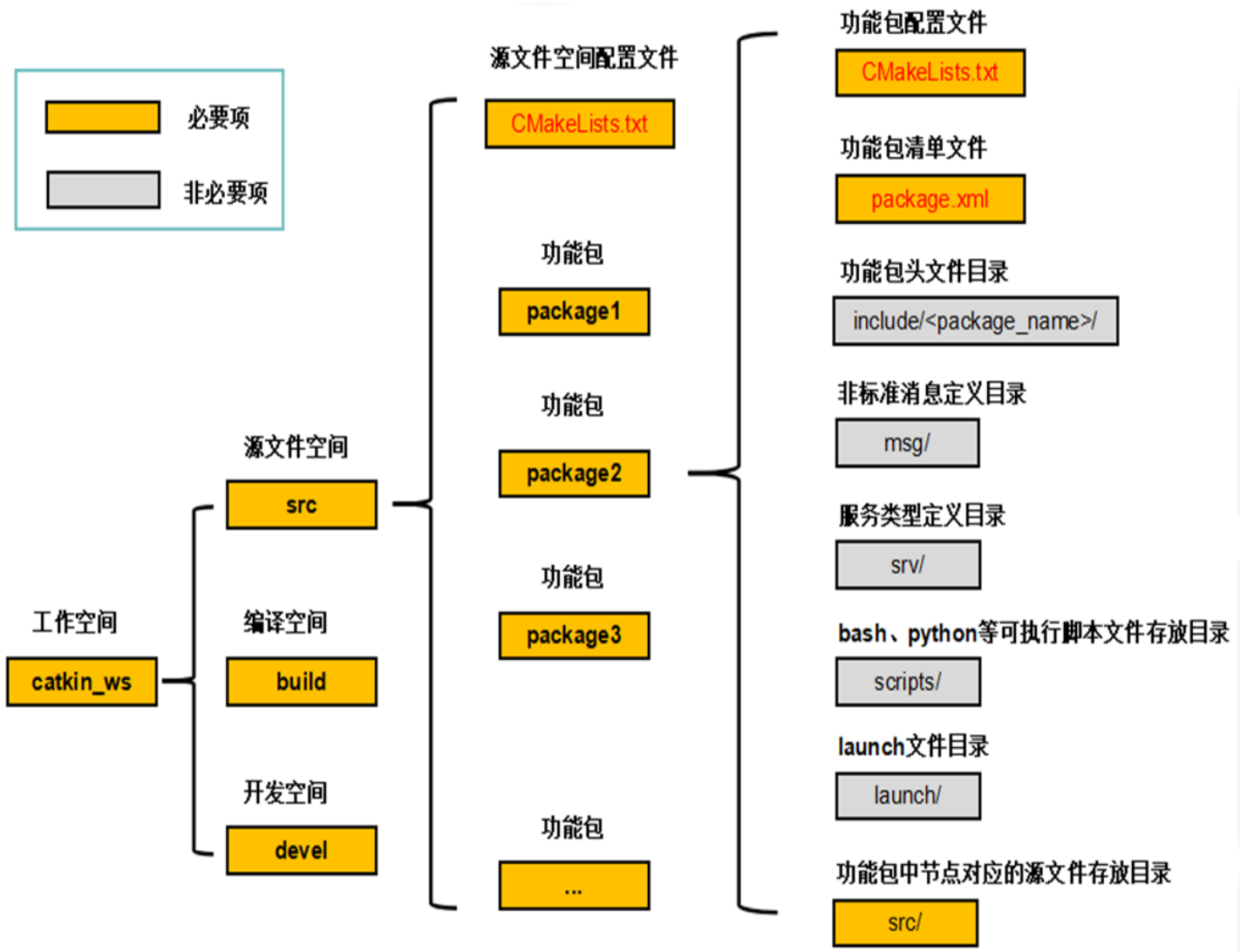
4、掌握ROS工具TF、QT、launch的使用

一、工作空间：结构内容

重庆大学 CHONGQING UNIVERSITY

workspace主要包含三部分：

- src：代码空间，存放和管理ROS软件包的源代码，每个子文件夹是一个pakage
- build：编译空间，存放catkin_make命令编译ROS软件包时生成的中间文件
- devel：开发空间，用于存放catkin_make命令编译ROS软件包时生成的目标文件、库文件、可执行文件等

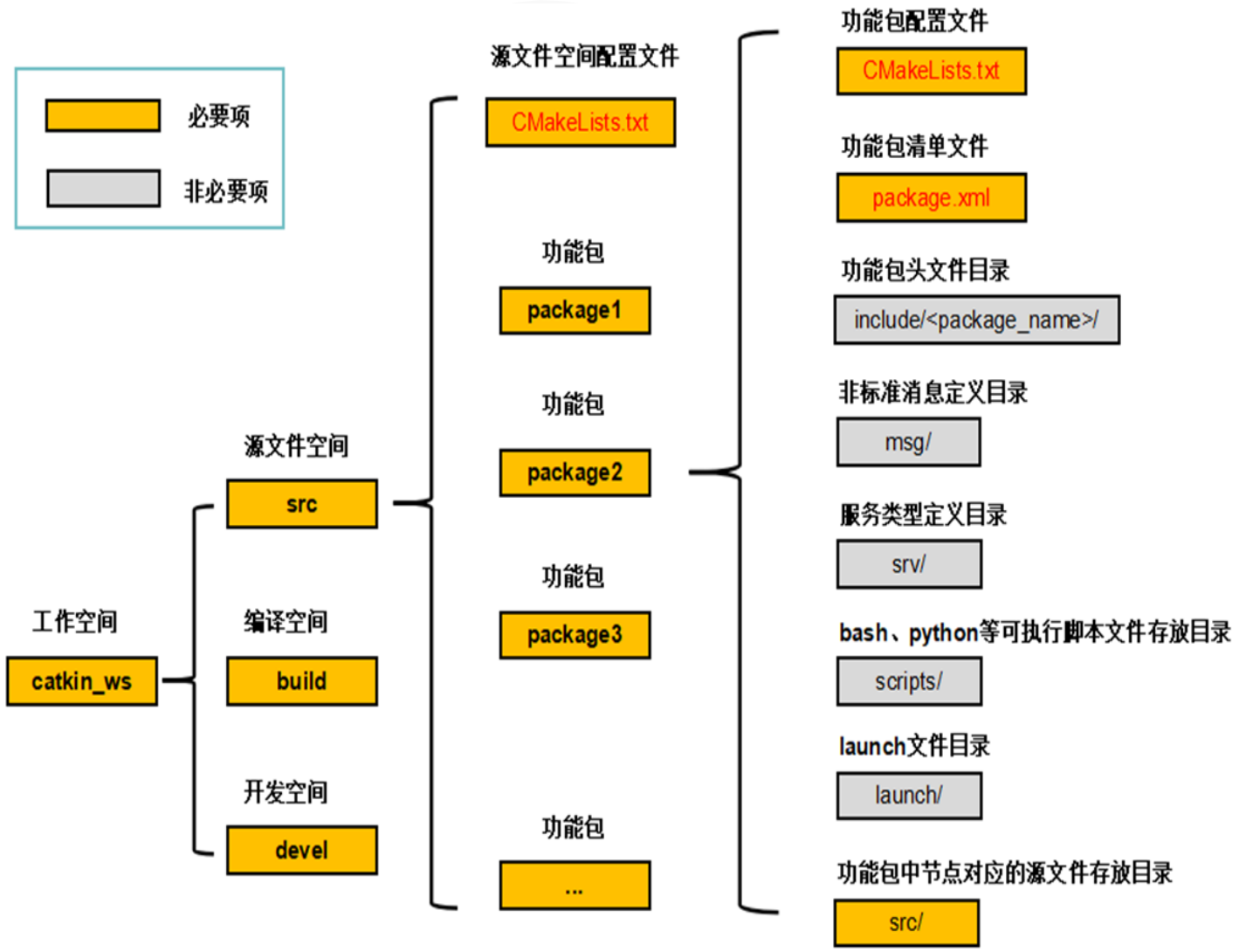


一、工作空间：结构内容

重庆大学 CHONGQING UNIVERSITY

packge 目录：

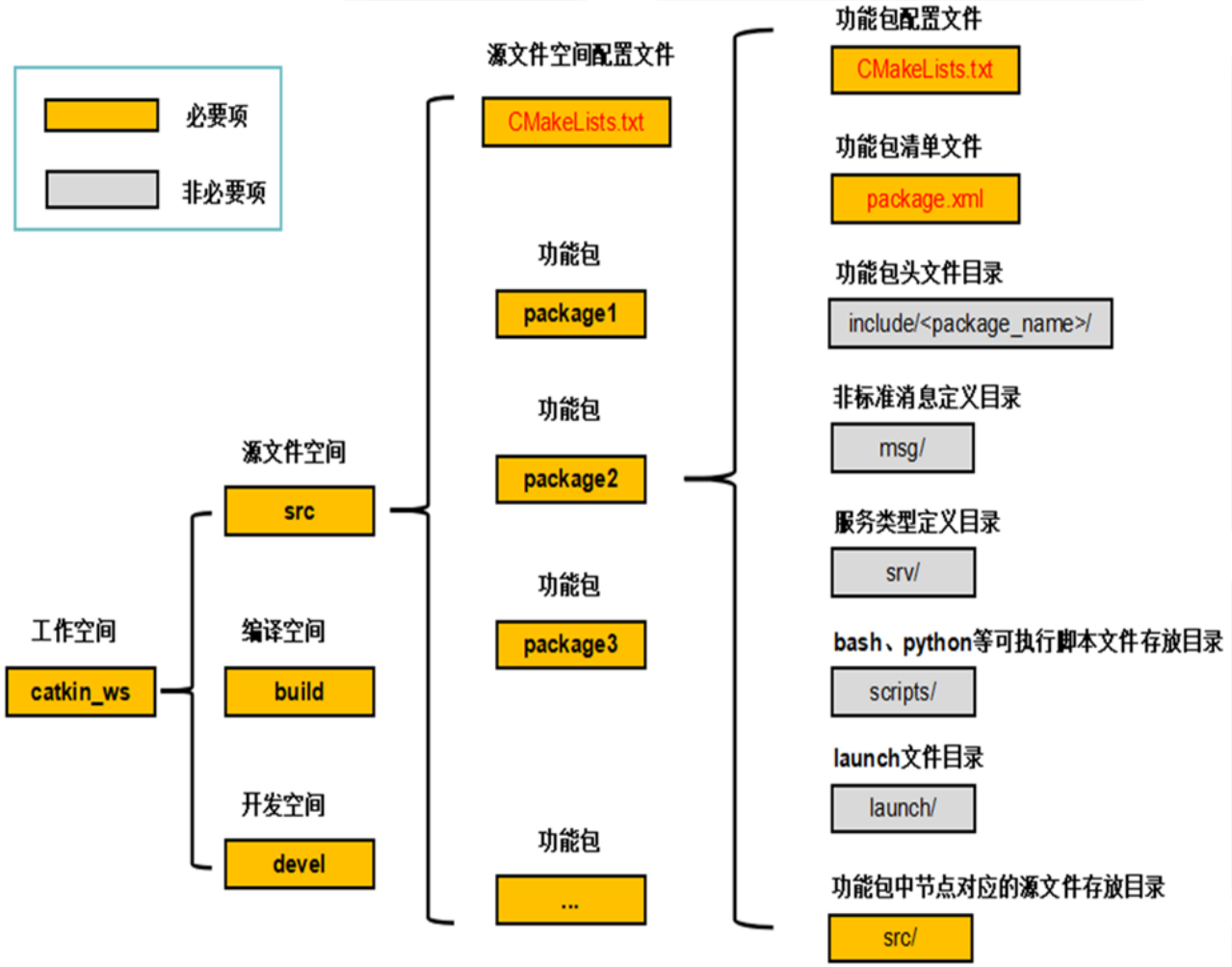
- CMakeLists.txt：用于CMake描述包的构建过程,包含了需要编译的**源文件、链接的库、生成的目标文件**等
- package.xml：描述了包的名称、版本、依赖关系等元信息，用于**CMake解析包**的设置
- src目录：存放ROS软件包的**源代码**，如**.cpp**
- Include：存放源代码的**头文件**



一、工作空间：结构内容

packge 目录：

- scripts：用于存放ROS软件包的**可执行脚本**，如.**py**等文件，这些脚本可以**直接运行**，用于启动该ROS包的相关功能
- launch：包含所有的**launch文件**，用于启动该ROS包相关的**节点、参数服务器**等
- msg：如果该包定义了ROS的**消息类型**，该目录下将包含所有**消息的定义文件**
- srv：如果该包定义了ROS的**服务类型**，该目录下将包含所有**服务的定义文件**



一、工作空间：创建工作空间（实践）

重庆大学 CHONGQING UNIVERSITY

1.创建工作空间：

```
$ mkdir -p name_ws/src  #name_ws为工作区的名字，可根据工程需求更改
$ cd name_ws/src   # 进入创建的文件夹
$ catkin_init_workspace  #将当前文件夹初始化为ROS的工作空间
```

2.创建功能包：

```
$ catkin_create_pkg name_pkg roscpp rospy std_msgs  #name_pkg为创建的功能包名，后面跟的是各个依赖
```

3.编译工作空间：

```
$ cd ..  # 回到name_ws工作空间 (或者cd name_ws)
$ catkin_make  # 创建编译空间开发空间
```

4.设置并检测环境变量：

```
$ source devel/setup.bash
$ echo $ROS_PACKAGE_PATH
```

一、工作空间：工作空间的查看与更改*

在工作空间的查看与修改中常常用到下列命令：

命令名	功能
rospack	rospack是对package管理的工具
roscd	roscd 类似于Linux系统的 cd，但 roscd 可以直接 cd 到ROS的软件包
rosls	rosls 类似于Linux指令的 ls，可以直接 ls ROS软件包的内容
rosdep	rosdep 是用于管理ROS package依赖项的命令行工具

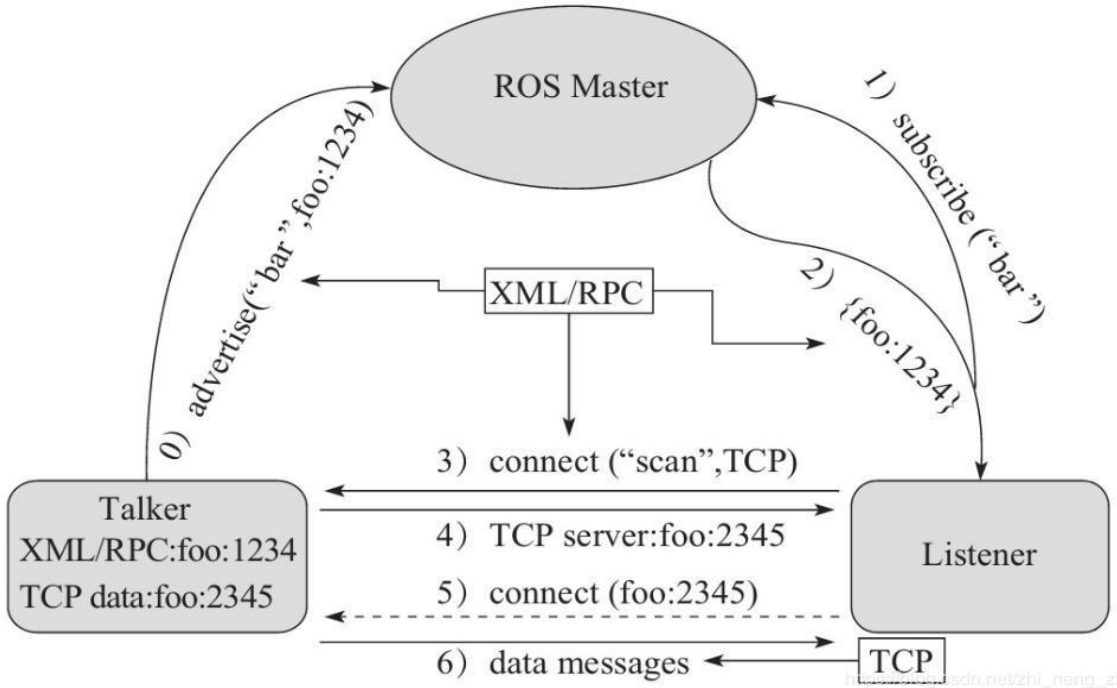
例1：工作空间中功能包的更改

- 功能包的添加：首先定位到该工作空间的根目录下，再通过“rosinstall” 命令或者“apt-get” 等命令下载所需功能包。
- 功能包的删除：可以通过“rospack remove <功能包名称> ”命令进行删除。
- 最后，删除或者添加完功能包后，一定要执行“ catkin_make --force-cmake”命令。

二、通讯编程：ROS通信要素

- 节点管理器（ros master）：命名、注册、管理和控制节点，跟踪和记录话题/服务通信
- 节点（node）：执行具体任务的功能单元
- 消息（message）：话题通信所传递的数据结构，由msg文件定义
- 话题（topic）：话题通信中订阅节点和发布节点交换消息的桥梁
- 服务（service）：服务通信中交换消息的媒介：服务端节点启动服务，客户端节点提交服务请求，服务端节点处理请求，返回被处理的消息，由srv文件定义
- 参数管理器：存储系统参数和用户定义的参数，通过参数管理器实现参数的增删改查

二、通讯编程：话题通信理论模型



0）发布节点**通过XML-RPC注册**话题名称bar和端口地址foo：1234

1）订阅节点注册话题名称bar

2）节点管理器master向订阅节点发送端口地址foo：1234

3）Connect()是一个连接函数，scan是tcp的端口扫描方式

4）发布节点通过**TCP**发布另一个端口地址：2345

5）发布节点和订阅节点相连接

6）发布节点和订阅节点通过tcp传递消息

步骤0-4是使用XML-RPC协议，步骤5-6是使用TCP协议。XML-RPC用于远程过程调用，但是它的冗长编码方式不适合高带宽和低延迟的任务。数据流传输协议TCPROS，它传输的原始数据几乎不需要解析，这种设计方式可以减少延迟，增加带宽。

二、通讯编程：话题通信案例

要求：编写**发布订阅**实现，发布方以10HZ(每秒10次)的频率发布文本消息，订阅方订阅消息并将消息内容打印输出。

1 创建发布（Publish）节点

- 声明发布的消息类型
- 组织被发布的数据，并编写逻辑发布数据
- 发布消息

2 创建订阅（Subscribe）节点

- 声明订阅消息类型
- 处理订阅的消息(回调函数)
- 设置循环调用回调函数

二、通讯编程：话题通信案例

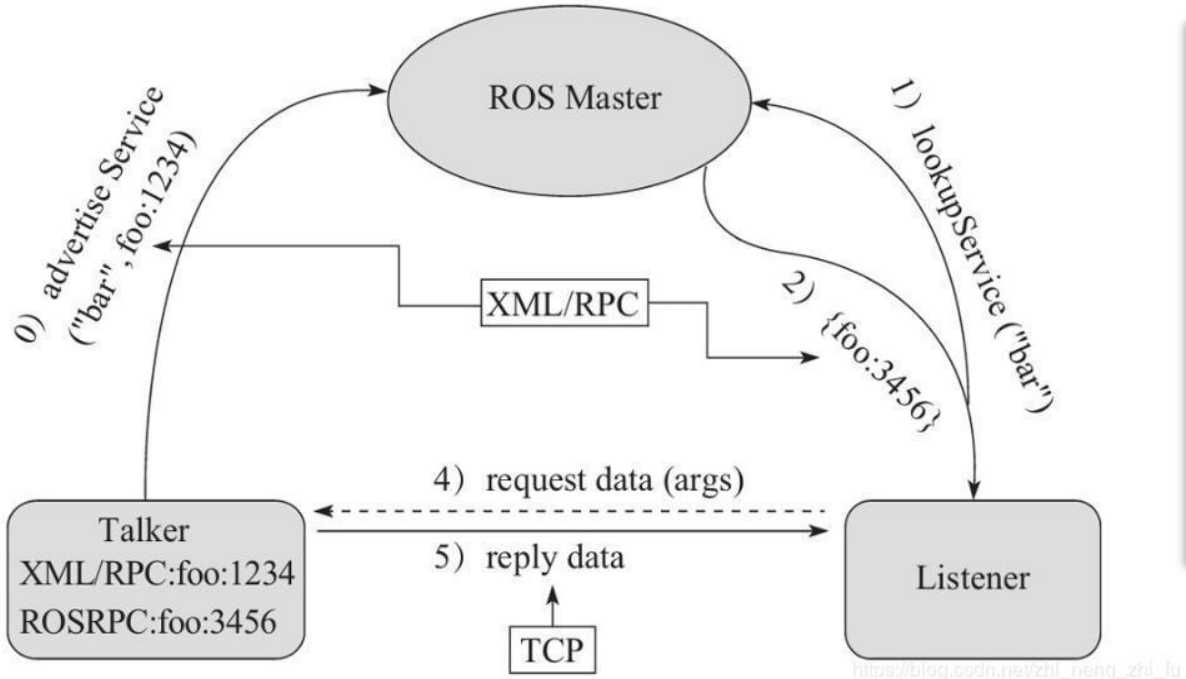
3 修改CMakeLists.txt文件

- add_executable(pub src/pub.cpp)
- add_executable(sub src/sub.cpp)
- target_link_libraries(pub ${catkin_LIBRARIES})
- target_link_libraries(sub ${catkin_LIBRARIES})

4 运行节点，获取运行结果

```
>> roscore ##打开ros核心
>> rosrun topic_com pub ##rosrun 功能包名 可执行文件名（运行发布节点）
>> rosrun topic_com sub ##rosrun 功能包名 可执行文件名（运行订阅节点）
```

二、通讯编程：服务通信理论模型



0) 服务端节点(Talker)通过XML-RPC注册节点名称bar和端口地址foo:1234

1) 客户端节点(Listener)注册节点名称bar

2) 节点管理器(ROS Master)发送端口地址foo:3456

3) 客户端节点（Listener）发送数据请求

4) 服务端节点响应请求

服务通信和话题通信的**区别**：

话题是单向的，不需要等待服务端上线，数据的实时性高，发布方可以快速地发布数据，但是服务通信是双向的，客户端发送请求后，服务端有响应，可以得知服务端的处理结果。频率较低，强调服务特性和反馈的场景一般使用服务实现

二、通讯编程：服务通信案例

要求：编写**服务通信**实现，在客户端提交两个正整数到服务端，服务端求和并将结果发送给客户端，并创建两者间的数据载体

1 定义srv文件：add.srv

```
int32 num1
int32 num2
##客户端请求发送的两个数据
---
int32 sum
##服务器相应发送的数据
```

2 修改package.xml文件：

```
<bulid_depend>message_generation</bulid_depend> ##添加编译依赖
<exec_depend>message_runtime</exec_depend> ##添加执行依赖
```

二、通讯编程：服务通信案例

重庆大学 CHONGQING UNIVERSITY

3 修改CMakeList.txt文件：

- 在find_package(catkin REQUIRED COMPONENTS）内添加message_generation
- 在add_service_files() 添加add.srv
- 在catkin_package() 添加message_runtime
- 修改可执行文件add_executable(server src/server.cpp)，add_executable(client src/client.cpp)
- 修改目标链接库target_link_libraries(server ${catkin_LIBRARIES})， target_link_libraries(client ${catkin_LIBRARIES})

4 编译：生成头文件add.h和addRequest.h, addResponse.h：

- Ctrl+shift+B编译
- 在devel下的include/sev_client的文件夹内包含目标头文件

二、通讯编程：服务通信案例

5 编写服务端节点

- 创建服务对象
- 编写回调函数
- 回调函数处理请求并响应

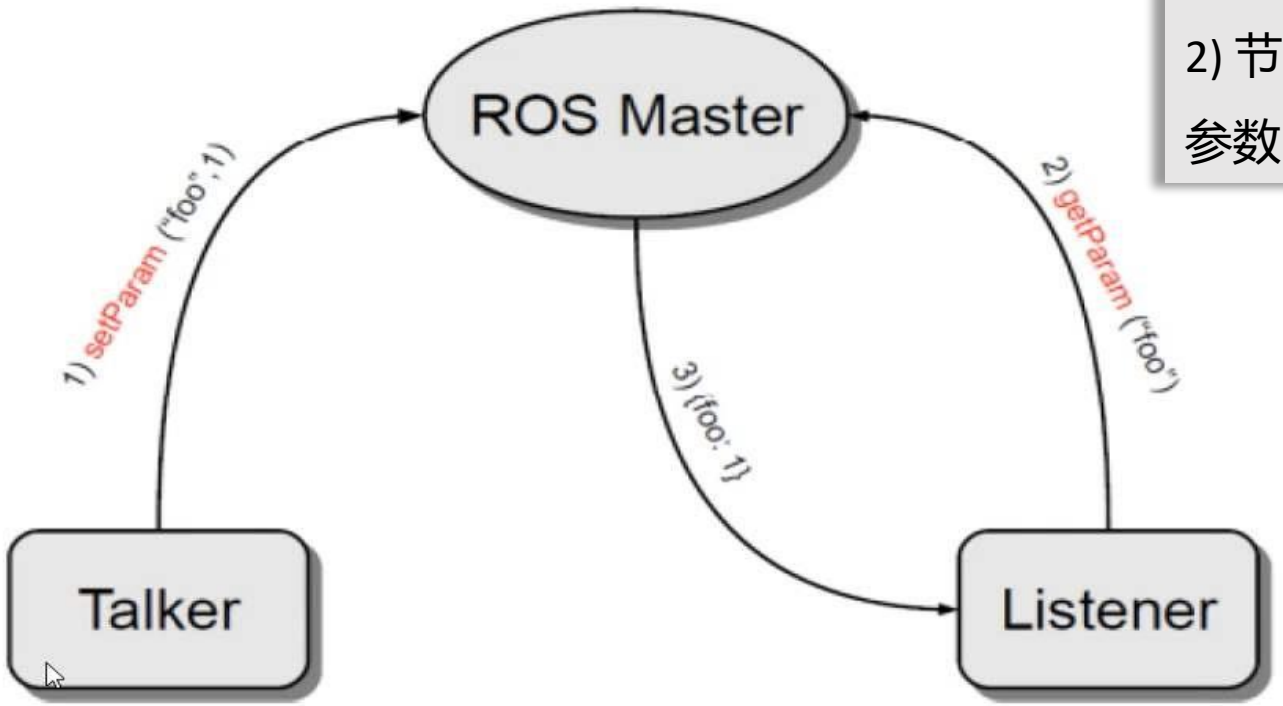
6 编写客户端节点

- 传递参数
- 创建客户对象
- 组织请求数据
- 发送请求

7 编译，运行

```
>> roscore ##打开ros核心
>> rosrun service_com server ##运行服务端节点
>> rosrun service _com client num1 num2 ##运行客户端节点
```

二、通讯编程：参数服务器理论模型



0) 参数设置节点(talker)向节点管理器(ROS Master)发送参数(参数名为foo，值为1)

1) 参数调用节点(Listener)向参数服务器发送查找请求

2) 节点管理器(ROS Master)向参数调用节点(Listener)发送参数（“foo”,1)

二、通讯编程：参数服务器案例

要求：编写程序，完成参数的增添、删除、修改、查找操作，实现参数服务器数据的增删改查

1 编写参数设置节点

- 增加或修改参数：nh.setParam(“键”,“值”)或ros::param::set(“键”,“值”)

  > > nh.setParam("radius","0.8")

- 查找参数：param(键,默认值)，如果存在键，则返回键的值，如果不存在，则返回默认值

  > > double radius1 = nh.param(“radius”,0.4);

  getParam(键,存储结果的变量)，存在,返回 true,且将值赋值给存储变量，若键不存在，那么返回值为false，且不为参数2赋值

  > > bool flag = nh.getParam("radius",radius1);

- 删除参数：del(“键”)，或者deleteParam("键"), 根据键删除参数，删除成功，返回 true，否则(参数不存在)，返回 false

  > > bool flag = nh.deleteParam(“radius");

二、通讯编程：参数服务器案例

2 修改CMakeLists.txt文件

- add_executable(param src/param.cpp)
- target_link_libraries(param ${catkin_LIBRARIES})

3 运行结果

>> roscore

>> rosrun param_com param

>> rosparam list

一 工作空间

二 通讯编程

三 指令深入

四 高效工具

五 内容小结

三、指令深入：roscore

roscore是ros系统的一些基础节点和程序的集合，必须运行 roscore 才能使 ROS 节点进行通信。

当启动roscore，系统会启动三个部分：

ros master（节点管理器）：管理节点

ros parameter server（参数服务器）：存储或处理参数

rosout logging nodes（日志节点）：收集其他节点的数据并写入日志节点文件，topic/rosout被节点rosout订阅，节点rosout向topic/rosout_agg发布消息

启动roscore：

... logging to /home/sasa/.ros/log/c7d84de8-eeb1-11eb-a4d5-48e2447b179d/roslaunch-sasa-ThinkPad-E550-5731.log

Checking log directory for disk usage. This may take awhile.

在/home/XXXX/.ros/log 创建了一个日志（log）文件，包含从ros节点中收集到的日志，根据日志信息可以进行调试。

三、指令深入：roscore

started roslaunch server http://XXXX-ThinkPad-E550:39331/ros_comm version 1.12.17

启动名为roscore.xml的launch文件。当launch文件启动的，系统自动启动ros_master和ros_parameter参数服务器，且显示了端口的ROS参数服务器的地址和ros_comm版本（ros_comm是ROS中的一个metapackage，它主要包括了ROS通信相关的package）

PARAMETERS

* /rosdistro: kinetic

* /rosversion: 1.12.17

ROS发行版本和具体版本号。

三、指令深入：roscore

NODES

auto-starting new master

process[master]: started with pid [5741]

ROS_MASTER_URI=http://XXXX-ThinkPad-E550:11311/

启动ros_master，并且ros_master节点使用ROS_MASTER_URI的环境变量。URI（Uniform Resource Identifier，统一资源标识符）就是在IMS网络中IMS用户的“名字”，也就是IMS用户的身份标识。ROS_MASTER_URI 是一个 IP 地址和 rosmaster 将要监听的端口的组合，可以根据 roscore 命令中给定的端口号更改端口号，当需要使用远程调用的时候，需要在系统中修改地址位远程调用的地址。

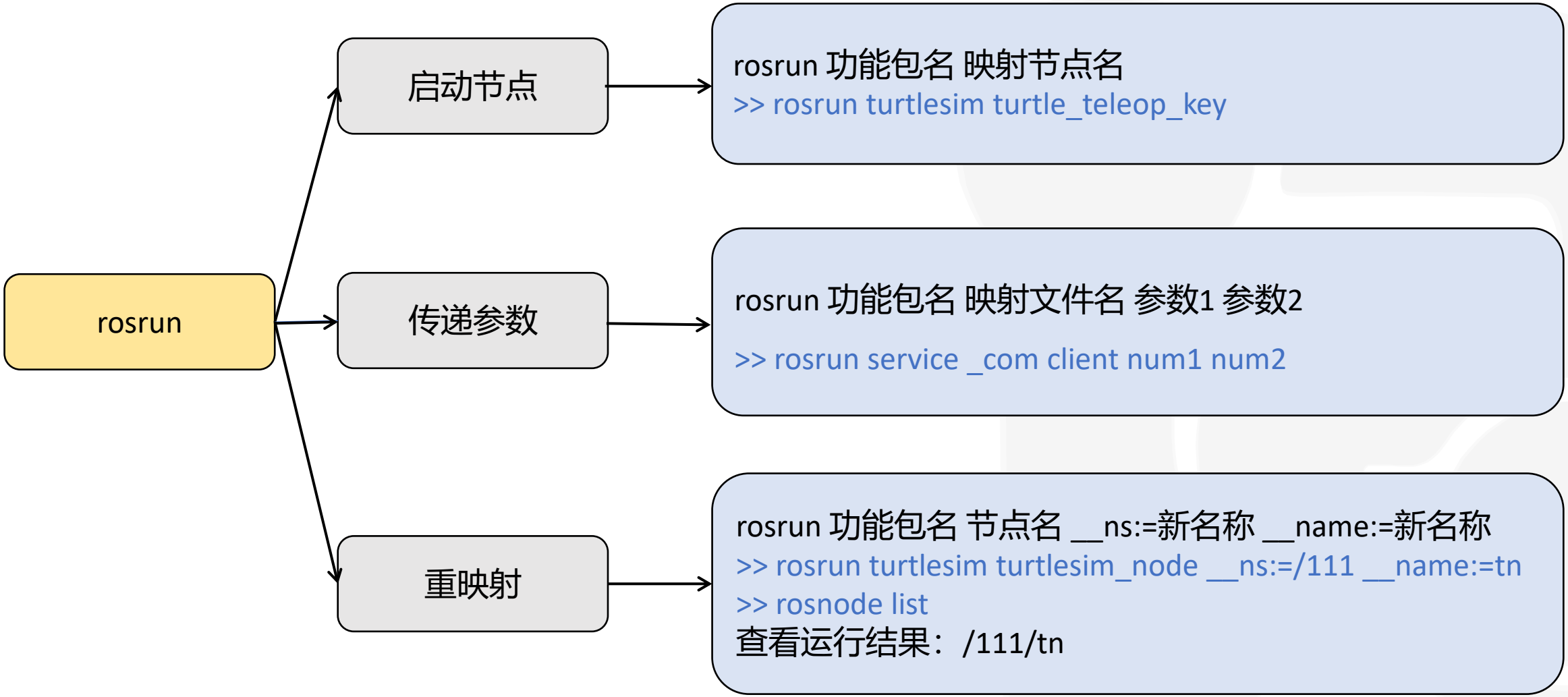
setting /run_id to c7d84de8-eeb1-11eb-a4d5-48e2447b179d

process[rosout-1]: started with pid [5754]

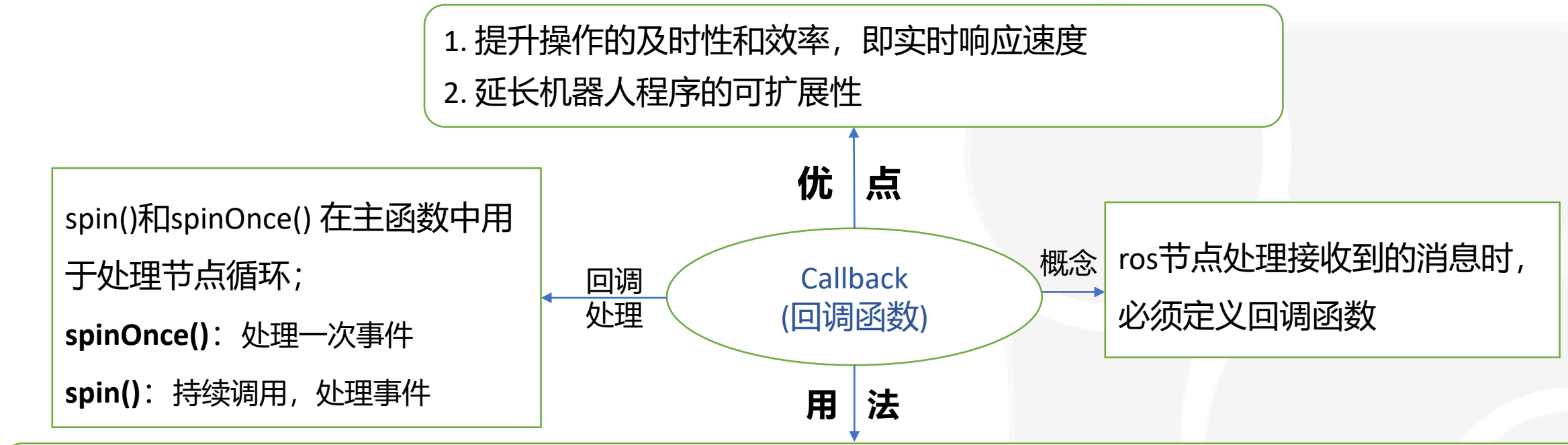
started core service [/rosout]

rosout节点启动，开始订阅/rosout话题，并发布到/rosout_agg上。

三、指令深入： rosrun



三、指令深入：callback




```
void dowork(const demo02_talker_listener::Person::ConstPtr& person_p){
   // code
}
```

四、高效工具：launch

重庆大学 CHONGQING UNIVERSITY

1.为什么要使用launch文件封装工具?

- 有时候需要启动的**节点较多**
- 节点与节点之间大多**存在依赖关系**
- roslaunch可以一次启动多个节点，提高了应用程序的**可靠性**和**可用性**

2.roslaunch包概述：

- roslaunch是ROS提供的**启动工具**，能够一次性把多个节点按照**预先的配置**启动起来
- roslaunch的调用：**roslaunch <package_name> <launch_file_name>**
- roslaunch配置文件使用**XML语言**编写，文件以**.launch**为扩展名，放在package的launch文件夹下

http://wiki.ros.org/roslaunch

四、高效工具：launch

launch文件语法：

标签名	作用
<launch>	launch文件的根节点，用于包含其他节点
<node>	用于配置一个ROS节点
<param>	用于配置ROS参数
<include>	用于包含其他的launch文件
<env>	用于设置ROS环境变量
<echo>	用于显示信息
<timer>	用于配置定时器
<remap>	用于重新映射话题名称
<arg>	用于定义和传递参数
<group>	用于分组节点

launch文件基本格式：

```
<launch>
  <!-- 节点配置 -->
  <node pkg="包名" type="节点类型" name="节点名称" output="输出等级">
    <!-- 节点参数 -->
    <param name="参数名称" value="参数值" />
  </node>
  <!-- 参数配置 -->
  <param name="参数名称" value="参数值" />
  <!-- 包含其他的launch文件 -->
  <include file="其他launch文件的路径" />
  <!-- 设置ROS环境变量 -->
  <env name="ROS环境变量名称" value="ROS环境变量值" />
  <!-- 显示信息 -->
  <echo message="要显示的信息" />
  <!-- 定时器 -->
  <node pkg="包名" type="节点类型" name="节点名称">
    <remap from="话题名称" to="新话题名称" />
    <param name="参数名称" value="参数值" />
    <timer period="定时器周期" />
  </node>
</launch>
```

四、高效工具：launch

例题1：

- 创建一个**launch**文件，在**Gazebo仿真环境**中启动Turtlebot3机器人**模型**，并通过**键盘控制**机器人**移动**。

➢ 思路：

- 使用**<include>标签**，将TurtleBot3机器人的空白世界仿真环境包含进来。

  launch文件的路径：**/launch/turtlebot3_empty_world.launch**

- 使用**<node>标签**，启动**turtlebot3_teleop**节点，完成键盘控制机器人运动功能。

  该节点来自ROS软件包turtlebot3_teleop，类型是turtlebot3_teleop_key

四、高效工具：*QT

1.什么是QT工具

- 概念：ROS基于 QT 框架，针对机器人开发提供了一系列可视化的工具，这些工具的集合就是rqt
- 作用：可以方便的实现 ROS 可视化调试，并且在同一窗口中打开多个部件，提高开发效率，优化用户体验。
- 组成：rqt 工具箱组成有三大部分：

- rqt——核心实现，开发人员无需关注
- rqt_common_plugins——rqt 中常用的工具套件
- rqt_robot_plugins——运行中和机器人交互的插件(比如: rviz)

2.rqt常用的插件

- rqt_graph：用于可视化显示计算图
- rqt_plot
- rqt_console：是 ROS 中用于显示和过滤日志的图形化插件。
- 启动：在 rqt 的 plugins 中添加，或在终端通过指令启动（不同插件的指令不同，可在ROS Wiki中查看）。

四、高效工具：tf

1.为什么要使用tf坐标工具?

- 功能：在 ROS 中实现不同坐标系之间的点或向量的转换。
- 优势：

- 适用广泛（适用于各种类型的数据）、灵活性高（支持多种坐标系转换方法）、易于使用（提供简单易用的API）、高效性能（使用高效的计算方法快速完成坐标系转换）、可视化支持（提供可视化工具）

2.tf坐标变换要素：

- 坐标系：是用来描述一个点在空间中位置的系统。
- 坐标变换：是指将一个坐标系中的点映射到另一个坐标系中的过程。
- tf树：ROS中所有的Frame构成了一个树状结构的tf树，其中每一个Frame都有一个唯一的父Frame和多个子Frame。在tf树中，每个Frame的坐标变换都是相对于其父Frame的。
- tf变换的发布和订阅：在ROS中，通过tf库可以实现对tf变换的发布和订阅。
- tf坐标变换的计算：在ROS中，可以通过tf库提供的API来进行tf坐标变换的计算。

http://wiki.ros.org/tf2

四、高效工具：tf

3.tf_ros库

- 功能：是ROS中用于处理tf变换的库，提供了一些用于发布、订阅和转换tf变换的工具和类。
- 主要的类如下：

- Buffer: tf2_ros库中最重要的类之一，用于存储和查询tf变换。可以使用tf2_ros::Buffer类来创建一个Buffer对象，并使用其中的方法来查询和转换tf变换。
- TransformListener: 用于订阅tf变换并将其转换为机器人坐标系中的变换。
- TransformBroadcaster: 用于发布tf变换。
- StaticTransformBroadcaster: 用于发布静态的tf变换。
- TransformStamped: 用于在ROS中传递tf变换。
- Transform: 用于表示tf变换中的平移和旋转。
- Quaternion: 用于在ROS中表示旋转。
- transformStampedMsgToTF()和transformStampedTFToMsg()函数：用于在ROS消息和tf变换之间进行转换。

四、高效工具：tf

4.坐标msg消息：

- 在坐标转换中常用的 msg为

  geometry_msgs/TransformStamped

  geometry_msgs/PointStamped

- geometry_msgs/TransformStamped（用于传输坐标系相关位置信息）：

- 可在命令行键入:**rosmsg info geometry_msgs/TransformStamped** 查看其具体构成。

```
std_msgs/Header header
string child_frame_id
geometry_msgs/Transform transform
```

- **geometry_msgs/PointStamped**（用于传输某个坐标系内坐标点的信息）：

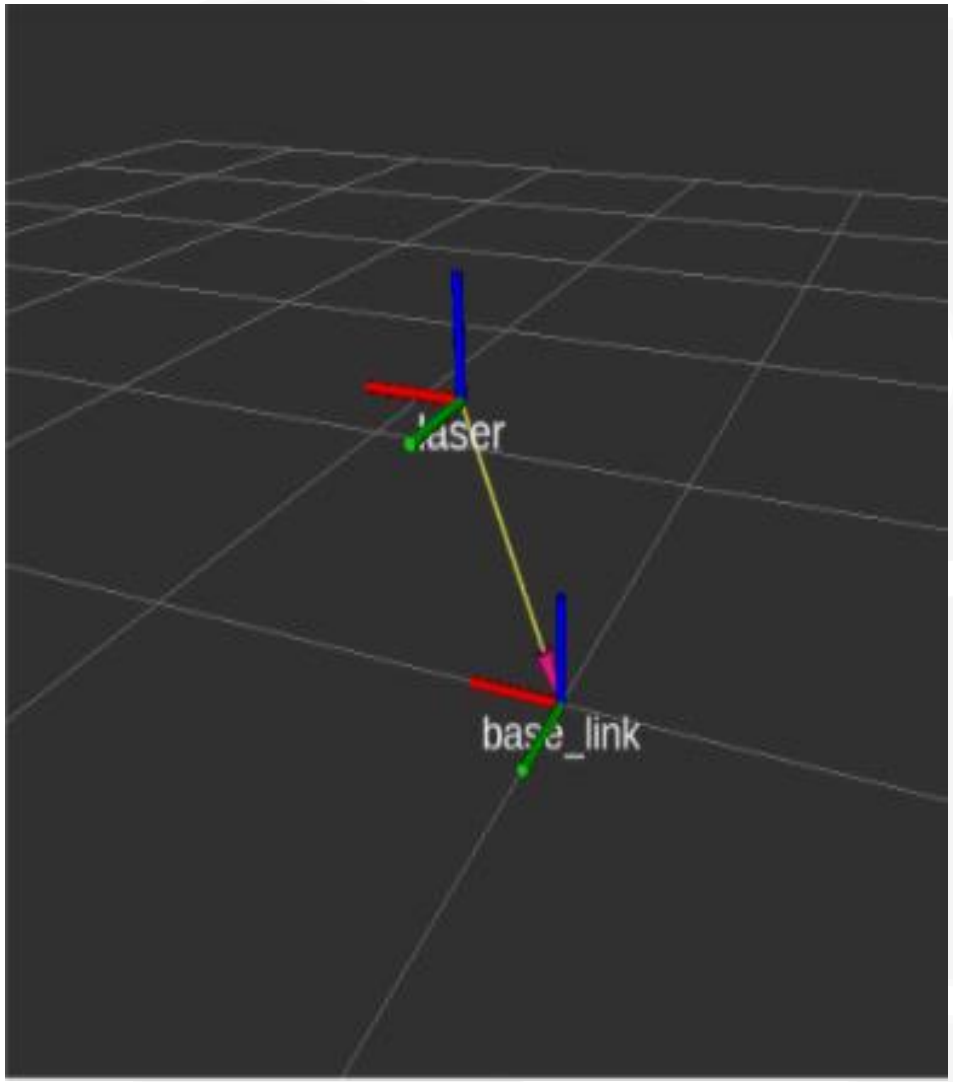
- 可在命令行键入:**rosmsg info geometry_msgs/PointStamped** 查看其具体构成。

```
std_msgs/Header header
geometry_msgs/Point point
```

四、高效工具：tf

例题2:

- 有laser和base_link两个坐标系，且laser**坐标系相对于base_link坐标系**的位置关系为：x： 0.2； y：0.0； z：0.5。已知base_link**原点相对于世界坐标系**的坐标为(2.0 3.0 5.0)，问laser**坐标系原点相对于世界坐标系**的坐标为多少?



➢ 分析：

- 坐标系相对关系，可以通过发布方发布
- 订阅方，订阅到发布的坐标系相对关系，再传入坐标点信息
- 借助 tf 实现坐标变换，并将结果输出

四、高效工具：tf

重庆大学
CHONGQING UNIVERSITY

C++实现：

1.创建功能包：

添加功能包依赖：tf2；tf2_ros；tf2_geometry_msgs；roscpp ； rospy std_msgs；geometry_msgs

2.发布方：

……

3.订阅方：

……

4.执行

```
>> roscore
>> rosrun tf01_static tf01_static_pub
>> rosrun tf01_static tf02_static_sub
```

一 工作空间

二 通讯编程

三 指令深入

四 高效工具

五 内容小结

五、内容小结

1、深入理解ROS操作系统的工作空间

2、掌握ROS通讯程序编写，完成话题、服务、参数间的收发

3、深入理解roscore，rosrun指令逻辑

4、掌握ROS工具包launch、 TF的使用

感谢聆听！