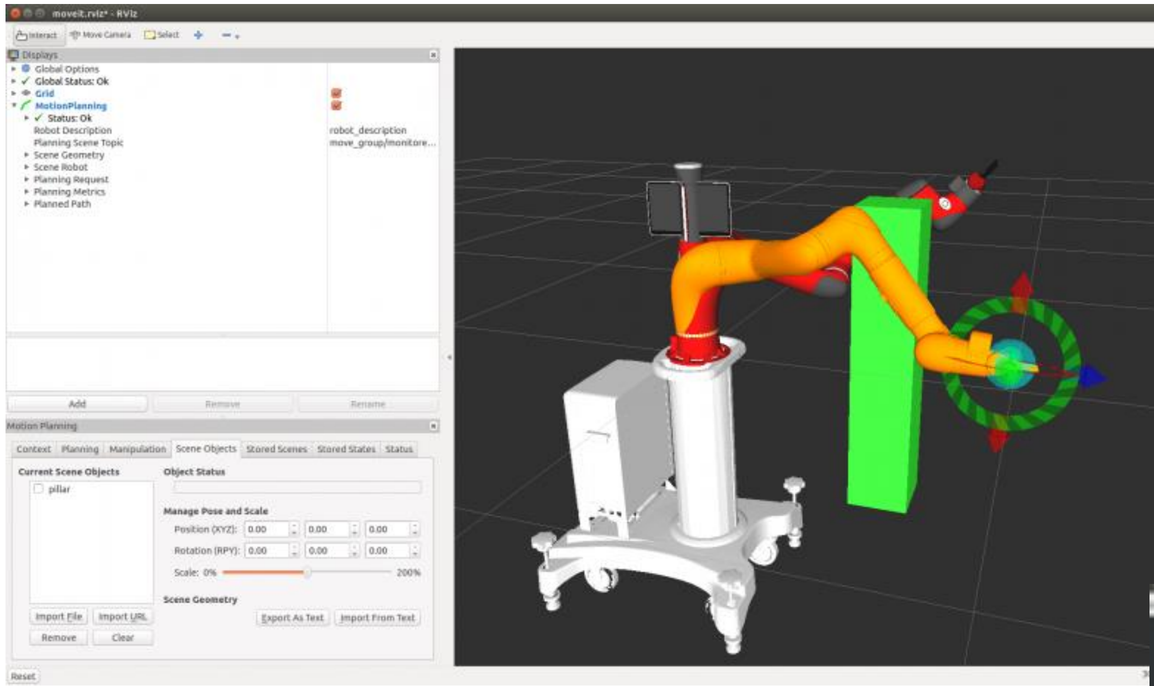
Introduction to ROS: Basic Concepts

Dr. Armin Biess & Mr. Shai Givati

Goal

- Learn to use ROS: the Robot Operating System
- Learn to use Gazebo: the ROS 3D simulator
- Apply this knowledge to do some simple programming on a real robot

ROS



- Global Options
- Global Status: Ok
- Grid ☑
- MotionPlanning ☑
  - Status: Ok
  - Robot Description: robot_description
  - Planning Scene Topic: move_group/monitore...
  - Scene Geometry
  - Scene Robot
  - Planning Request
  - Planning Metrics
  - Planned Path

Current Scene Objects

□ pillar

Object Status

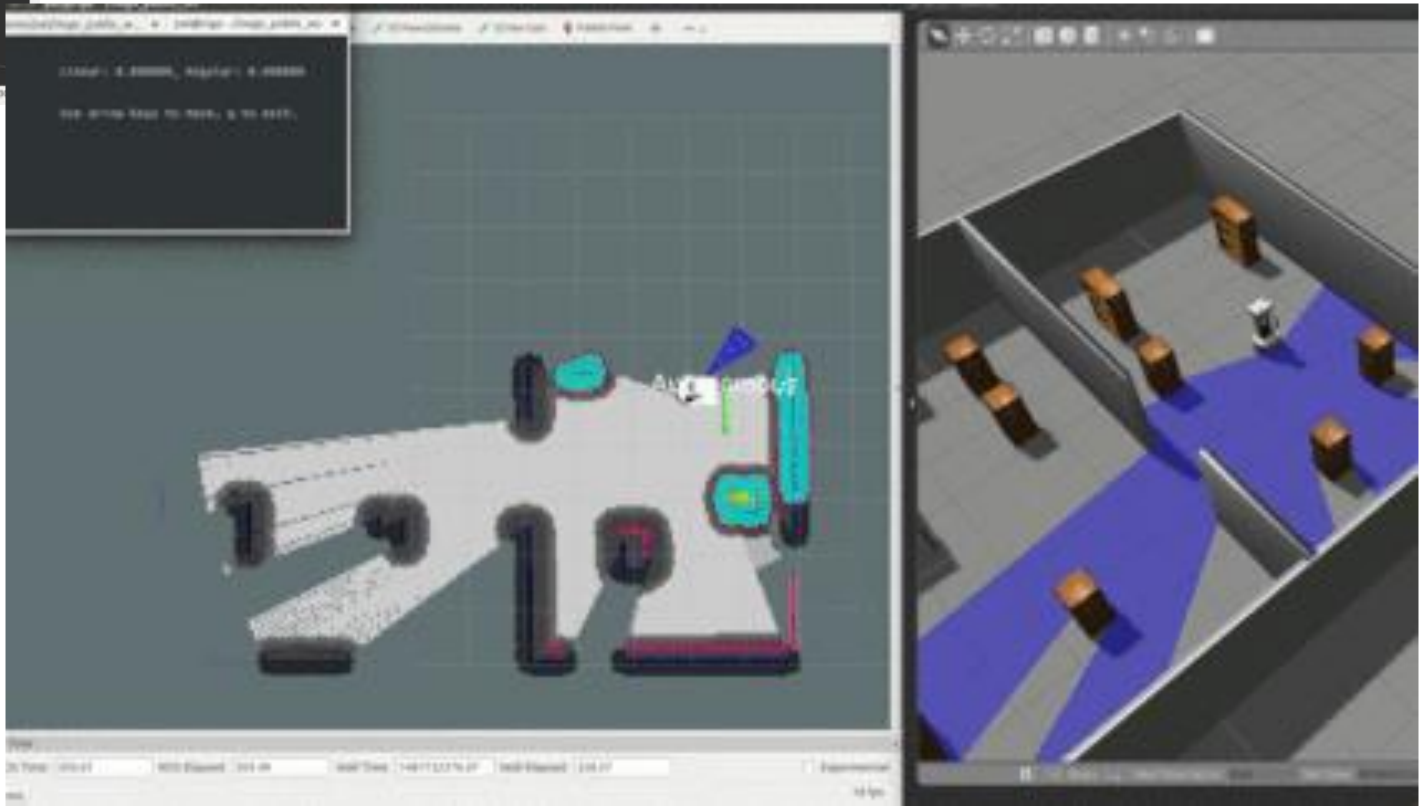
Manage Pose and Scale

Position (XYZ): 0.00 0.00 0.00

Rotation (RPY): 0.00 0.00 0.00

Scale: 0% 200%

Scene Geometry



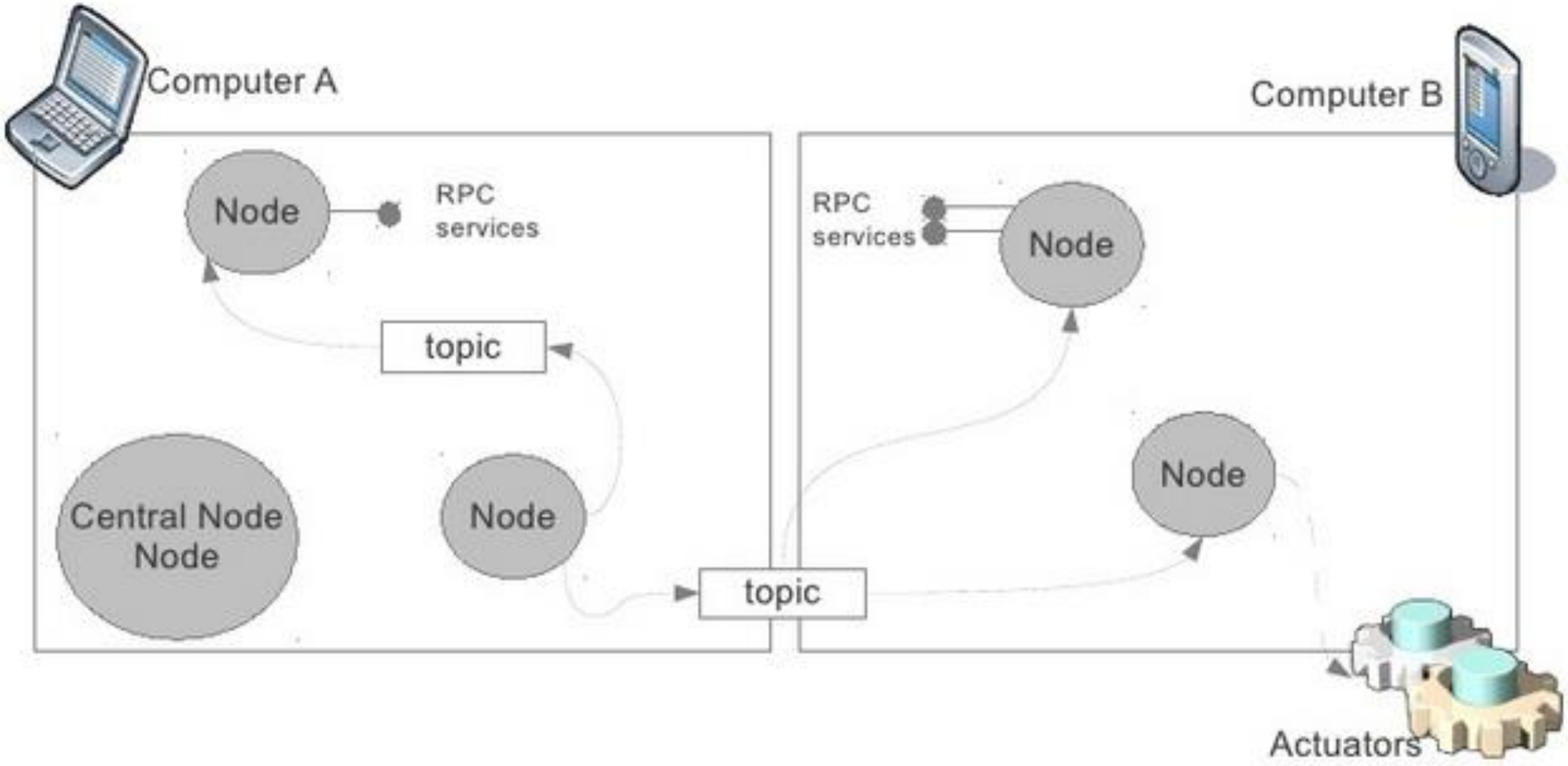
```
Linear: 0.000000, Angular: 0.000000

Use arrow keys to move, q to exit.
```

ROS

- An open source, operating system for robots
- Provides following services:
    - hardware abstraction
    - low-level device control
    - implementation of commonly used functionality
    - message passing between processes
    - package management
    - Tools and libraries for obtaining, building, writing, and running code across multiple computers
- ROS provides many packages for diverse robotic tasks, starting with manipulation and navigation, to mapping environments and doing automated planning

ROS Distributed Architecture



Run-time

- A peer-to-peer network of processes (potentially distributed over multiple machines) that are loosely coupled and use the ROS communication infrastructure
  - Synchronous communication over services
  - Asynchronous communication over topics

ROS Core Concepts

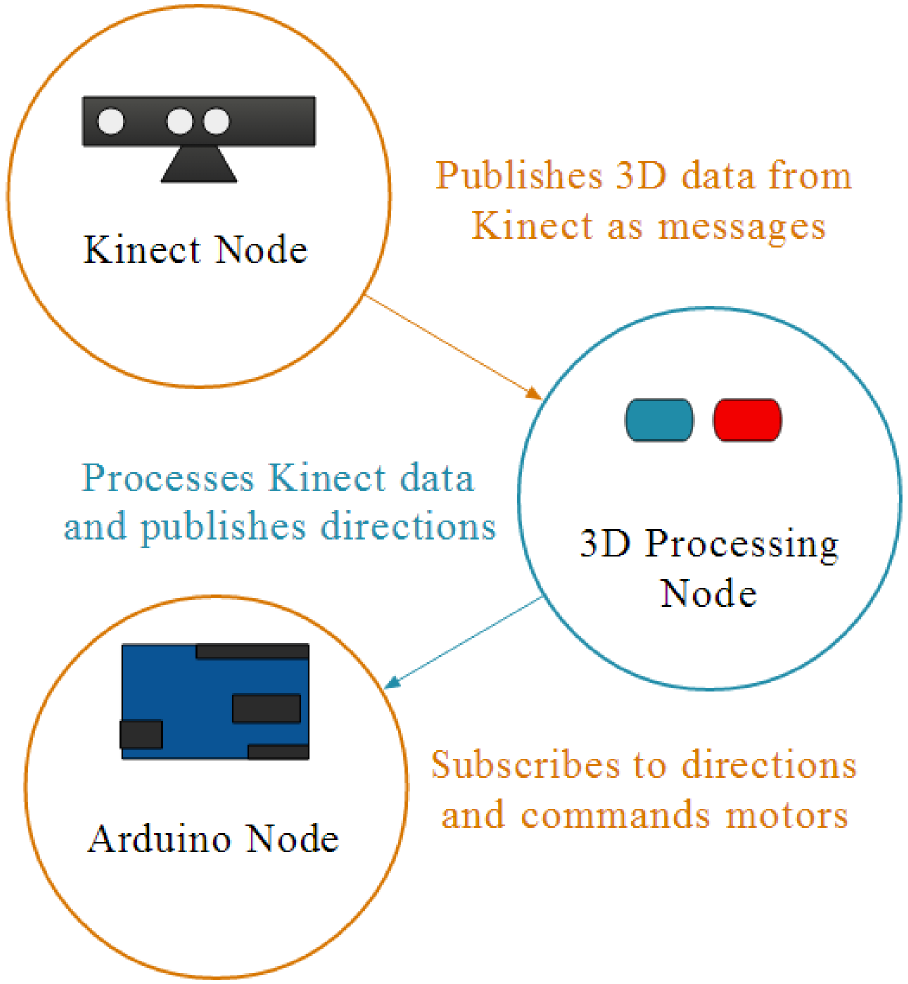
- Nodes
- Messages and Topics
- Services
- ROS Master
- Parameters
- Stacks and packages

ROS Nodes

- Single-purposed executable programs
  - e.g. sensor driver(s), actuator driver(s), mapper, planner, UI, etc.
- Modular design
  - Individually compiled, executed, and managed
- Nodes are written using a ROS **client library**
  - roscpp – C++ client library
  - rospy – python client library
- Nodes can publish or subscribe to a Topic
- Nodes can also provide or use a Service

ROS Topics

- Nodes communicate with each other by publishing messages to topics
- Publish/Subscribe model: 1-to-N broadcasting



Kinect Node
Publishes 3D data from Kinect as messages
Processes Kinect data and publishes directions
3D Processing Node
Arduino Node
Subscribes to directions and commands motors

Topics

- Topics: Messages are routed via publish/subscribe semantics
  - A node sends a message by publishing to a topic
  - The topic is a name that is used to identify the content of a message
  - A node interested in certain messages will subscribe to this topic
  - Multiple nodes can publish/subscribe to the same topic
  - Publishers/subscribers are unaware of each other
  - A form of asynchronous communication
  - Example: a sensor node publishes its reading to a topic. Other nodes can process it. They can publish the processed data to a different topic. Controller nodes can use that to decide how to control the motors

ROS Messages

- Strictly-typed data structures for inter-node communication
- For example, geometry_msgs/Twist is used to express velocity broken into linear and angular parts:

```
Vector3 linear
Vector3 angular
```

- Vector3 is another message type composed of:

```
float64 x
float64 y
float64 z
```

Demo

ROS Services

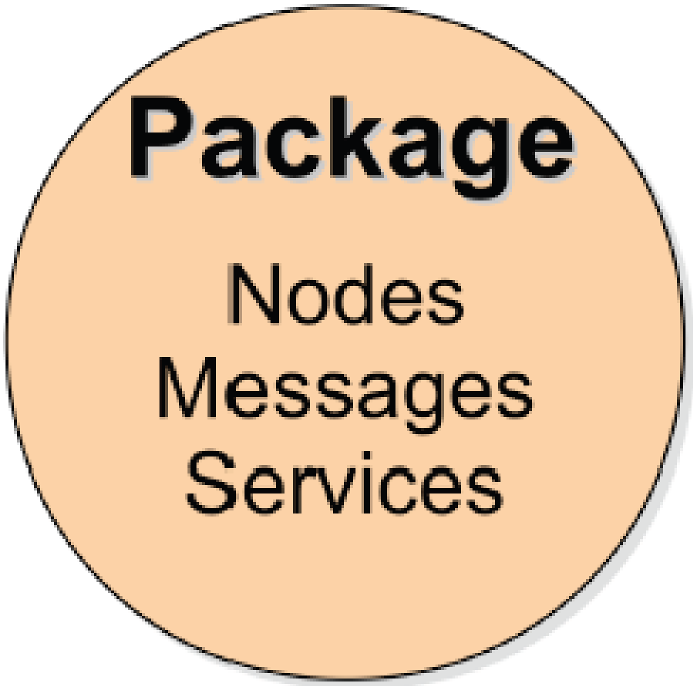
- Synchronous inter-node transactions / RPC
- Service/Client model: 1-to-1 request-response
- Service roles:
  - carry out remote computation
  - trigger functionality / behavior
- Example:
  - map_server/static_map – retrieves the current grid map used by the robot for navigation

File System Support

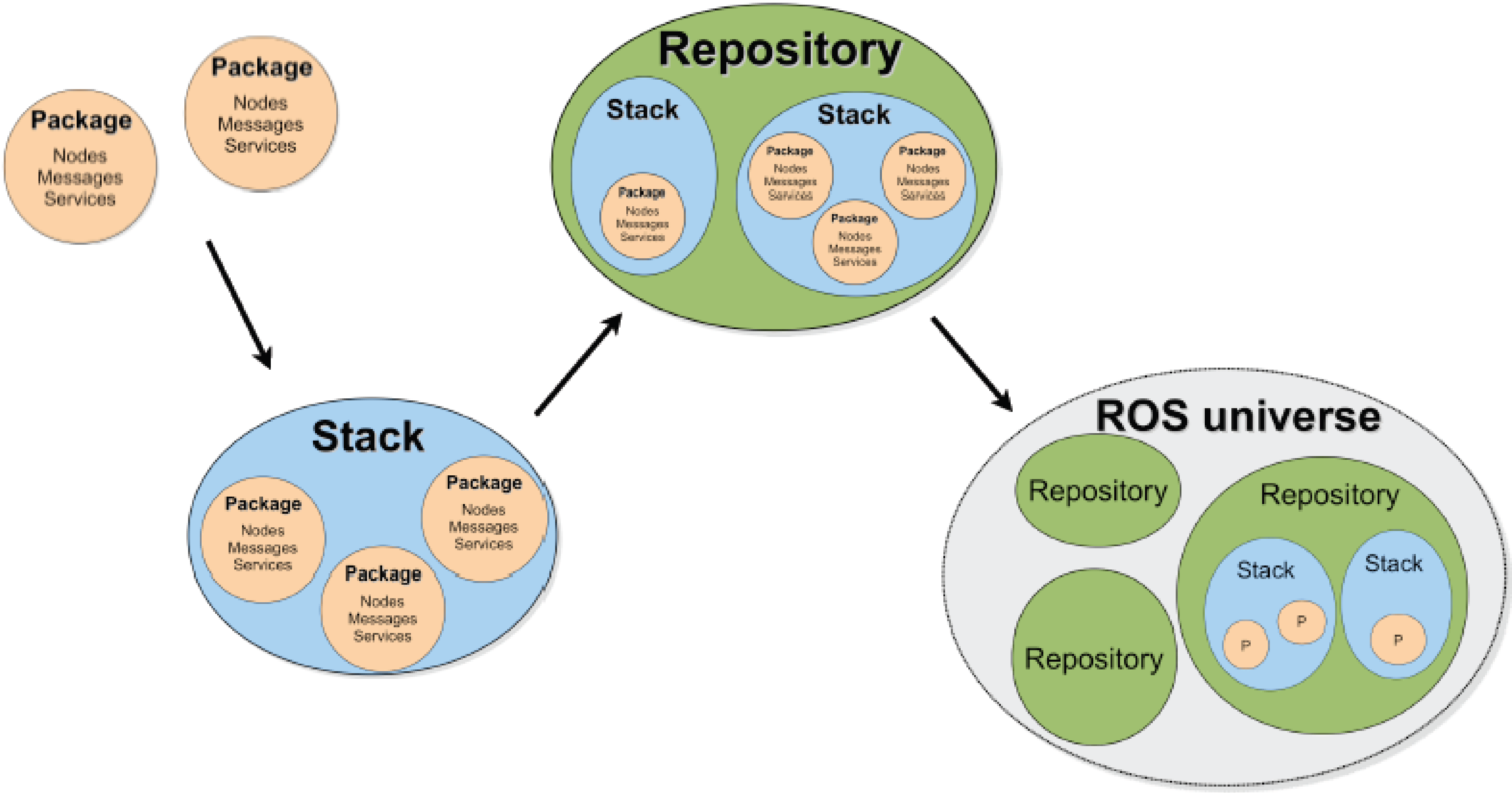
- Packages: the main unit for organizing software in ROS. Contains runtime processes (nodes), datasets, configuration, etc.
- This is the most granular thing you can build and release
- Message types: message descriptions that define the data structures for messages sent in ROS
- Service types: service description that define the request and response data structure for services

ROS Packages

- Software in ROS is organized in *packages*.
- A package contains one or more nodes and provides a ROS interface
- Most of ROS packages are hosted in GitHub



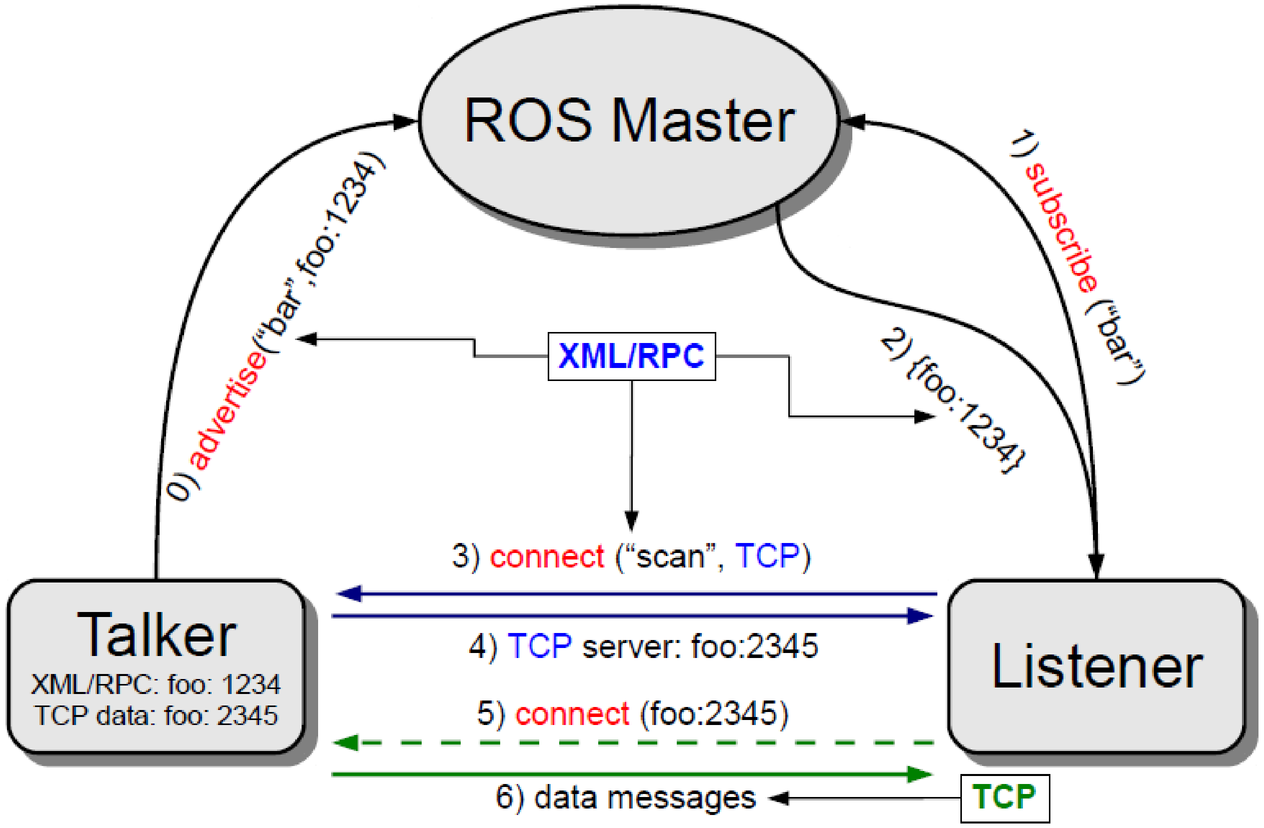
ROS Package System



Taken from Sachin Chitta and Radu Rusu (Willow Garage)

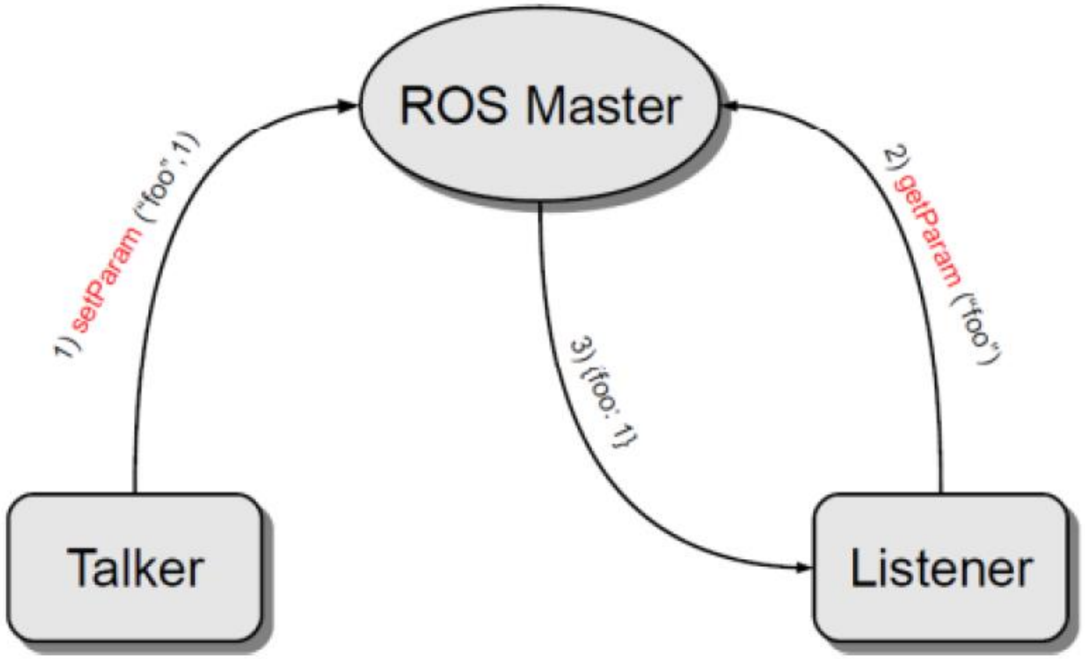
ROS Master

- Enable ROS nodes to locate one another
- Think of it as a ROS directory service, sort of DNS
  - Provides naming & registration services for nodes, topics, services, etc



Parameter Server

- A shared, multi-variate dictionary that is accessible via network APIs.
- Best used for static, non-binary data such as configuration parameters.
- Runs inside the ROS master



Various Supplied Capabilities

- Coordinate transforms — useful for geometric reasoning
- ActionLib — an interface for interacting with preemptable actions (such as move to a location, perform scan)
    - This is like a service, but one that may take a long time, and requires periodic feedback about progress and the ability to stop the service
    - One can specify goals, feedback, and result
- Different classes of messages (actions, diagnostics, etc.)
- Plugin support — enables loading/unloading plugins dynamically without the application being aware of these earlier.
- Filters — various filters for data processing
- Robot models

Easy Integration with Popular Open Source Projects

- Gazebo — a 3D robot simulator. A model of our Komodo robot already exists

- OpenCV — a large machine vision library

- PointCloudLibrary — library for manipulation and processing 3d data and depth image. For example, the Kinect we have returns this type of data

- MoveIt — a motion planning library

Introduction to linux

- A family of free and open-source software operating systems built around the Linux kernel

- Ubuntu – a Linux distribution

- See more here - http://aeswiki.datasys.swri.edu/rositraining/indigo/Exercises/

What to Expect

- ROS requires working in Linux using C++ and Python
  - More advanced work on the robot can be done with JAVA — but not here
  - Programming a robot is hard, but rewarding. Unlike software in the virtual world, it is influenced by the real world, and does not always have the expected results
- A lot of self study

  You will learn all the material from online tutorials, other resources, and from experiencing things on your own
- However, we provide help in the form of office.
- Cooperation among groups in learning the material is encouraged

Work Plan

- [Stage 1] Install
  - Install ROS Kinetic on your laptop. Requires Ubuntu 16.04 (recommended) or another linux distribution. See the ROS installation instructions. You will find installation instructions at http://wiki.ros.org/ROS/Installation

- [Stage 2] Basics
  - Run **all** (about 20) the beginner tutorials (you can choose either C++ or Python, where relevant) + Assignment 1
  - http://wiki.ros.org/ROS/Tutorials
  - Read the ROS Introduction: http://wiki.ros.org/ROS/Introduction

- [Stage 3] Gazebo
  - Class tutorial on Gazebo. Install the turtlebot3 package and run the simulation tutorial http://emanual.robotis.com/docs/en/platform/turtlebot3/simulation/#simulation
  - Assigment 2 – implement basic motions in Gazebo

- [Stage 4] Real Robot
  - Class tutorial on real robot
  - Assigment 3 - Implement basic motions on robot

Sources

- Much material is available online.
  - ROS Wiki: http://wiki.ros.org/ROS/Introduction
  - Installation: http://wiki.ros.org/ROS/Installation
  - Tutorials: http://wiki.ros.org/ROS/Tutorials
  - Book: **Programming Robots with ROS** by Morgan Quigley, Brian Gerkey, and William D. Smar - O'Reilly books.
    - http://file.allitebooks.com/20151124/Programming%20Robots%20with%20ROS.pdf
  - ROS Tutorial Videos http://www.youtube.com/playlist?list=PLDC89965A56E6A8D6
  - ROS Cheat Sheet http://www.tedusar.eu/files/summerschool2013/ROScheatsheet.pdf
  - Very good course slides from Bar-Ilan by Roi Yehoshua including basic of installation, code examples, etc. http://u.cs.biu.ac.il/~yehoshr1/89-685/
  - www.theconstructsim.com (if it works)
  - Many other tutorials, videos, etc.
  - http://aeswiki.datasys.swri.edu/rositraining/indigo/Exercises/ Has short Linux Intro

Questions?

Thank You
And Good Luck!