蘇州大學

SOOCHOW UNIVERSITY

关于ROS及ROS基础知识分享

诞生背景

机器人是一种高度复杂的系统性实现。机器人设计包含了机械加工、机械结构设计、硬件设计、嵌入式软件设计、上层软件设计等，是各种硬件与软件集成。

诞生背景

机器人体系是极其庞大的，复杂程度非常高，以至于没有任何个人、组织甚至公司能够独立完成系统性的机器人研发工作。
所以诞生出一种合适的策略：让研发者专注于自己擅长的领域，其他模块直接复用相关领域更专业研发团队的实现，自身的研究也可以被他人复用，这样可以大大提高机器人的研发效率，尤其是随着机器人硬件越来越丰富，软件库越来越庞大，复用性和模块化开发需求愈发强烈，“ROS”就是在这样的背景下诞生出来。

在此大背景下，于2007年，一家名为柳树车库(Willow Garage)的机器人公司发布了ROS(机器人操作系统)，ROS是一套机器人通用软件框架。可以提升功能模块的复用性，并且随着该系统的不断迭代与完善，如今ROS已经成为机器人领域的事实标准。

概念

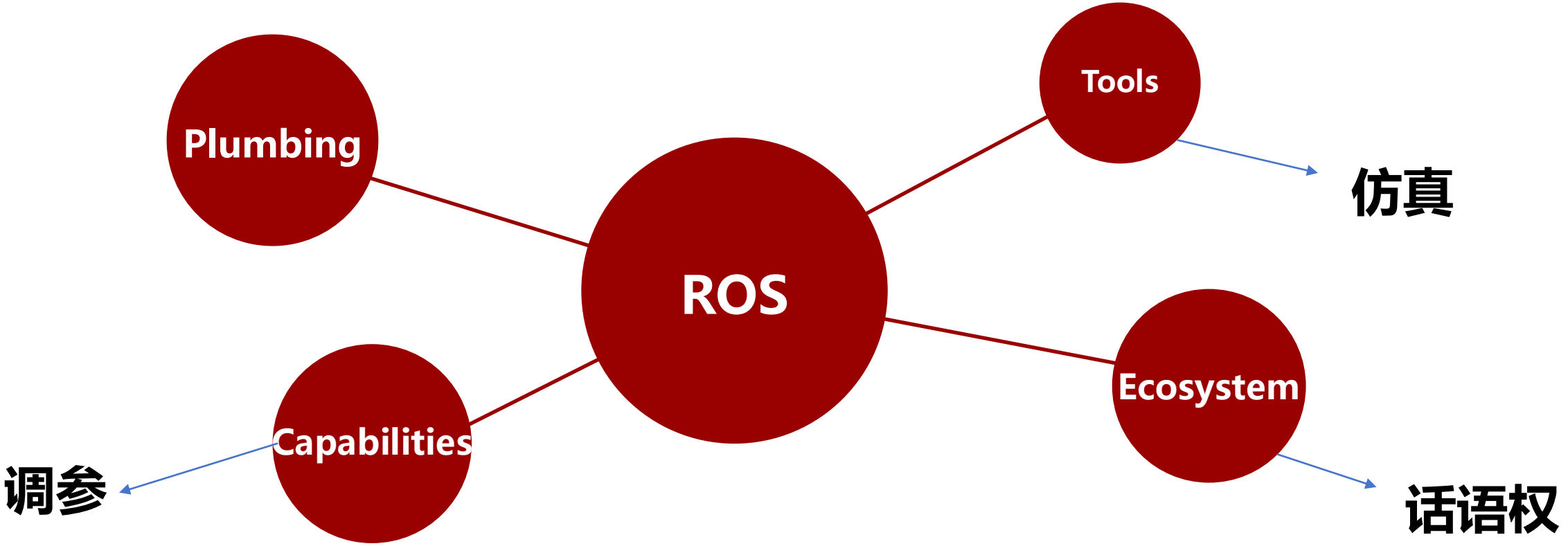
ROS全称Robot Operating System（机器人操作系统）

ROS是适用于机器人的**开源**元操作系统

ROS集成了大量的工具，库，协议，提供类似OS所提供的功能，简化对机器人的控制

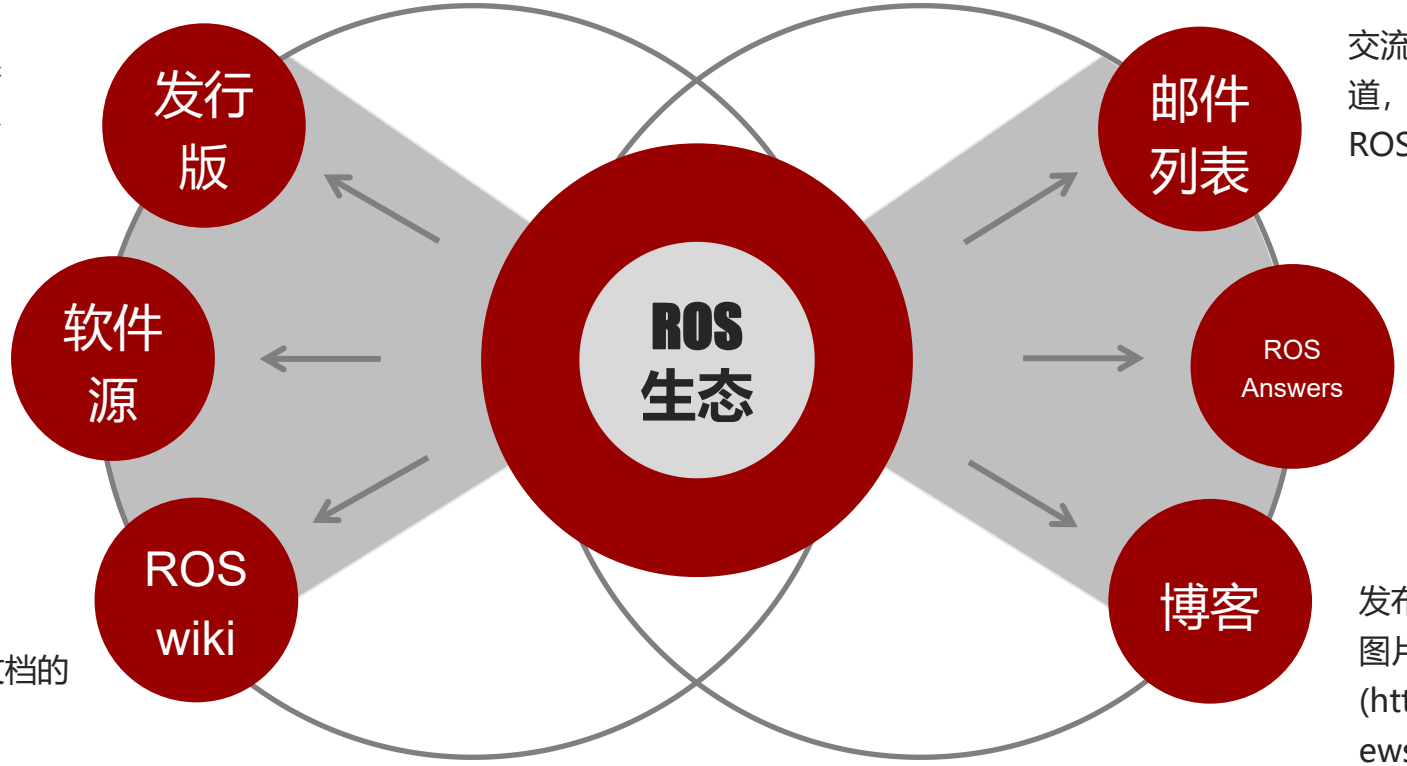
还提供了用于在多台计算机上获取，构建，编写和运行代码的工具和库，ROS在某些方面类似于机器人

概念



ROS设计者将ROS表述为"ROS = Plumbing + Tools + Capabilities + Ecosystem',即ROS是通讯机制，工具软件包，机器人高层技能以及机器人生态系统的集合体

ROS中的生态系统



ROS发行版包括一系列带有版本号、可以直接安装的功能包。

ROS依赖于共享网络上的开源代码,不同的组织机构可以开发或者共享自己的机器人软件。

记录ROS信息文档的主要论坛。

交流ROS更新的主要渠道，同时也可以交流ROS开发的各种疑问。

咨询ROS相关问题的网站。

发布ROS社区中的新闻、图片、视频(http://www.ros.org/news)

设计目标

01

提高代码复用率

在机器人研发领域，通过提供统一的框架，使不同项目之间的代码可以方便地复用，从而提高研发效率。

02

缩短机器人研发周期

通过ROS提供的各种工具和库，机器人研发人员可以更快速地开发和测试新的机器人系统，从而缩短产品的研发周期。

03

灵活可扩展的平台

ROS可以灵活地适应各种不同的机器人硬件和软件需求，通过将不同类型的硬件和软件模块集成到机器人系统中，方便地构建出完整的机器人。

04

简单易用的开发环境

ROS提供了友好的用户界面和丰富的功能，使开发人员可以专注于应用程序的实现而不必关心底层的细节。

发展历程

发展历程

ROS是一个由来已久、贡献者众多的大型软件项目。在ROS诞生之前，很多学者认为，机器人研究需要一个开放式的协作框架，并且已经有不少类似的项目致力于实现这样的框架。在这些工作中，斯坦福大学在2000年年中开展了一系列相关研究项目，如斯坦福人工智能机器人(Standford AI Robot, STAIR)项目、个人机器人(Personal Robots, PR)项目等，在上述项目中，在研究具有代表性、集成式人工智能系统的过程中，创立了用于室内场景的高灵活性、动态软件系统，其可以用于机器人学研究。

2007年，柳树车库(Willow Garage)提供了大量资源，用于将斯坦福大学机器人项目中的软件系统进行扩展与完善，同时，在无数研究人员的共同努力下，ROS的核心思想和基本软件包逐渐得到完善。

ROS的发行版本(ROS distribution) 指ROS软件包的版本，其与Linux的发行版本(如Ubuntu)的概念类似。推出ROS发行版本的目的在于使开发人员可以使用相对稳定的代码库，直到其准备好将所有内容进行版本升级为止。因此，每个发行版本推出后，ROS开发者通常仅对这一版本的bug进行修复， 同时提供少量针对核心软件包的改进。

发展历程

2007
诞生于斯坦福
STAIR项目
Morgan Quigley

2010
ROS 1.0 发布

2012
第一届ROScon

2014
ROS Indigo发布

2
2017
ROS 2.0 Ardent发布

2008
Willow Garage接手

Willow Garage

2011
TurtleBot发布

Open Source Robotics Foundation

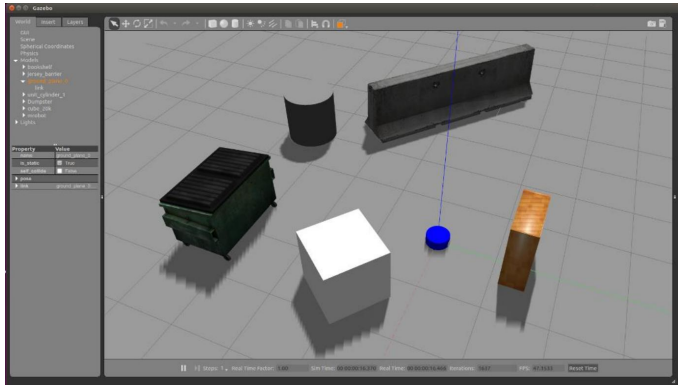
2016
ROS Kinetic发布

2018
ROS Melodic发布

你可以用ROS来干什么

A

A 机器人控制与仿真



B

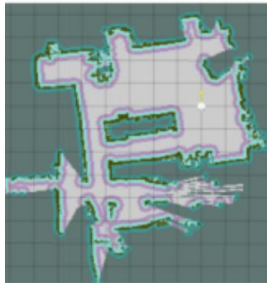
B 即时定位与地图建模

深度信息

IMU信息

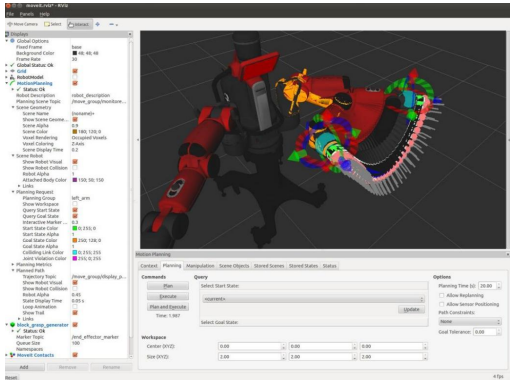
里程计信息

栅格地图



C

C 机械臂运动规划、避障与仿真



ROS中的开发工具

WORKSPACES

Create Workspace

```
mkdir catkin_ws && cd catkin_ws
wstool init src
catkin_make
source devel/setup.bash
```

Add Repo to Workspace

```
roscd; cd ../src
wstool set repo_name \
--git http://github.com/org/repo_name.git \
--version=kinetic-devel
wstool up
```

Resolve Dependencies in Workspace

```
sudo rosdep init  # only once
rosdep update
rosdep install --from-paths src --ignore-src \
--rosdistro=${ROS_DISTRO} -y
```

PACKAGES

Create a Package

```
catkin_create_pkg package_name [dependencies ...]
```

Package Folders

include/package_name	C++ header files
src	Source files. Python libraries in subdirectories
scripts	Python nodes and scripts
msg, srv, action	Message, Service, and Action definitions

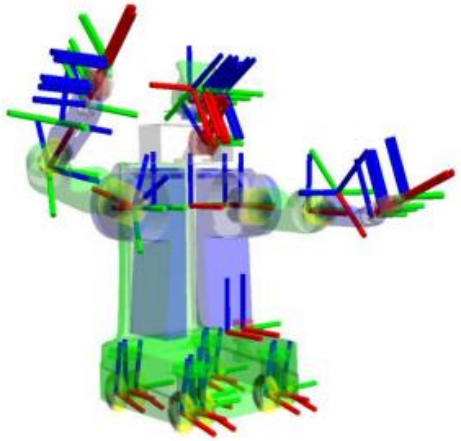
Release Repo Packages

```
catkin_generate_changelog
# review & commit changelogs
catkin_prepare_release
bloom-release --track kinetic --ros-distro kinetic repo_name
```

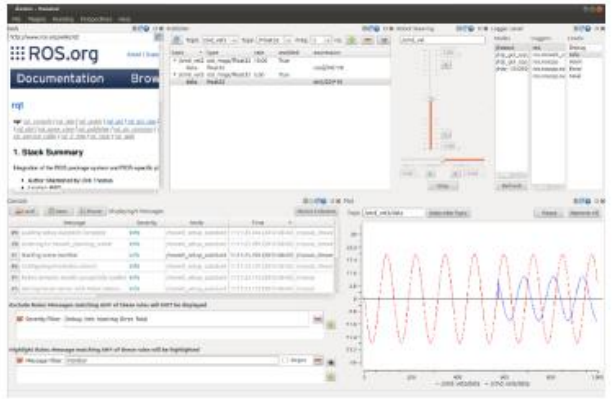
Reminders

- Testable logic
- Publish diagnostics
- Desktop dependencies in a separate package

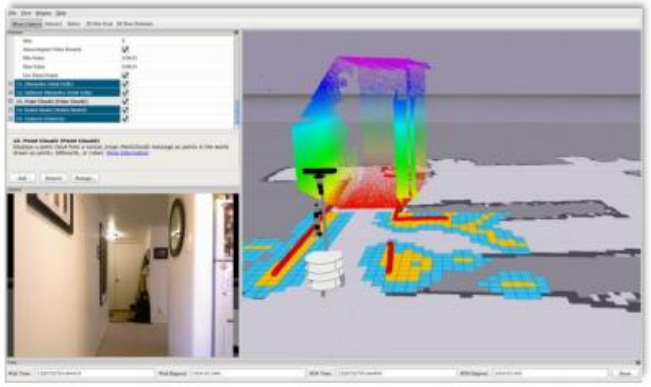
命令行&编译器



TF坐标变换



QT工具箱

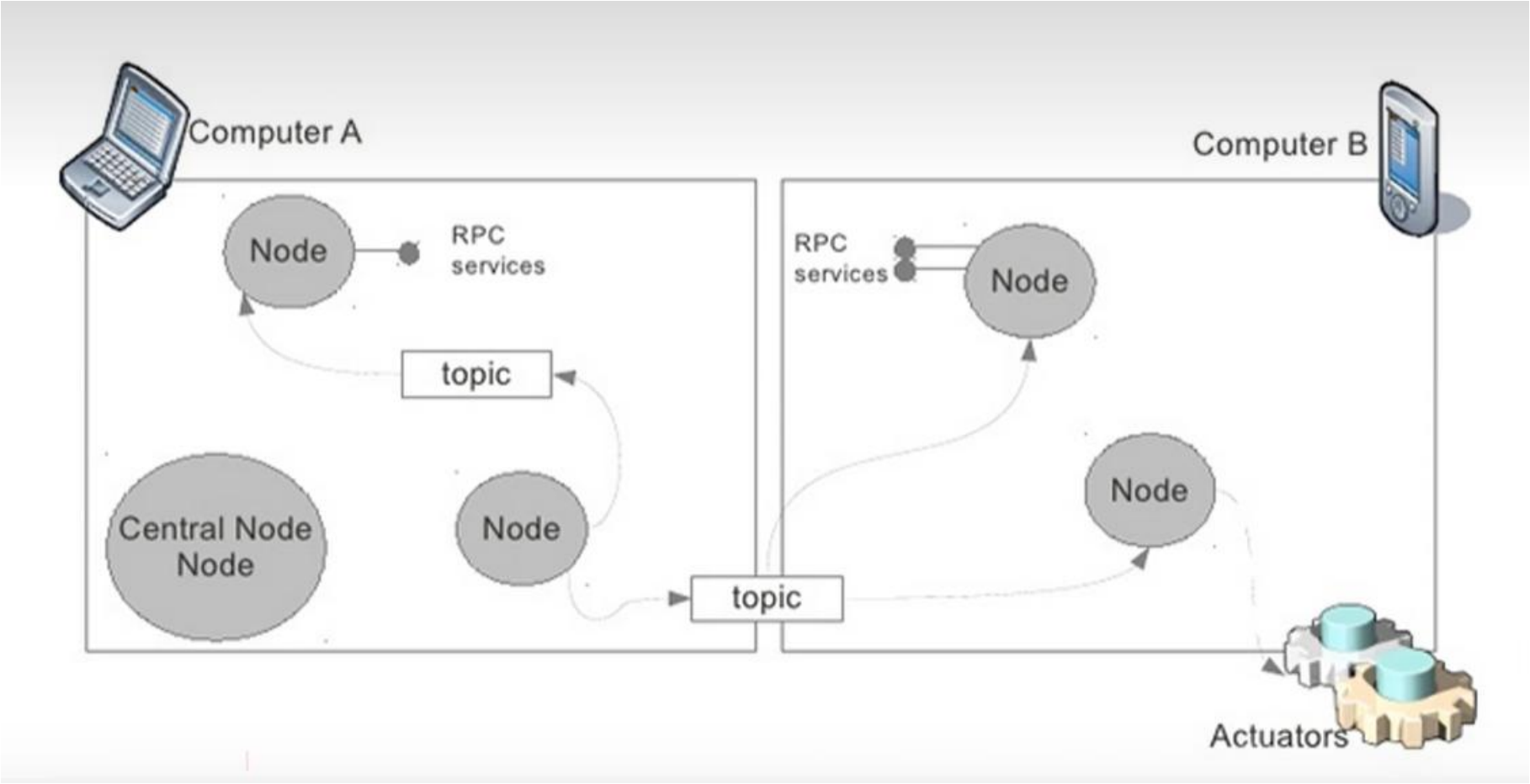


Rviz

Gazebo

通信机制

松耦合式通信机制



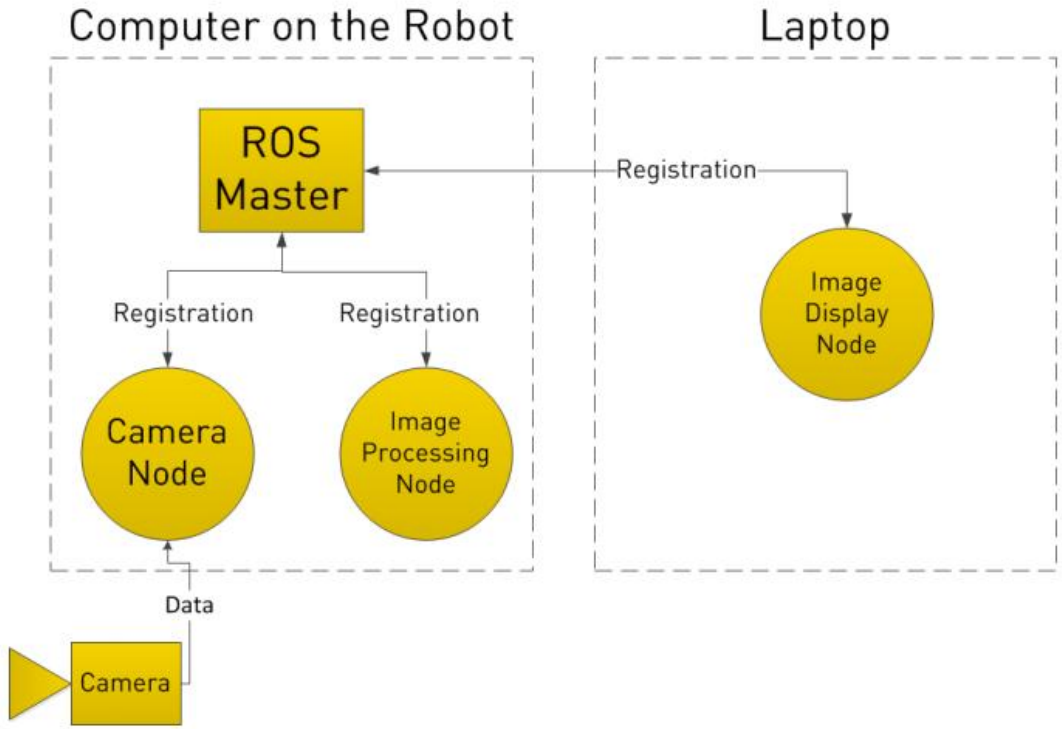
节点与节点管理器

节点(Node) ——执行单元

- 执行具体任务的进程、独立运行的可执行文件;
- 不同节点可使用不同的编程语言，可分布式运行在不同的主机;
- 节点在系统中的名称必须是唯一的。

节点管理器(ROS Master)——控制中心

- 为节点提供命名和注册服务;
- 跟踪和记录话题/服务通信，辅助节点相互查找、建立连接;
- 提供参数服务器， 节点使用此服务器存储和检索



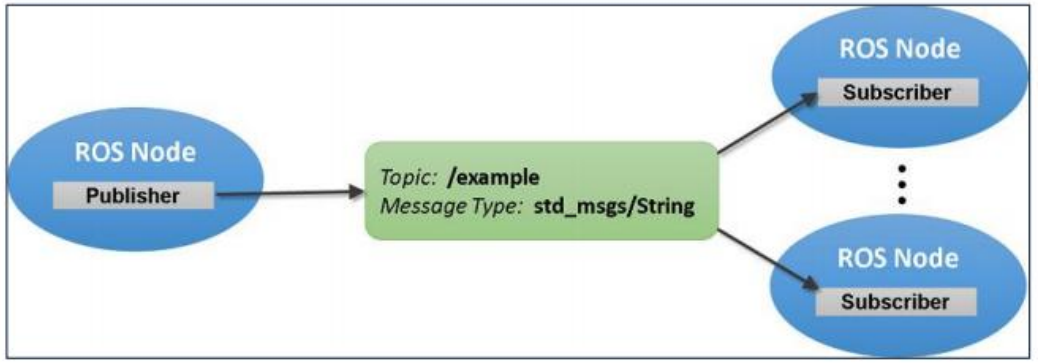
话题通信

话题(Topic) ——异步通信机制

- 节点间用来传输数据的重要总线；
- 使用发布/订阅模型，数据由发布者传输到订阅者，同一个话题的订阅者和发布者可以不唯一。

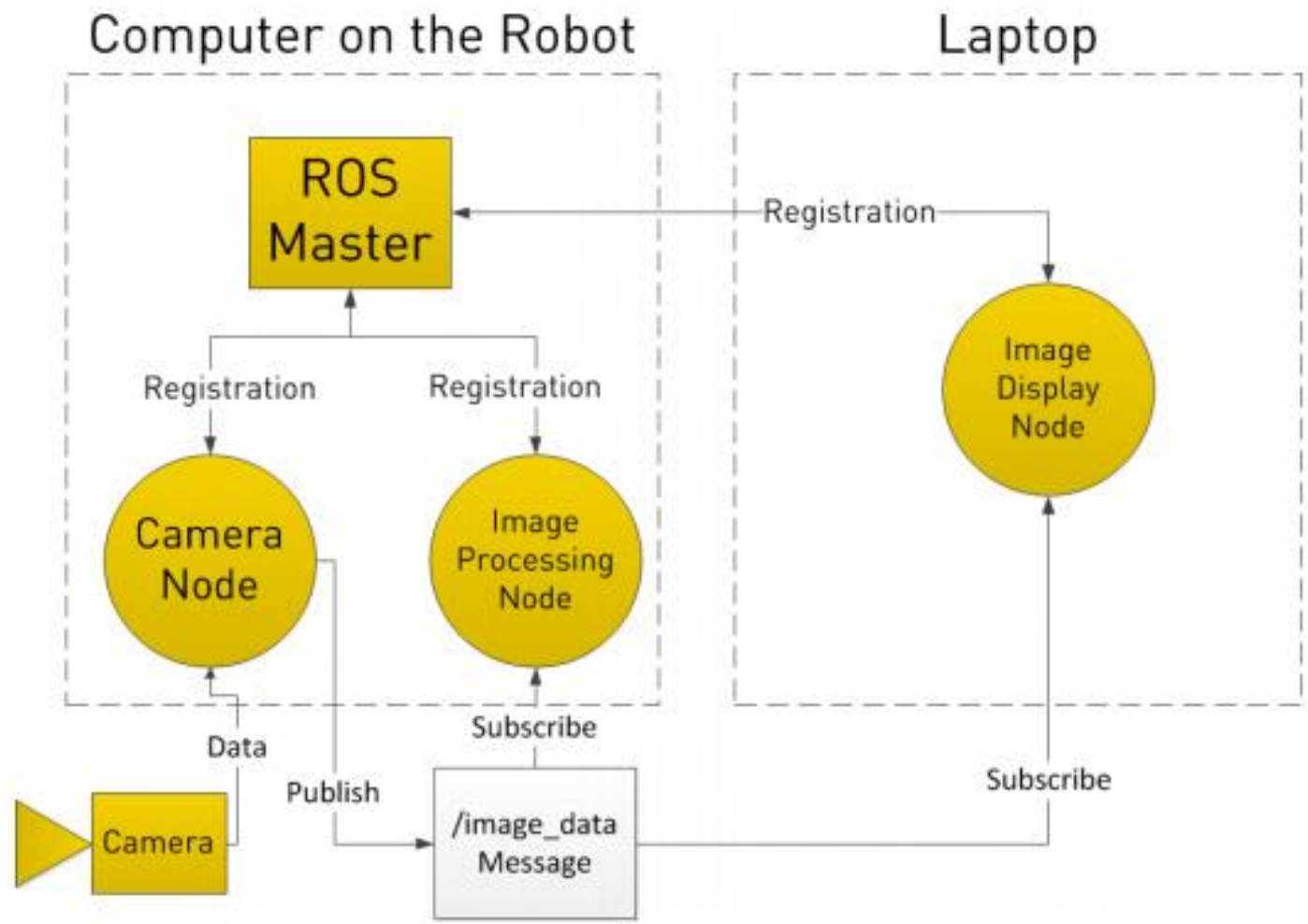
消息(Message)——话题数据

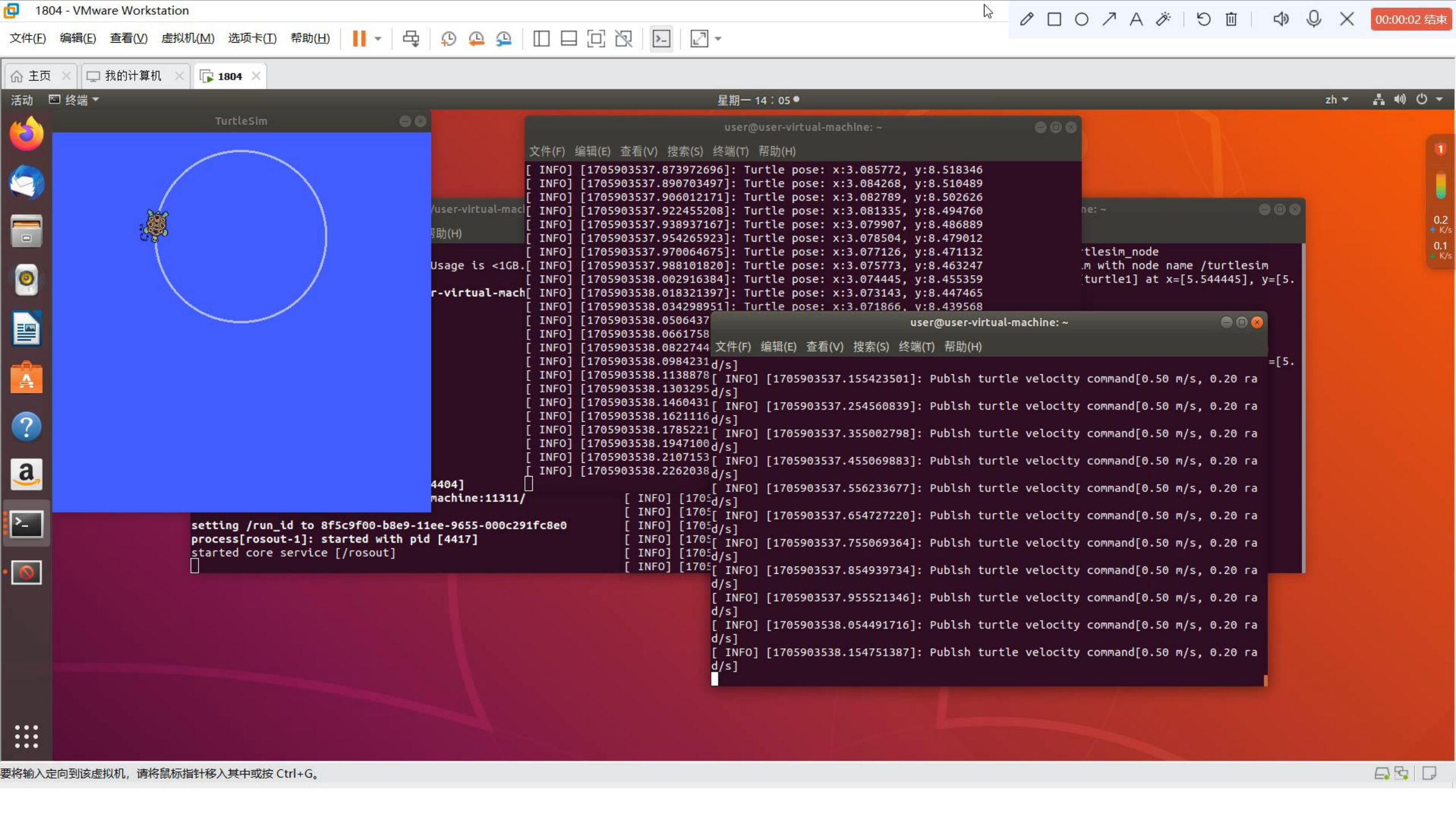
- 具有一定的类型和数据结构，包括ROS提供的标准类型和用户自定义类型；
- 使用编程语言无关的.msg文件定义，编译过程中生成对应的代码文件。

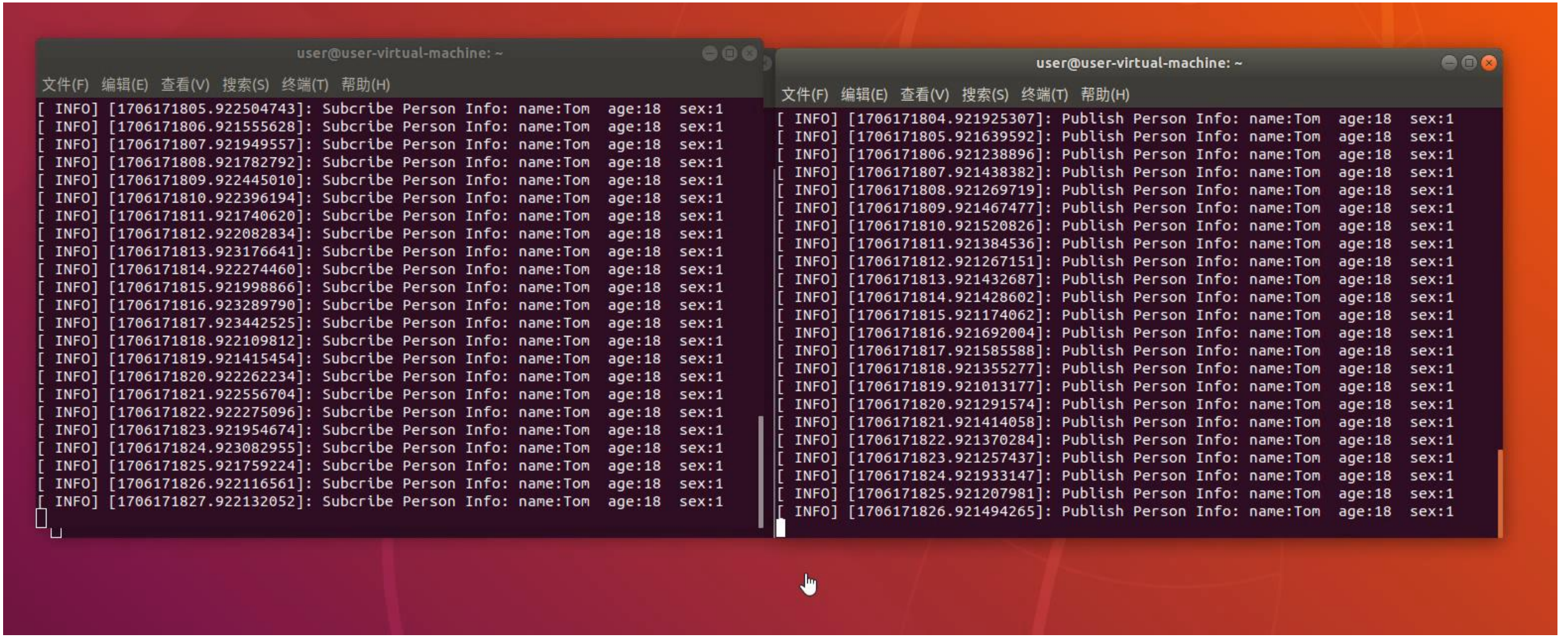


话题模型（发布/订阅）

话题通信


Computer on the Robot
Laptop
ROS Master
Registration
Registration
Registration
Image Display Node
Camera Node
Image Processing Node
Data
Publish
Subscribe
Subscribe
Camera
/image_data Message

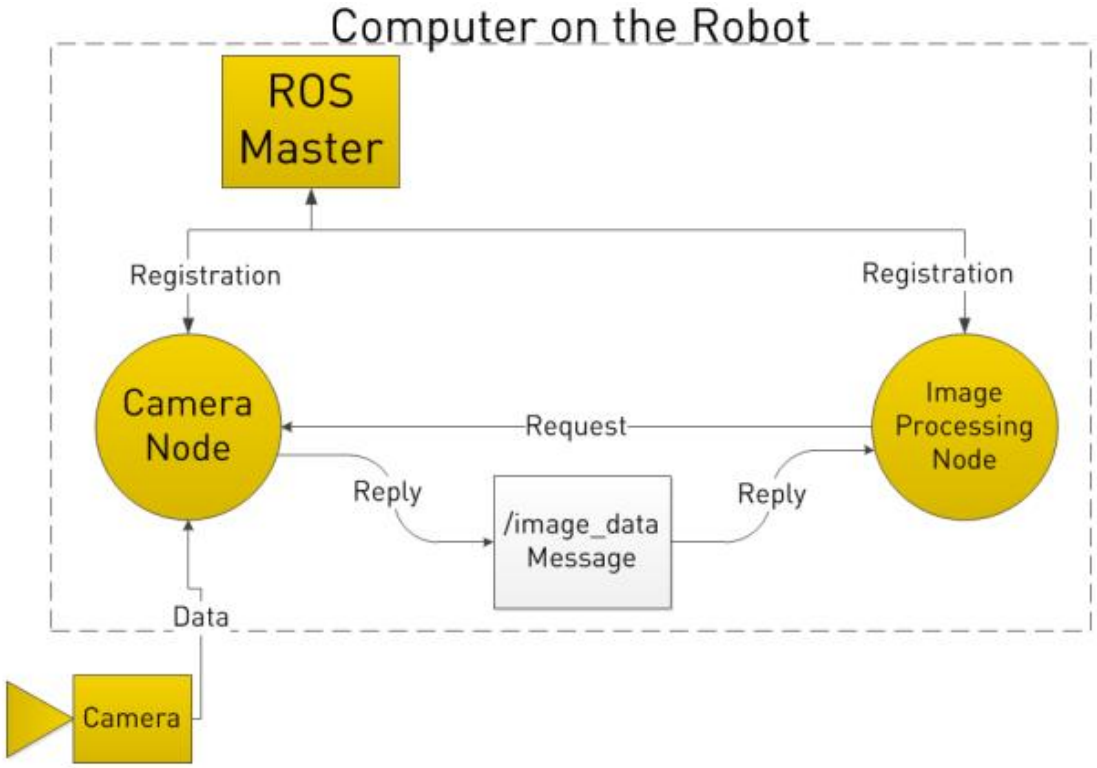

1804 - VMware Workstation
文件(F) 编辑(E) 查看(V) 虚拟机(M) 选项卡(T) 帮助(H)
00:00:02 结束
主页
我的计算机
1804
活动
终端
星期一 14：05
zh
TurtleSim
user@user-virtual-machine: ~
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[INFO] [1705903537.873972696]: Turtle pose: x:3.085772, y:8.518346
[INFO] [1705903537.890703497]: Turtle pose: x:3.084268, y:8.510489
[INFO] [1705903537.906012171]: Turtle pose: x:3.082789, y:8.502626
[INFO] [1705903537.922455208]: Turtle pose: x:3.081335, y:8.494760
[INFO] [1705903537.938937167]: Turtle pose: x:3.079907, y:8.486889
[INFO] [1705903537.954265923]: Turtle pose: x:3.078504, y:8.479012
[INFO] [1705903537.970064675]: Turtle pose: x:3.077126, y:8.471132
[INFO] [1705903537.988101820]: Turtle pose: x:3.075773, y:8.463247
[INFO] [1705903538.002916384]: Turtle pose: x:3.074445, y:8.455359
[INFO] [1705903538.018321397]: Turtle pose: x:3.073143, y:8.447465
[INFO] [1705903538.034298951]: Turtle pose: x:3.071866, y:8.439568
[INFO] [1705903538.0506437
[INFO] [1705903538.0661758
[INFO] [1705903538.0822744
[INFO] [1705903538.0984231
[INFO] [1705903538.1138878
[INFO] [1705903538.1303295
[INFO] [1705903538.1460431
[INFO] [1705903538.1621116
[INFO] [1705903538.1785221
[INFO] [1705903538.1947100
[INFO] [1705903538.2107153
[INFO] [1705903538.2262038
4404]
machine:11311/
setting /run_id to 8f5c9f00-b8e9-11ee-9655-000c291fc8e0
process[rosout-1]: started with pid [4417]
started core service [/rosout]
user@user-virtual-machine: ~
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[INFO] [1705903537.155423501]: Publsh turtle velocity command[0.50 m/s, 0.20 rad/s]
[INFO] [1705903537.254560839]: Publsh turtle velocity command[0.50 m/s, 0.20 rad/s]
[INFO] [1705903537.355002798]: Publsh turtle velocity command[0.50 m/s, 0.20 rad/s]
[INFO] [1705903537.455069883]: Publsh turtle velocity command[0.50 m/s, 0.20 rad/s]
[INFO] [1705903537.556233677]: Publsh turtle velocity command[0.50 m/s, 0.20 rad/s]
[INFO] [1705903537.654727220]: Publsh turtle velocity command[0.50 m/s, 0.20 rad/s]
[INFO] [1705903537.755069364]: Publsh turtle velocity command[0.50 m/s, 0.20 rad/s]
[INFO] [1705903537.854939734]: Publsh turtle velocity command[0.50 m/s, 0.20 rad/s]
[INFO] [1705903537.955521346]: Publsh turtle velocity command[0.50 m/s, 0.20 rad/s]
[INFO] [1705903538.054491716]: Publsh turtle velocity command[0.50 m/s, 0.20 rad/s]
[INFO] [1705903538.154751387]: Publsh turtle velocity command[0.50 m/s, 0.20 rad/s]
要将输入定向到该虚拟机，请将鼠标指针移入其中或按 Ctrl+G。

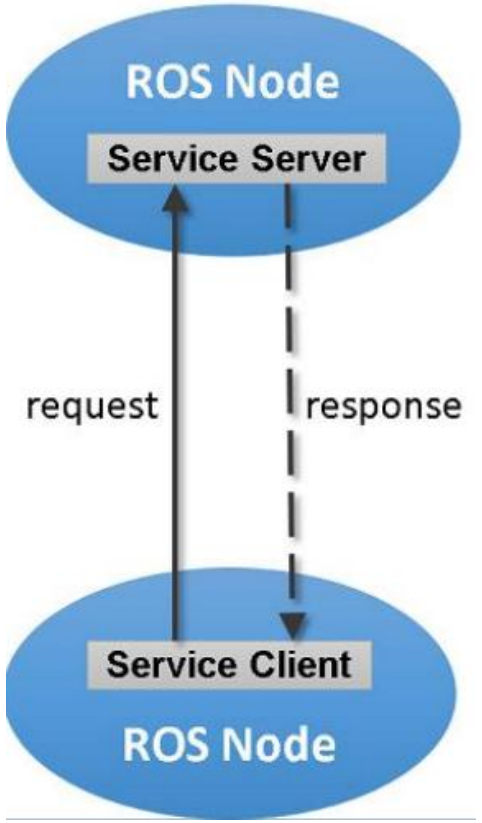

user@user-virtual-machine: ~
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[INFO] [1706171805.922504743]: Subcribe Person Info: name:Tom age:18 sex:1
[INFO] [1706171806.921555628]: Subcribe Person Info: name:Tom age:18 sex:1
[INFO] [1706171807.921949557]: Subcribe Person Info: name:Tom age:18 sex:1
[INFO] [1706171808.921782792]: Subcribe Person Info: name:Tom age:18 sex:1
[INFO] [1706171809.922445010]: Subcribe Person Info: name:Tom age:18 sex:1
[INFO] [1706171810.922396194]: Subcribe Person Info: name:Tom age:18 sex:1
[INFO] [1706171811.921740620]: Subcribe Person Info: name:Tom age:18 sex:1
[INFO] [1706171812.922082834]: Subcribe Person Info: name:Tom age:18 sex:1
[INFO] [1706171813.923176641]: Subcribe Person Info: name:Tom age:18 sex:1
[INFO] [1706171814.922274460]: Subcribe Person Info: name:Tom age:18 sex:1
[INFO] [1706171815.921998866]: Subcribe Person Info: name:Tom age:18 sex:1
[INFO] [1706171816.923289790]: Subcribe Person Info: name:Tom age:18 sex:1
[INFO] [1706171817.923442525]: Subcribe Person Info: name:Tom age:18 sex:1
[INFO] [1706171818.922109812]: Subcribe Person Info: name:Tom age:18 sex:1
[INFO] [1706171819.921415454]: Subcribe Person Info: name:Tom age:18 sex:1
[INFO] [1706171820.922262234]: Subcribe Person Info: name:Tom age:18 sex:1
[INFO] [1706171821.922556704]: Subcribe Person Info: name:Tom age:18 sex:1
[INFO] [1706171822.922275096]: Subcribe Person Info: name:Tom age:18 sex:1
[INFO] [1706171823.921954674]: Subcribe Person Info: name:Tom age:18 sex:1
[INFO] [1706171824.923082955]: Subcribe Person Info: name:Tom age:18 sex:1
[INFO] [1706171825.921759224]: Subcribe Person Info: name:Tom age:18 sex:1
[INFO] [1706171826.922116561]: Subcribe Person Info: name:Tom age:18 sex:1
[INFO] [1706171827.922132052]: Subcribe Person Info: name:Tom age:18 sex:1
user@user-virtual-machine: ~
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[INFO] [1706171804.921925307]: Publish Person Info: name:Tom age:18 sex:1
[INFO] [1706171805.921639592]: Publish Person Info: name:Tom age:18 sex:1
[INFO] [1706171806.921238896]: Publish Person Info: name:Tom age:18 sex:1
[INFO] [1706171807.921438382]: Publish Person Info: name:Tom age:18 sex:1
[INFO] [1706171808.921269719]: Publish Person Info: name:Tom age:18 sex:1
[INFO] [1706171809.921467477]: Publish Person Info: name:Tom age:18 sex:1
[INFO] [1706171810.921520826]: Publish Person Info: name:Tom age:18 sex:1
[INFO] [1706171811.921384536]: Publish Person Info: name:Tom age:18 sex:1
[INFO] [1706171812.921267151]: Publish Person Info: name:Tom age:18 sex:1
[INFO] [1706171813.921432687]: Publish Person Info: name:Tom age:18 sex:1
[INFO] [1706171814.921428602]: Publish Person Info: name:Tom age:18 sex:1
[INFO] [1706171815.921174062]: Publish Person Info: name:Tom age:18 sex:1
[INFO] [1706171816.921692004]: Publish Person Info: name:Tom age:18 sex:1
[INFO] [1706171817.921585588]: Publish Person Info: name:Tom age:18 sex:1
[INFO] [1706171818.921355277]: Publish Person Info: name:Tom age:18 sex:1
[INFO] [1706171819.921013177]: Publish Person Info: name:Tom age:18 sex:1
[INFO] [1706171820.921291574]: Publish Person Info: name:Tom age:18 sex:1
[INFO] [1706171821.921414058]: Publish Person Info: name:Tom age:18 sex:1
[INFO] [1706171822.921370284]: Publish Person Info: name:Tom age:18 sex:1
[INFO] [1706171823.921257437]: Publish Person Info: name:Tom age:18 sex:1
[INFO] [1706171824.921933147]: Publish Person Info: name:Tom age:18 sex:1
[INFO] [1706171825.921207981]: Publish Person Info: name:Tom age:18 sex:1
[INFO] [1706171826.921494265]: Publish Person Info: name:Tom age:18 sex:1

服务通信

服务(Service) ——同步通信机制

- 使用客户端/服务器(C/S) 模型，客户端发送请求数据，服务器完成处理后返回应答数据;
- 使用编程语言无关的.srv文件定义请求和应答数据结构，编译过程中生成对应的代码文件。





服务模型（请求/应答）

话题与服务的区别

	话题	服务
同步性	异步	同步
通信模型	发布/订阅	服务器/客户端
底层协议	ROSTCP/ROSUDP	ROSTCP/ROSUDP
反馈机制	无	有
缓冲区	有	无
实时性	弱	强
节点关系	多对多	一对多（一个server）
适用场景	数据传输	逻辑处理

文件系统

元功能包(Meta Packages)

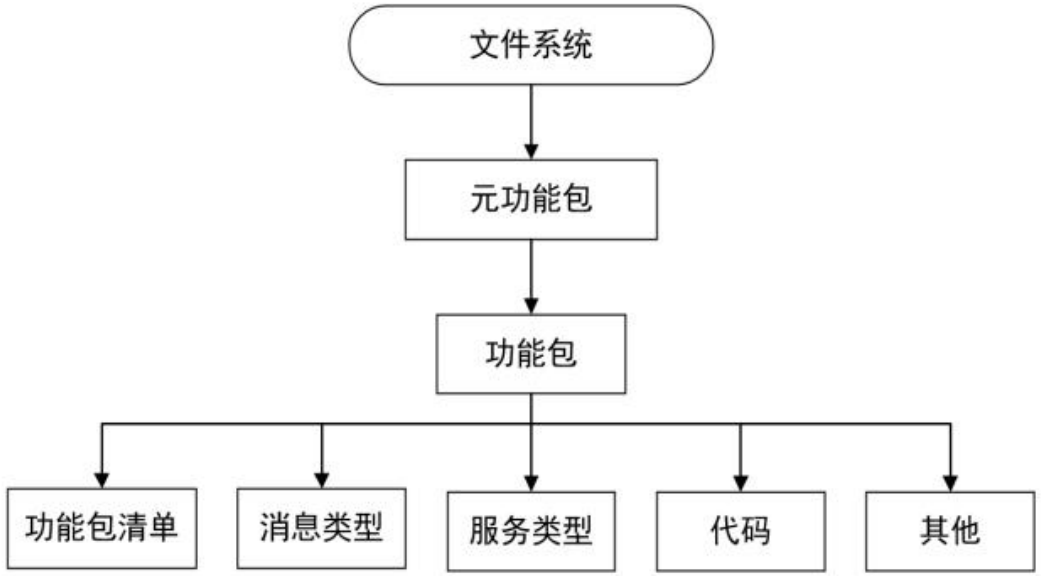
- 组织多个用于同一目的功能包

功能包（Package）

- ROS软件中的基本单元，包含节点源码、配置文件、数据定义等

功能包清单(Package manifest)

- 记录功能包的基本信息，包含作者信息、许可信息、依赖选项、编译标志等



ROS命令行工具

常用命令：

- rostopic
- rosservice
- rosnode
- rosparam
- rosmsg
- rossrv

WORKSPACES

Create Workspace

```
mkdir catkin_ws && cd catkin_ws
wstool init src
catkin_make
source devel/setup.bash
```

Add Repo to Workspace

```
roscd; cd ../src
wstool set repo_name \
--git http://github.com/org/repo_name.git \
--version=kinetic-devel
wstool up
```

Resolve Dependencies in Workspace

```
sudo rosdep init  # only once
rosdep update
rosdep install --from-paths src --ignore-src \
--rosdistro=${ROS_DISTRO} -y
```

PACKAGES

Create a Package

```
catkin_create_pkg package_name [dependencies ...]
```

Package Folders

include/package_name	C++ header files
src	Source files. Python libraries in subdirectories
scripts	Python nodes and scripts
msg, srv, action	Message, Service, and Action definitions

Release Repo Packages

```
catkin_generate_changelog
# review & commit changelogs
catkin_prepare_release
bloom-release --track kinetic --ros-distro kinetic repo_name
```

Reminders

- Testable logic
- Publish diagnostics
- Desktop dependencies in a separate package

CMakeLists.txt

Skeleton

```
cmake_minimum_required(VERSION 2.8.3)
project(package_name)
find_package(catkin REQUIRED)
catkin_package()
```

Package Dependencies

To use headers or libraries in a package, or to use a package's exported CMake macros, express a build-time dependency:

```
find_package(catkin REQUIRED COMPONENTS roscpp)
```

Tell dependent packages what headers or libraries to pull in when your package is declared as a catkin component:

```
catkin_package(
  INCLUDE_DIRS include
  LIBRARIES ${PROJECT_NAME}
  CATKIN_DEPENDS roscpp)
```

Note that any packages listed as CATKIN_DEPENDS dependencies must also be declared as a <run_depend> in package.xml.

Messages, Services

These go after find_package(), but before catkin_package().
Example:

```
find_package(catkin REQUIRED COMPONENTS message_generation
std_msgs)
add_message_files(FILES MyMessage.msg)
add_service_files(FILES MyService.msg)
generate_messages(DEPENDENCIES std_msgs)
catkin_package(CATKIN_DEPENDS message_runtime std_msgs)ww
```

Build Libraries, Executables

Goes after the catkin_package() call.

```
add_library(${PROJECT_NAME} src/main)
add_executable(${PROJECT_NAME}_node src/main)
target_link_libraries(
  ${PROJECT_NAME}_node ${catkin_LIBRARIES})
```

Installation

```
install(TARGETS ${PROJECT_NAME}
  DESTINATION ${CATKIN_PACKAGE_LIB_DESTINATION})
install(TARGETS ${PROJECT_NAME}_node
  DESTINATION ${CATKIN_PACKAGE_BIN_DESTINATION})
install(PROGRAMS scripts/myscript
  DESTINATION ${CATKIN_PACKAGE_BIN_DESTINATION})
install(DIRECTORY launch
  DESTINATION ${CATKIN_PACKAGE_SHARE_DESTINATION})
```

RUNNING SYSTEM

Run ROS using plain:

```
roscore
```

Alternatively, roslaunch will run its own roscore automatically if it can't find one:

```
roslaunch my_package package_launchfile.launch
```

Suppress this behaviour with the `--wait` flag.

Nodes, Topics, Messages

```
rosnode list
rostopic list
rostopic echo cmd_vel
rostopic hz cmd_vel
rostopic info cmd_vel
rosmsg show geometry_msgs/Twist
```

Remote Connection

Master's ROS environment:

- ROS_IP or ROS_HOSTNAME set to this machine's network address.
- ROS_MASTER_URI set to URI containing that IP or hostname.

Your environment:

- ROS_IP or ROS_HOSTNAME set to your machine's network address.
- ROS_MASTER_URI set to the URI from the master.

To debug, check ping from each side to the other, run roswtf on each side.

ROS Console

Adjust using rqt_logger_level and monitor via rqt_console. To enable debug output across sessions, edit the `$HOME/.ros/config/rosconsole.config` and add a line for your package:

```
log4j.logger.ros.package_name=DEBUG
```

And then add the following to your session:

```
export ROSCONSOLE_CONFIG_FILE=$HOME/.ros/config/rosconsole.config
```

Use the roslaunch --screen flag to force all node output to the screen, as if each declared <node> had the `output="screen"` attribute.

以海龟仿真器为例

启动ROS Master $roscore

↓

启动海龟仿真器 $rosrun turtlesim turtlesim_node

↓

启动海龟控制节点 $rosrun turtlesim turtle_teleop_key

海龟仿真器页面

```
user@user-virtual-machine:~$ rosrun turtlesim turtlesim_node
[ INFO] [1706010966.684341520]: Starting turtlesim with node name /turtlesim
[ INFO] [1706010966.691353205]: Spawning turtle [turtle1] at x=[5.544445], y=[5.
544445], theta=[0.000000]
```

启动海龟仿真器节点

```
user@user-virtual-machine:~$ rosrun turtlesim turtle_teleop_key
Reading from keyboard
---------------------------
Use arrow keys to move the turtle. 'q' to quit.
```

启动海龟控制节点

以海龟仿真器为例

查看话题列表

```
$rosnode list
```

发布话题消息

```
$rostopic pub -r 10 /turtle1/cmd_vel geometry_msgs/Twist "linear:
x: 1.0
y: 0.0
Z: 0.0
angular:
X: 0.0
y:0.0
z: 0.0"
```

发布服务请求

```
$rosservice call /spawn "x: 5.0
y: 5.0
theta: 0.0
name: 'turtle2"
```

以海龟仿真器为例

话题记录

```
$rosbag record –a –O cmd_record
```

话题复现

```
$rosbag play cmd_record.bag
```