Wrocław University of Technology

Chair of Cybernetics and Robotics

ROS framework

Mariusz Janiak
p. 331 C-3, 71 320 26 44

Contents

1. Introduction
2. Basics
3. Tools

Introduction

ROS in one-liners

- *Robot Operating System* (NOT a complete standalone OS)
- Tools and libraries for developing robot applications
- Hardware abstraction, device drivers, visualizers, debugging tools, message-passing, package management, etc.
- Open source BSD license

Web page

- http://www.ros.org

Introduction

History

- Originally developed in 2007 at the Stanford Artificial Intelligence Laboratory and development continued at Willow Garage
- Since 2013 it is managed by OSRF (Open Source Robotics Foundation)
- Nine major releases so far, last *Jade*, future *Kinetic Kame* (planned May, 2016)
- Upcoming ROS2 (real-time, DDS, embedded devices first class citizen)

Introduction

Philosophical goals

- Agent based programming model
- Peer to peer
- Tools based software design
- Multiple language support (C++/Java/Python)
- Lightweight: runs only at the edge of your modules
- Free, open-source
- Suitable for large scale research and industry

Introduction

Toolset

- Creating ROS packages
- Building ROS nodes
- Running ROS nodes
- Viewing network topology
- Monitoring network traffic
- Not a single, monolithic program, lots of small processes instead

Introduction

Supported platforms

- Mobile manipulators
- Mobile robots
- Manipulators
- Autonomous cars
- Social Robots
- Humanoid
- UAVs
- AUVs
- UWVs
- Others

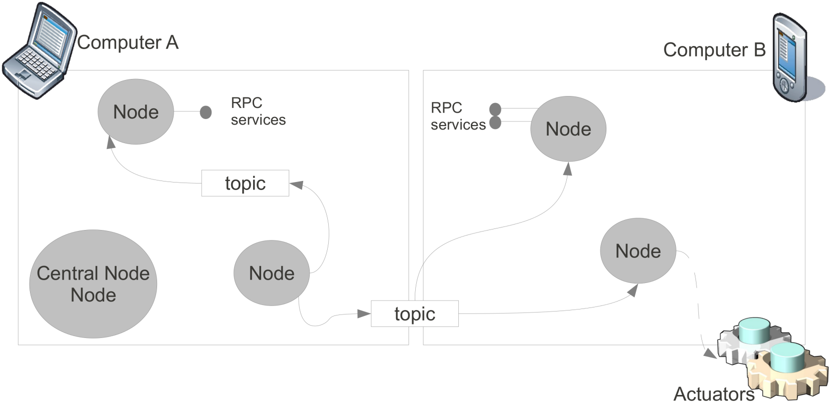
Basics

Distributed Architecture

- Hybrid P2P Architecture
- Distributed Components
- Several node communication mechanisms (message passing based)
- Focused on node communication mechanisms
- Free internal node design

Basics

Distributed Architecture



1

[1] P. I. Blasco, *Distributed Architecture, Deployment and Introspection*

Basics

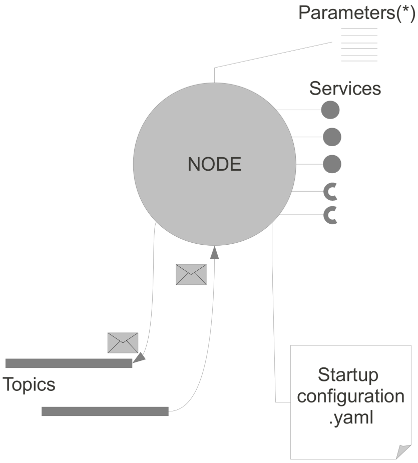
The ROS core (the minimal ROS runtime)

- ROS Master
  - Centralized XML-RPC server
  - Directory for publisher / subscribers / services
  - Negotiates communication connections
- Parameter Server
  - Centralized parameter repository
  - Provides parameter access to all nodes
  - XML-RPC data type
  - Not automatically update inside nodes
- rosout
  - Network-based stdout for human-readable messages
  - Subscribes to /out topic
  - Store output on filesystem

Basics

Node

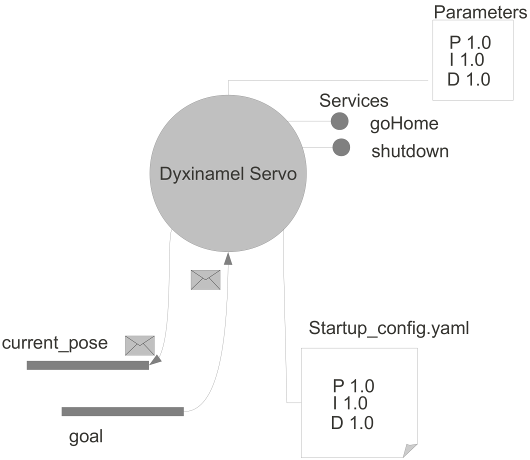
- Minimal building block
- Own control flow
- Single-purposed executable program
- Any supported language
- Communication mechanisms: topics, services, parameters
- Configurable (YAML files)



2

[2] P. I. Blasco, *Distributed Architecture, Deployment and Introspection*

Basics



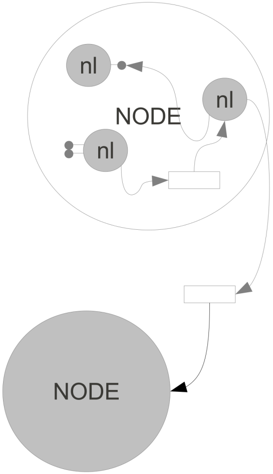
3

[3] P. I. Blasco, *Distributed Architecture, Deployment and Introspection*

Basics

Nodelet

- Threads
- Compatible node communication mechanisms
- Zero copy communication between nodelets
- Share Machine
- Only C++



4

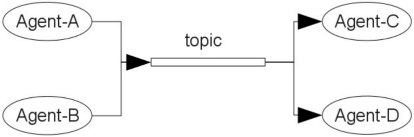
[4] P. I. Blasco, *Distributed Architecture, Deployment and Introspection*

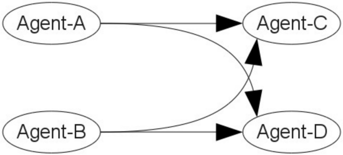
Basics

Topic

- Publish/Subscribe Model
- Many-To-Many
- Message passing based
- Strongly-typed (ROS .msg spec)
- Underlying Transport Layer
  - TCP, UDP, Shared Memory
  - rosserial, ethercat

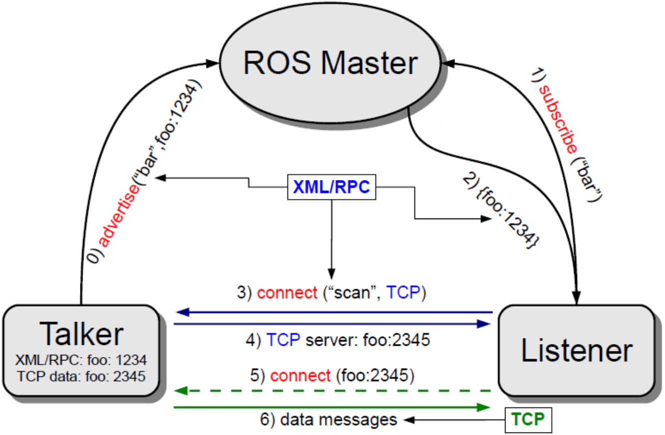
[5]





[5] P. I. Blasco, *Distributed Architecture, Deployment and Introspection*

Basics



6

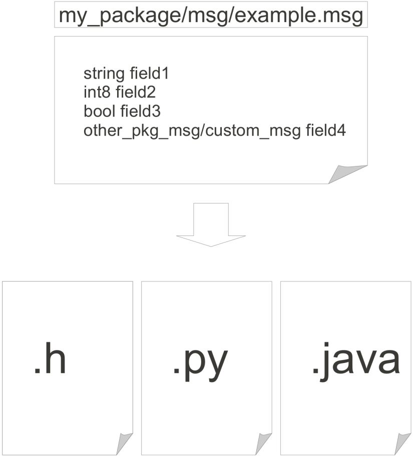
[6] R. Barraquand, A., Negre, *The Robotic Operating System at a Glance*

Basics

Message

- Data Structure
- Message Interface Description File
- Marshaling code generation: C++, Python, . . .
- Statically defined
- Standard message packages
  - geometry_msgs
  - sensor_msgs
  - navigation_msgs
  - . . .

[7]



my_package/msg/example.msg

```
string field1
int8 field2
bool field3
other_pkg_msg/custom_msg field4
```

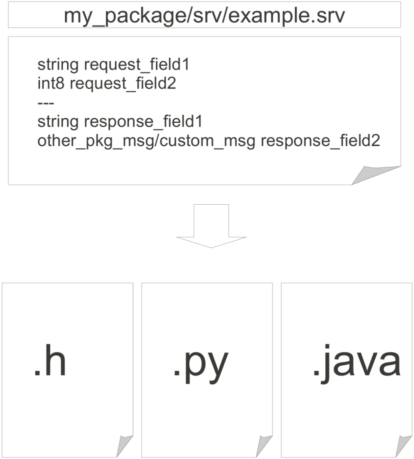
[7] P. I. Blasco, *Distributed Architecture, Deployment and Introspection*

Basics

Service

- RPC
- One-To-One (request-response)
- Strongly-typed (ROS .srv spec)
- Marshaling code generation: C++, Python, . . .
- Statically defined
- TCP/IP or UDP Transport

[8]



my_package/srv/example.srv

```
string request_field1
int8 request_field2
---
string response_field1
other_pkg_msg/custom_msg response_field2
```

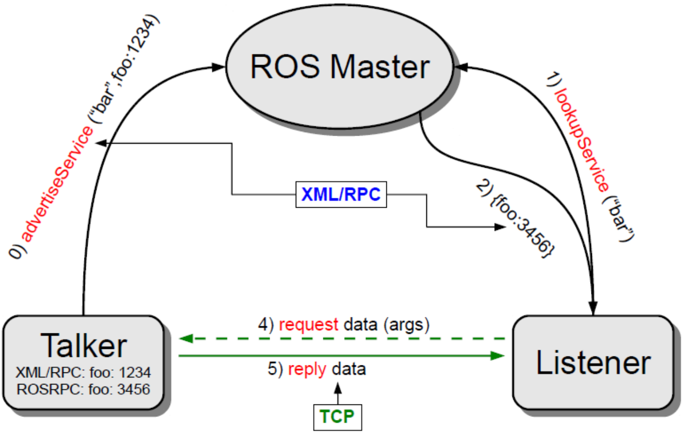
.h

.py

.java

[8] P. I. Blasco, *Distributed Architecture, Deployment and Introspection*

Basics



9

[9] R. Barraquand, A., Negre, *The Robotic Operating System at a Glance*

Basics

The ROS package

- Atomic unit of building
- Can contain anything
  - nodes
  - messages
  - tools
  - scripts
  - launch files
- In the most basic form
  - `package_name/CMakeLists.txt` – package cmake file
  - `package_name/package.xml` – catkin package manifest

The End

Thank you for your kind attention.