INSTITUTO SUPERIOR TÉCNICO

INSTITUTO DE SISTEMAS E ROBÓTICA

Introduction to ROS
(Robot Operating System)

Rodrigo Ventura
Institute for Systems and Robotics — Lisboa
rodrigo.ventura@isr.ist.utl.pt

(July - 2011)

INSTITUTO SUPERIOR TÉCNICO

INSTITUTO DE SISTEMAS E ROBÓTICA

What is ROS?

- ROS = Robot Operating System

ROS.org

- Framework for robot software development providing operating system-like functionality
- Originated at Stanford Artificial Intelligence Lab, then further developed at Willow Garage

Willow Garage

- Supports all major host Operating Systems

(Ubuntu recommended) Linux Mac OS X Windows FreeBSD

- Large user base; getting widespread use
- ROS users forum: http://answers.ros.org

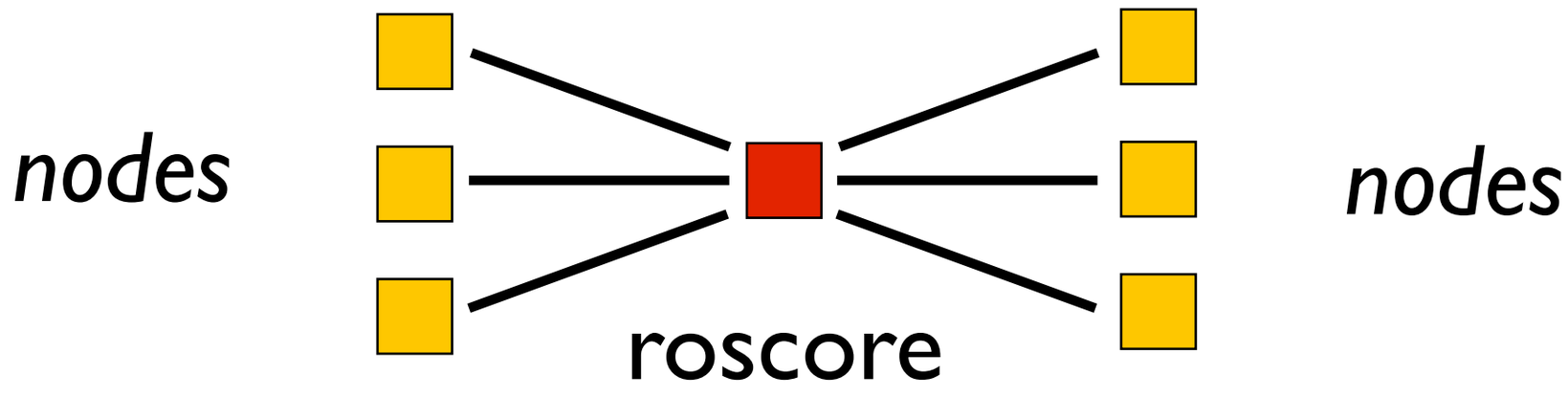
ROS ANSWERS

INSTITUTO SUPERIOR TÉCNICO

INSTITUTO DE SISTEMAS E ROBÓTICA

Basic concept #1: Node

- Modularization in ROS is achieved by separated operating system processes
- *Node* = a process that uses ROS framework
- Nodes may reside in different machines transparently
- Nodes get to know one another via roscore



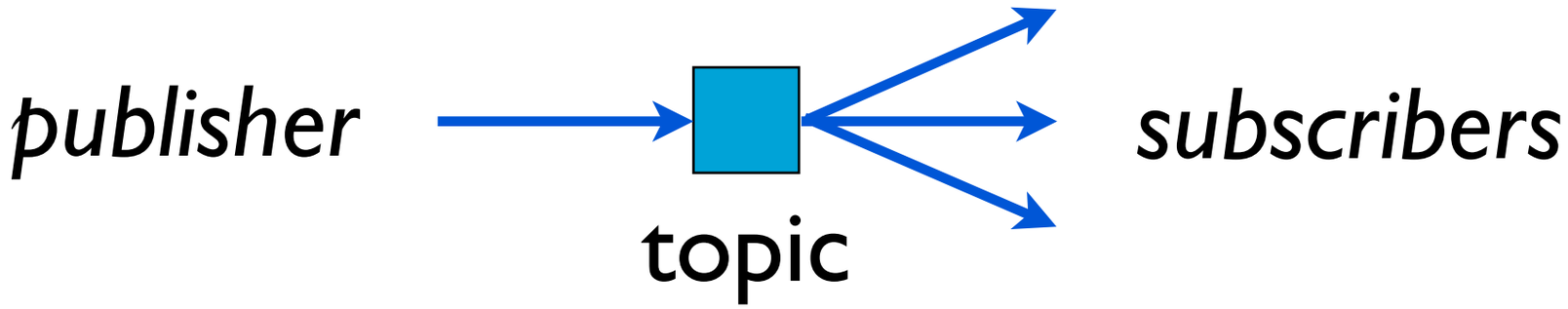
- roscore acts primarily as a name server
- Nodes use the roscore running in localhost by default overriden by the env. var. ROS_MASTER_URI

INSTITUTO SUPERIOR TÉCNICO

INSTITUTO DE SISTEMAS E ROBÓTICA

Basic concept #2: Topic

- *Topic* is a mechanism to send messages from a node to one or more nodes
- Follows a publisher-subscriber design pattern

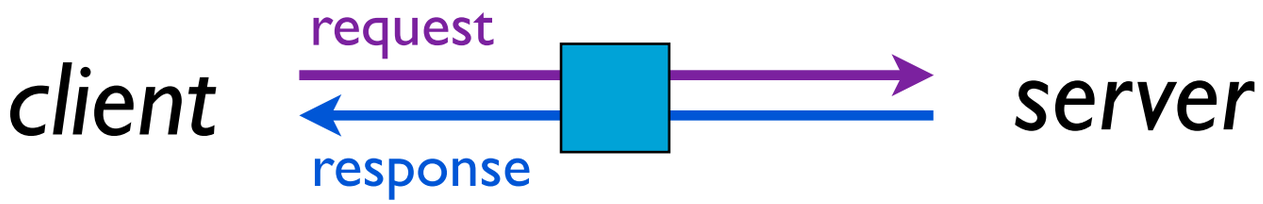


- *Publish* = to send a message to a topic

  Subscribe = get called whenever a message is published
- Published messages are broadcast to all Subscribers
- Example: LIDAR publishing scan data

Basic concept #3: Service

- *Service* is a mechanism for a node to send a request to another node and receive a response in return
- Follows a request-response design pattern



- A service is called with a request structure, and in return, a response structure is returned
- Similar to a Remote Procedure Call (RPC)
- Example: set location to a localization node

Message types

- All messages (including service requests/responses) are defined in text files
- Example: *built-in laser scan data message*

```
--- sensor_msgs/msg/LaserScan.msg ---

Header header            # timestamp in the header is the acquisition time of
                         # the first ray in the scan.
                         #
                         # in frame frame_id, angles are measured around
                         # the positive Z axis (counterclockwise, if Z is up)
                         # with zero angle being forward along the x axis

float32 angle_min        # start angle of the scan [rad]
float32 angle_max        # end angle of the scan [rad]
float32 angle_increment  # angular distance between measurements [rad]

float32 time_increment   # time between measurements [seconds] - if your scanner
                         # is moving, this will be used in interpolating position
                         # of 3d points
float32 scan_time        # time between scans [seconds]

float32 range_min        # minimum range value [m]
float32 range_max        # maximum range value [m]

float32[] ranges         # range data [m] (Note: values < range_min or > range_max should be discarded)
float32[] intensities    # intensity data [device-specific units].  If your
                         # device does not provide intensities, please leave
                         # the array empty.
```

INSTITUTO SUPERIOR TÉCNICO

INSTITUTO DE SISTEMAS E ROBÓTICA

Message types

- Another example: *remote interface service in Cobot*

request

response

```
--- cobot_msgs/srv/CobotRemoteInterfaceSrv.srv ---

# "Joystick" velocity commands:
float32 drive_x #Distance to move along x in meters
float32 drive_y #Distance to move along x in meters
float32 drive_r #Distance to turn in radians

# command_num must increment every time the service is called - used to reject out of sync commands
int32 command_num

# valid command flags:
#   CmdMove = 0x0001
#   CmdSetLocation = 0x0002
#   CmdGetLocation = 0x0004
#   CmdGetParticlesSampling = 0x0010
#   CmdSetTarget = 0x0020
int32 command_type

# The following parameters are used for commands CmdSetLocation and CmdSetTarget
float32 loc_x
float32 loc_y
float32 orientation
string map

---

float32 loc_x
float32 loc_y
float32 orientation

float32[] particles_x
float32[] particles_y
float32[] particles_weight
float32[] locations_weight

int8 err_code
```

INSTITUTO SUPERIOR TÉCNICO

INSTITUTO DE SISTEMAS E ROBÓTICA

How - To

- Two major languages are supported:
  - Python
  - C++
- ROS provides ready-to-use build system
- *Package* = self-contained directory containing sources, makefiles, builds, etc.
  - how to create an empty package:

    ```
    $ roscreate-pkg name
    ```
- *Stack* = set of packages grouped together
  - how to create an empty stack:

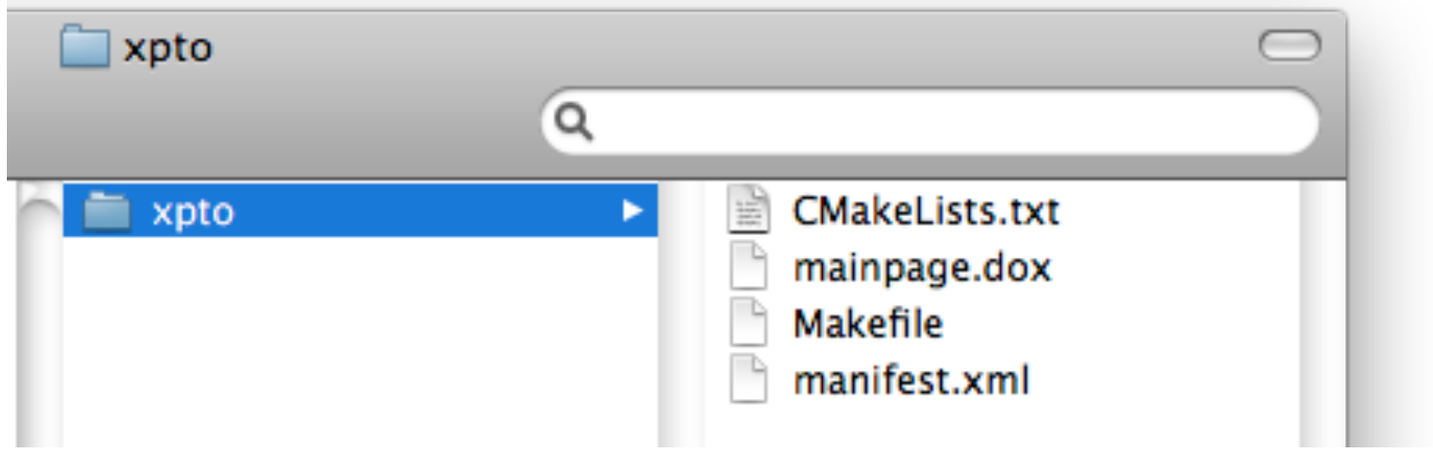
    ```
    $ roscreate-stack name
    ```

INSTITUTO SUPERIOR TÉCNICO

INSTITUTO DE SISTEMAS E ROBÓTICA

Example: *creating package xpto*

```
$ roscreate-pkg xpto
```



--- xpto/manifest.xml ---

```
<package>
  <description brief="xpto">

     xpto

  </description>
  <author>Rodrigo Ventura</author>
  <license>BSD</license>
  <review status="unreviewed" notes=""/>
  <url>http://ros.org/wiki/xpto</url>

</package>
```

--- xpto/CMakeLists.txt ---

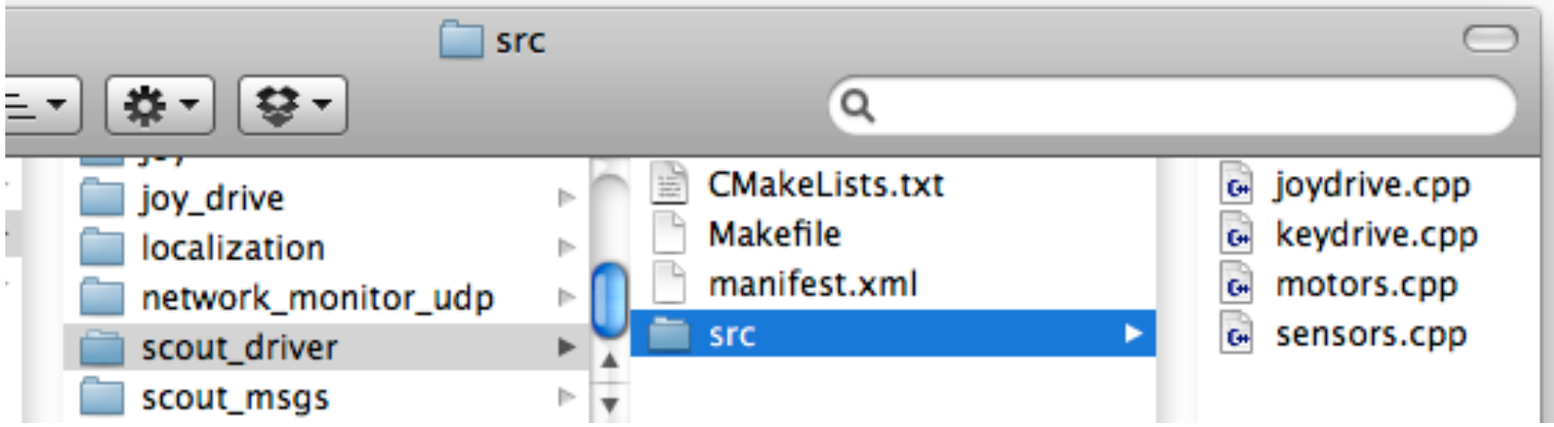
```
[...]
#uncomment if you have defined messages
#rosbuild_genmsg()
#uncomment if you have defined services
#rosbuild_gensrv()

#common commands for building c++ executables and libraries
#rosbuild_add_library(${PROJECT_NAME} src/example.cpp)
#target_link_libraries(${PROJECT_NAME} another_library)
#rosbuild_add_boost_directories()
#rosbuild_link_boost(${PROJECT_NAME} thread)
#rosbuild_add_executable(example examples/example.cpp)
#target_link_libraries(example ${PROJECT_NAME})
```

INSTITUTO SUPERIOR TÉCNICO

INSTITUTO DE SISTEMAS E ROBÓTICA

Example: *drivers for scout robot*



--- scout_drivers/manifest.xml ---

```
<package>
  <description brief="scout_driver">

     scout_driver

  </description>
  <author>Rodrigo Ventura</author>
  <license>BSD</license>
  <review status="unreviewed" notes=""/>
  <url></url>
  <depend package="std_msgs"/>
  <depend package="roscpp"/>
  <depend package="scout_msgs"/>
  <depend package="joy"/>

</package>
```

--- scout_drivers/CMakeLists.txt ---

```
[...]
rosbuild_add_executable(motors src/motors.cpp)

rosbuild_add_executable(sensors src/sensors.cpp)

rosbuild_add_executable(keydrive src/keydrive.cpp)
target_link_libraries(keydrive ncurses)

rosbuild_add_executable(joydrive src/joydrive.cpp)
```

How to build a package:

```
$ make
```

or

```
$ rosmake
```

Subscribing to a Topic

- Example: subscribing to LIDAR scan data

```
# Initialize rospy
NODE_NAME = 'localization'
import roslib; roslib.load_manifest(NODE_NAME)
import rospy

# Import LaserScan message type
from sensor_msgs.msg import *

# Scan message handler
def lidar_handler(msg):
   # this code is executed whenever a scan is published
   [...]

# Main function
def main():
   rospy.init_node(NODE_NAME)
   rospy.Subscriber("/scan", LaserScan, lidar_handler)
   rospy.spin()
```

topic name

message type (python class)

INSTITUTO SUPERIOR TÉCNICO

INSTITUTO DE SISTEMAS E ROBÓTICA

Publishing to a Topic

- Example: sending a string to a topic

```
# Initialize rospy
NODE_NAME = 'localization'
import roslib; roslib.load_manifest(NODE_NAME)
import rospy

# Import standard String message type
from std_msgs.msg import *

# Main function
def main():
   pub = rospy.Publisher(“/scout/viewer”, String)
   rospy.init_node(NODE_NAME)
   [...]
   msg = “Hello world”
   pub.publish( String(msg) )
   [...]
```

topic name

message type

publish method

message constructor

INSTITUTO SUPERIOR TÉCNICO

INSTITUTO DE SISTEMAS E ROBÓTICA

Calling a Service

- Example: reseting localization algorithm

```
# Initialize rospy
NODE_NAME = 'viewer'
import roslib; roslib.load_manifest(NODE_NAME)
import rospy

# Import LocalizationSrv service description
from scout_msgs.srv import *

# Main function
def main():
   srv = rospy.ServiceProxy("/scout/localization", LocalizationSrv)
   rospy.init_node(NODE_NAME)
   [...]
   response = srv(1, x, y, theta)
   [...]
```

service call arguments service name service type

INSTITUTO SUPERIOR TÉCNICO

INSTITUTO DE SISTEMAS E ROBÓTICA

Implementing a Service

- Example: reseting localization algorithm

```
# Initialize rospy
NODE_NAME = 'localization'
import roslib; roslib.load_manifest(NODE_NAME)
import rospy

# Import LocalizationSrv service description
from scout_msgs.srv import *

# Service handler
def handler(req):
   # this code is executed whenever the service is called
   [...]
   return LocalizationSrvResponse()

# Main function
def main():
   rospy.init_node(NODE_NAME)
   rospy.Service("/scout/localization", LocalizationSrv, handler)
   rospy.spin()
```

response constructor

service name

service type

INSTITUTO SUPERIOR TÉCNICO

INSTITUTO DE SISTEMAS E ROBÓTICA

Command line tools

```
$ rostopic
```

```
rostopic is a command-line tool for printing information about ROS Topics.

Commands:
        rostopic bw     display bandwidth used by topic
        rostopic echo   print messages to screen
        rostopic find   find topics by type
        rostopic hz     display publishing rate of topic
        rostopic info   print information about active topic
        rostopic list   list active topics
        rostopic pub    publish data to topic
        rostopic type   print topic type

Type rostopic <command> -h for more detailed usage, e.g. 'rostopic echo -h'
```

INSTITUTO SUPERIOR TÉCNICO

INSTITUTO DE SISTEMAS E ROBÓTICA

Command line tools

```
$ rosservice

Commands:
        rosservice args print service arguments
        rosservice call call the service with the provided args
        rosservice find find services by service type
        rosservice info print information about service
        rosservice list list active services
        rosservice type print service type
        rosservice uri  print service ROSRPC uri

Type rosservice <command> -h for more detailed usage, e.g. 'rosservice call -h'
```

INSTITUTO SUPERIOR TÉCNICO

INSTITUTO DE SISTEMAS E ROBÓTICA

Command line tools

```
$ rosbag
```

```
Usage: rosbag <subcommand> [options] [args]

Available subcommands:
   check
   compress
   decompress
   filter
   fix
   help
   info
   play
   record
   reindex

For additional information, see http://code.ros.org/wiki/rosbag/
```

INSTITUTO SUPERIOR TÉCNICO

INSTITUTO DE SISTEMAS E ROBÓTICA

Q & A