H2020 funded
GA no. 732287

rosin-project.eu

ROS-I Academy Training

ROS Computation Graph

MASCOR Institute

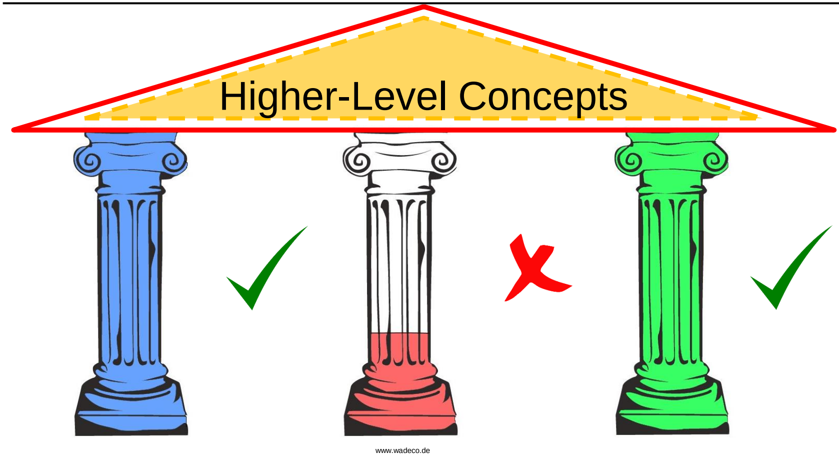
FH Aachen University

2017ff

FH AACHEN UNIVERSITY OF APPLIED SCIENCES

H2020 funded
GA no. 732287

ROS Level of concepts

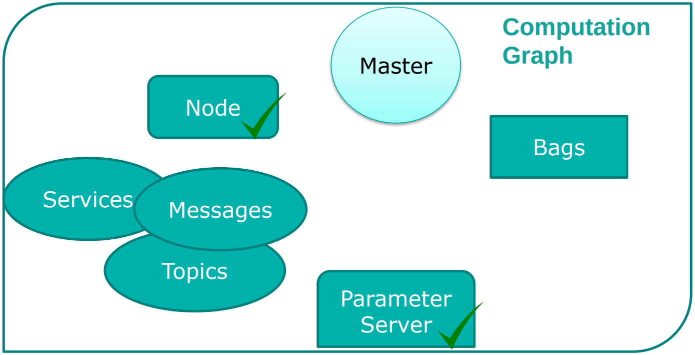


Filesystem Computation Graph Community

rosin-project.eu

ROS
Computation Graph

- The Computation Graph is the peer-to-peer network of ROS processes that are processing data together.



- Communication, Computation and Logging

rosin-project.eu

H2020 funded
GA no. 732287

ROS Computation Graph
Master

- One Master per system
- Registry for:
  - Nodes
  - Topics
  - Services
  - Parameters
- Part of the *roscore*

$\longrightarrow$ essential for all kind of processing and communication

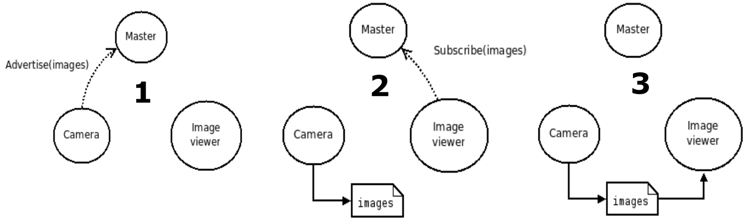
- To start the roscore:

```
roscore
```

H2020 funded
GA no. 732287

ROS Computation Graph
Master

- ROS Master provides naming and registration services:
  - **Nodes** Topics Services Parameters



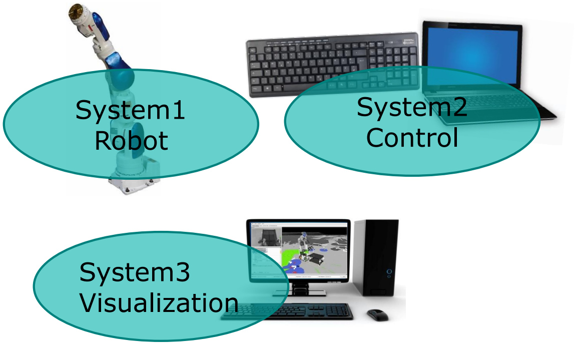
Camera likes to publish an image

Viewer likes to display an image

Master notifies Camera and viewer

After the nodes have located each other they communicate "peer-to-peer"

ROS Computation Graph
Master



→ Complex communication between different systems via Ethernet or serial connection or ...

ROSin

H2020 funded
GA no. 732287

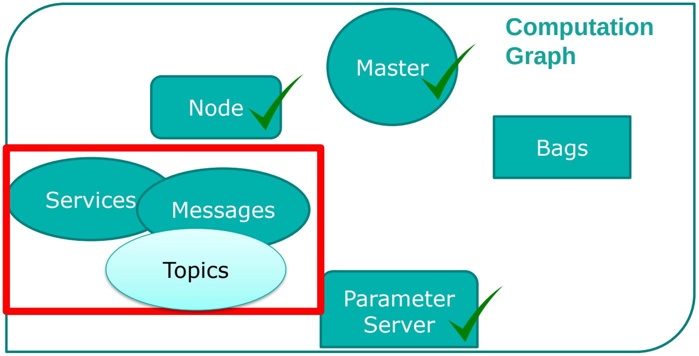
rosin-project.eu

ROS Computation Graph Master

→ ROS is a distributed computing environment. A running ROS system can comprise dozens, even hundreds of nodes, spread across multiple machines.

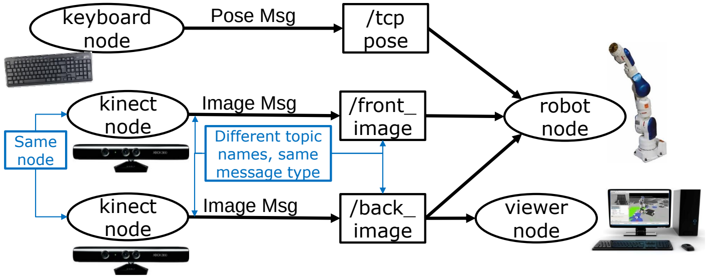
H2020 funded
GA no. 732287

ROS
Computation Graph



- Communication between nodes at runtime via Services, Messages and Topics

ROS Computation Graph
Topics

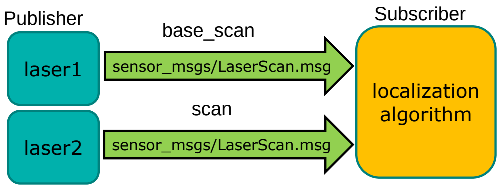


- **Topics** are named software buses over which nodes exchange messages
- A node sends out a message by publishing it to a given topic
- A node that is interested in a certain kind of data will subscribe to the appropriate topic
- A single node may publish and/or subscribe to multiple topics

H2020 funded
GA no. 732287

rosin-project.eu

ROS Computation Graph Topics



[partly from ROS-Industrial Basic Developer's Training Class, SWRI]

Characteristics

- Different topics can use the same message type
- Several nodes can subscribe to the same topic
- One node can subscribe to several topics
- Messages can be dropped
- Subscribers are event triggered
- Asynchronous Communication

Typical use

- Sensor data: laser scans, images, distance, I/O
- Feedback: robot position, status, battery level
- Open loop commands: desired position

ROS Computation Graph Publisher

Publisher

Definition

```
pub = rospy.Publisher('topic_name', std_msgs.msg.String, queue_size=1)
```

Publisher variable

Topic name

Message type

Queue size

Hint: Queue size for asynchronous publishing. Too large queue will lead to large latency. One lagging subscriber can block all publishing!

Fill message with data

```
ros_string      = std_msgs.msg.String()
ros_string.data = "Hello ROS!"
```

ROS Message

Variable in the message

Publication

```
pub.publish(ros_string)
```

H2020 funded
GA no. 732287

ROS Computation Graph
Subscriber

python

Subscriber

Definition

```
rospy.Subscriber('topic_name', std_msgs.msg.String, callback)
```

Topic name

Message type

Invoked function

Accessing to incoming message data

```
def callback(msg):
        value = msg.data
```

Always called if new data appear

First argument is the incoming message

rosin-project.eu

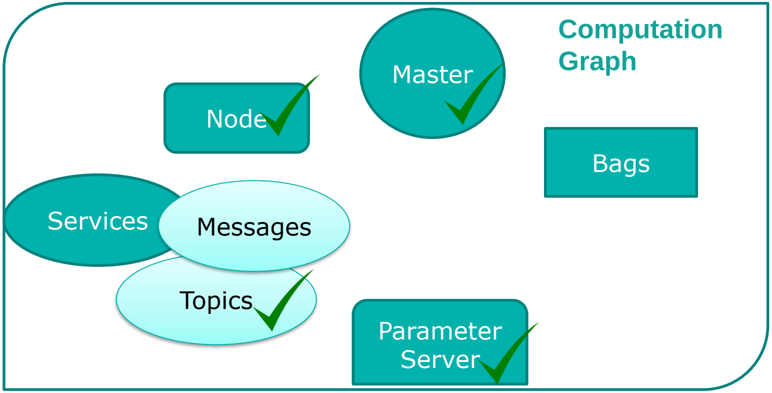
FH AACHEN
UNIVERSITY OF APPLIED SCIENCES

H2020 funded
GA no. 732287

rosin-project.eu

ROS
Computation Graph

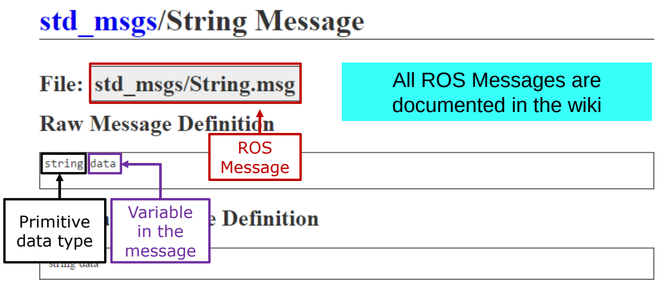
- ROS *Topics & Messages*



FH AACHEN UNIVERSITY OF APPLIED SCIENCES

H2020 funded
GA no. 732287

ROS Computation Graph Messages



std_msgs/String Message

File: std_msgs/String.msg

Raw Message Definition

```
string data
```

e Definition

```
string data
```

All ROS Messages are documented in the wiki

All ROS Messages are based on primitive data types like string, int32, uint8, float64 ... or on other ROS Messages.

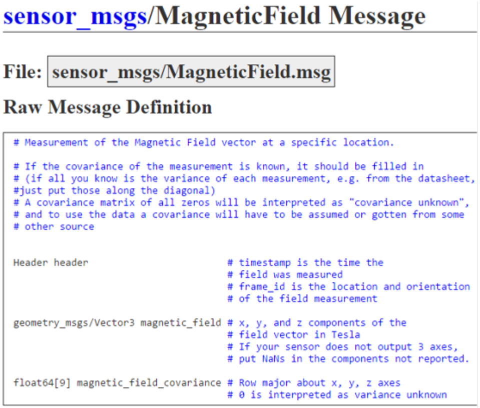
Careful! Use ROS messages for communication instead of Python or C++ variables!

rosin-project.eu

ROS Computation Graph Messages

Which data can I send / receive via topics?
Common ROS Message Types

- std_msgs (32 Types)
  - Bool
  - Byte
  - Float32
  - String ...
- geometry_msgs (29 Types)
  - Accel
  - Point
  - Vector3 ...
- sensor_msgs (26 Types)
  - Image
  - LaserScan
  - PointCloud ...
- actionlib_msgs, diagnostic_msgs, nav_msgs, viszualization_msgs
- + custom messages (unlimited types)


sensor_msgs/MagneticField Message

File: sensor_msgs/MagneticField.msg

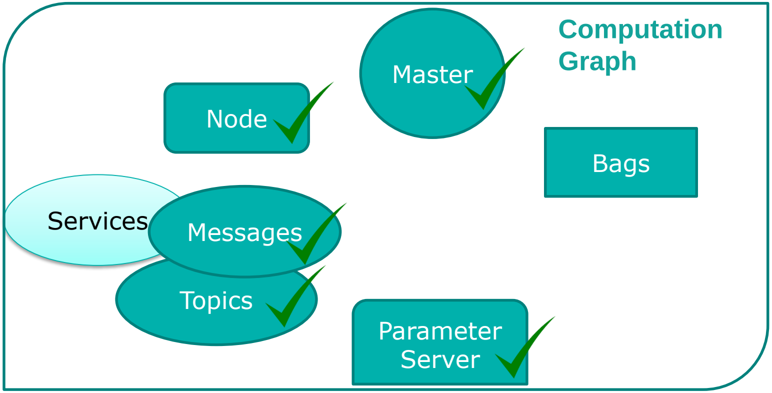
Raw Message Definition

```
# Measurement of the Magnetic Field vector at a specific location.

# If the covariance of the measurement is known, it should be filled in
# (if all you know is the variance of each measurement, e.g. from the datasheet,
#just put those along the diagonal)
# A covariance matrix of all zeros will be interpreted as "covariance unknown",
# and to use the data a covariance will have to be assumed or gotten from some
# other source


Header header                            # timestamp is the time the
                                         # field was measured
                                         # frame_id is the location and orientation
                                         # of the field measurement

geometry_msgs/Vector3 magnetic_field     # x, y, and z components of the
                                         # field vector in Tesla
                                         # If your sensor does not output 3 axes,
                                         # put NaNs in the components not reported.

float64[9] magnetic_field_covariance     # Row major about x, y, z axes
                                         # 0 is interpreted as variance unknown
```

H2020 funded
GA no. 732287

ROS
Computation Graph

- ROS *Services*

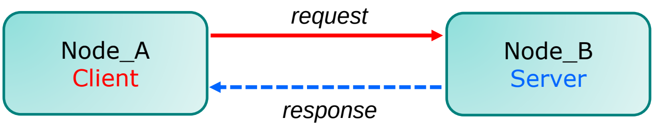


rosin-project.eu

H2020 funded
GA no. 732287

ROS Computation Graph
Services

rosin-project.eu



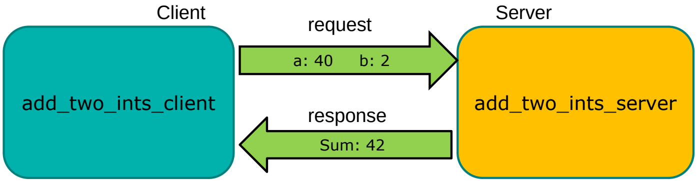
Services

- Request/reply is done via services
- Publish/subscribe model is not appropriate for request/reply interactions
- Pair of message structures: one for the request and one for the reply.
- Providing nodes offer a service under a name
- Client uses the service by sending the request message and awaiting the reply.

```
rosservice call /servicename [arg1] [arg2]
```

H2020 funded
GA no. 732287

ROS Computation Graph
Services



[partly from ROS-Industrial Basic Developer's Training Class, SWRI]

Characteristics

- Services are like remote function calls
- Code waits for service call to complete
- Use of message structures
- Synchronous Communication

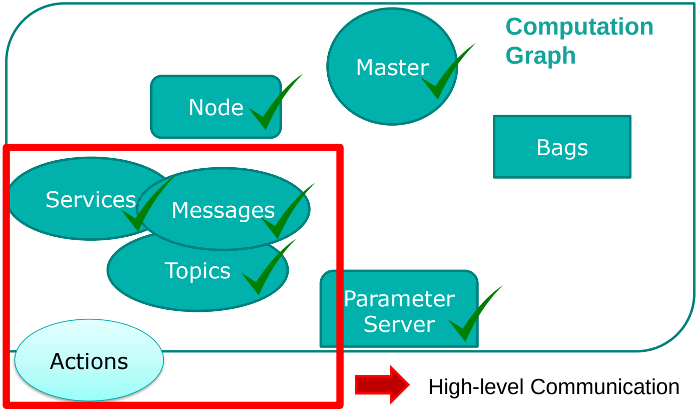
Typical use

- Algorithms: Forward or inverse transformation
- Closed-Loop Commands: Open gripper

rosin-project.eu

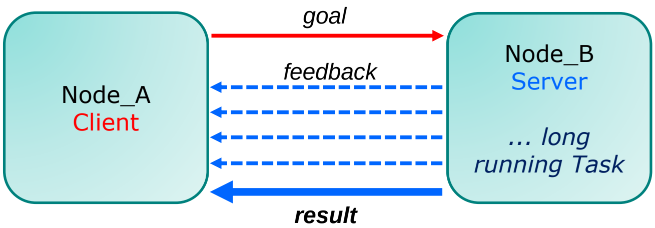
ROS
Computation Graph

- ROS *Actions* are not part of the computation graph
- Provided by the distributed ROS Tool action_lib



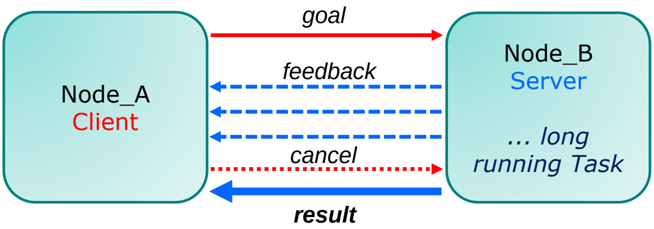
High-level Communication

ROS Computation Graph
Actions



- Comparable to services, but for long running tasks
  - Goal is sent by client
  - Feedbacks and Result are generated by server
- Client calls the action service by sending the goal message
  - **<u>Non</u>** Blocking mechanism (optional)
  - Continuous feedback to monitor current task

ROS Computation Graph
Actions



- Comparable to services, but for long running tasks
  - Goal is sent by client
  - Feedbacks and Result are generated by server
- Client calls the action service by sending the goal message
  - **Non** Blocking mechanism (optional)
  - Continuous feedback to monitor current task
  - Ability to cancel the request

H2020 funded
GA no. 732287

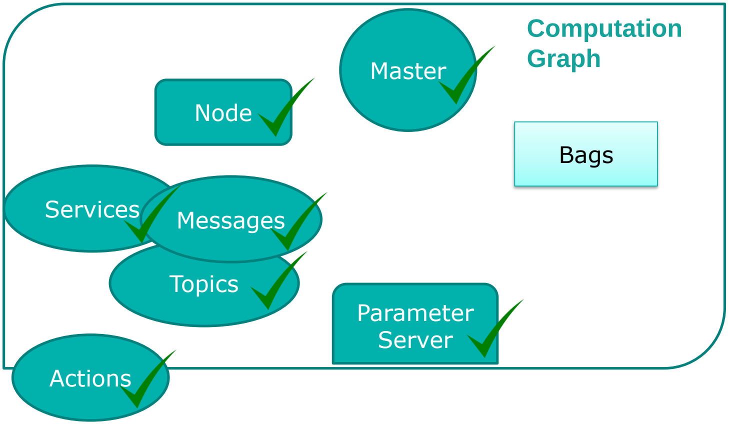
ROS Computation Graph Communication Summary

Type	Benefit	Drawback
Topic	• Good for most sensors • Easy to implement • One Pub – many Subs	• Messages can be dropped without knowlegde • System can be overloaded by too many messages
Service	• Knowledge of missed call • Well defined feedback	• Blocks until completion • Each service call has own connection: lower performance
Action	• Monitor long-running processes • Handshaking: knowledge of missed connection	• Quite complicated

rosin-project.eu

H2020 funded
GA no. 732287

ROS
Computation Graph



rosin-project.eu

ROS Computation Graph
Actions

Bags

- File format *(*.bag)* for storing and playing back messages
- Primary mechanism for data logging
- Important tool for analyzing, storing, visualizing data and testing algorithms.
- Use *rqt_bag* (rxbag is deprecated since Groovy) to visualize the data in a bag file

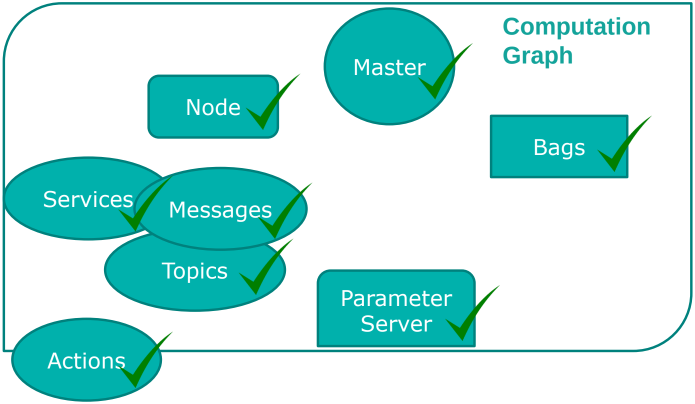
Using bag files within a ROS Computation Graph is generally no different from having ROS nodes send the same data!

```
rosbag record /topicname1 /topicname2
```

```
rosbag play /path/to/rosbag_file.bag
```

ROS
Computation Graph

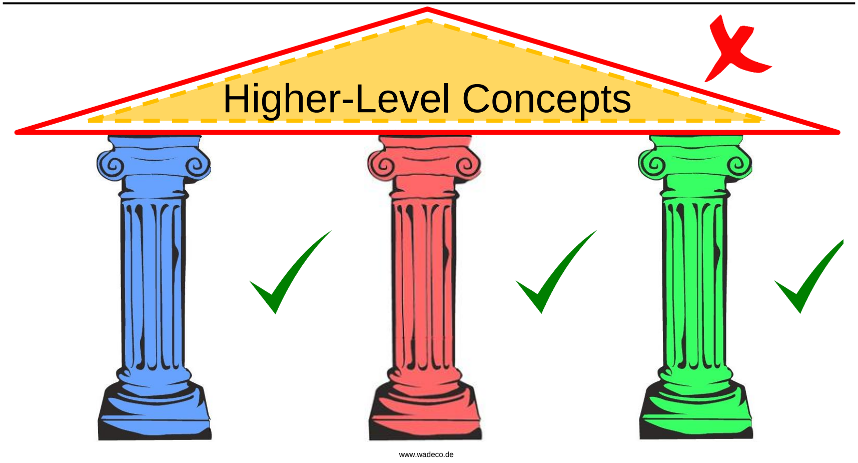
- Communication, Computation and Logging



BY-NC-ND

H2020 funded
GA no. 732287

ROS Level of concepts



Filesystem Computation Graph Community

rosin-project.eu

H2020 funded
GA no. 732287

ROS Tools

Start-up and Process Launch Tools

- roscore
- rosrun
- roslaunch
- roscd
- roscpack

Logging Tools

- rosbag
- rqt_bag

Introspection and Communication Tools

- rosmsg
- rossrv
- rqt_reconfigure

ROS Wiki: Cheatsheet

H2020 funded
GA no. 732287

ROS
Any questions?

http://www.allonrobots.com/

FH AACHEN UNIVERSITY OF APPLIED SCIENCES