

Inria
INVENTORS FOR THE DIGITAL WORLD

Need for a middleware

Robotic system:

- many hardware components:
 - computers,
 - network,
 - motor controllers,
 - sensors,
 - ...
- many software components:
 - operating system,
 - drivers,
 - control,
 - perception,
 - ...
- research.

Putting it all together: **middleware**.

Need for a middleware

Robotic system:

- many hardware components:
 - computers,
 - network,
 - motor controllers,
 - sensors,
 - ...
- many software components:
 - operating system,
 - drivers,
 - control,
 - perception,
 - ...
- research.

Putting it all together: **middleware**.

Need for a middleware

Robotic system:

- many hardware components:
 - computers,
 - network,
 - motor controllers,
 - sensors,
 - ...
- many software components:
 - operating system,
 - drivers,
 - control,
 - perception,
 - ...
- research.

Putting it all together: **middleware**.

Need for a middleware

Robotic system:

- many hardware components:
 - computers,
 - network,
 - motor controllers,
 - sensors,
 - ...
- many software components:
 - operating system,
 - drivers,
 - control,
 - perception,
 - ...
- research.

Putting it all together: **middleware**.

Structure

Central concept:

- processing,

Structure

Central concept:

- processing,

Processing units:

- **node** (process),
- nodelet (thread);

Structure

Central concept:

- processing,

Processing units:

- **node** (process),
- nodelet (thread);

Organization:

- **package**:
 - node(s),
 - definitions,
 - compilation unit;
- **catkin**:
 - compilation system based on `cmake`,
 - dependency handling,
 - packaging/deployment.

Communication

Communication between nodes:

- **messages**:
 - message passing,
 - grouped in **topics**;

Communication

Communication between nodes:

- **messages:**
 - message passing,
 - grouped in **topics**;
- **services:**
 - remote procedure call,
 - request and answer messages;

Communication

Communication between nodes:

- **messages:**
 - message passing,
 - grouped in **topics**;
- **services:**
 - remote procedure call,
 - request and answer messages;
- **actions:**
 - tasks with significant duration,
 - preemptible,
 - given feedback;

Communication

Communication between nodes:

- **messages:**
 - message passing,
 - grouped in **topics**;
- **services:**
 - remote procedure call,
 - request and answer messages;
- **actions:**
 - tasks with significant duration,
 - preemptible,
 - given feedback;
- statically typed.

Topics

Initialization:

- **publisher**: node declaring writing on a topic,
- **subscriber**: node declaring listening to a topic (using a callback),

Topics

Initialization:

- **publisher**: node declaring writing on a topic,
- **subscriber**: node declaring listening to a topic (using a callback),
- several publishers/subscribers allowed,
- order irrelevant,
- require a directory;

Topics

Initialization:

- **publisher**: node declaring writing on a topic,
- **subscriber**: node declaring listening to a topic (using a callback),
- several publishers/subscribers allowed,
- order irrelevant,
- require a directory;

Communication:

- publisher transmits to each subscriber,
- no need for directory.

Services

Initialization:

- server: node advertising a service,
- client: node asking for a proxy on a given service,

Services

Initialization:

- server: node advertising a service,
- client: node asking for a proxy on a given service,
- require a directory;

Services

Initialization:

- server: node advertising a service,
- client: node asking for a proxy on a given service,
- require a directory;

Request:

- client sends a request,
- server processes and answers,
- no need for directory.

Actions

Initialization:

- action server: node advertising an action server,
- action client: node asking connection to action server,

Actions

Initialization:

- action server: node advertising an action server,
- action client: node asking connection to action server,
- require a directory;

Actions

Initialization:

- action server: node advertising an action server,
- action client: node asking connection to action server,
- require a directory;

Request and execution:

- client sends a **goal**,
- server starts execution (interrupting current task if needed),
- server gives goal task reference to client,
- server gives continuous **feedback**,
- task finished: server report **result**,
- no need for directory.

rosmaster

rosmaster

- directory:
 - publishers,
 - subscribers,
 - services;
- provides XMLRPC API;
- not a central communication node;
- part of **roscore**;
- nodes know of it through ROS_MASTER_URI environment variable.

roscore

roscore:

- rosmaster,
- parameter server,
- log aggregator.

roscore

roscore:

- rosmaster,
- parameter server,
- log aggregator.

Parameter server:

- centralized parameter repository,
- XMLRPC data types.

Launching

Launching a robotic system:

- several processes,
- on different computers;

Launch files specify:

- list of nodes,
- parameter values,
- in XML syntax.

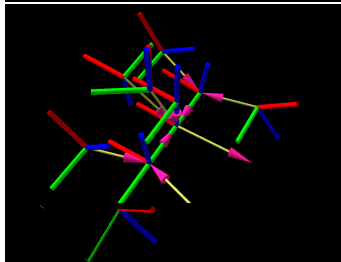
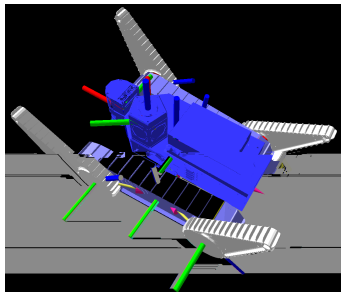
Transformation frames

Robot:

- set of rigid bodies

In ROS:

- set of transformation frames,
- linked by transformations,
- arranged in a directed tree,
- published on a single `/tf` topic,
- rich API to extract information from that tree.



Summary of concepts

Structure:

- nodes, in packages;

Communication:

- messages,
- services,
- actions,
- peer-to-peer;

Launching:

- launch files;

Transformations:

- /tf.

Runtime inspection

Nodes:

- list nodes,
- get communication information;

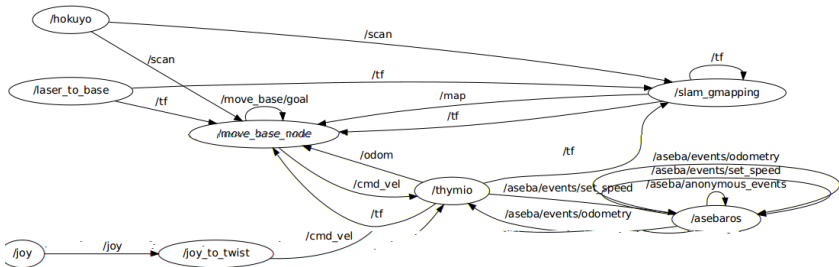
Runtime inspection

Nodes:

- list nodes,
- get communication information;

Connection:

- rqtgraph



Runtime inspection

Topics:

- list topics,
- see messages,
- get type information;

Runtime inspection

Topics:

- list topics,
- see messages,
- get type information;

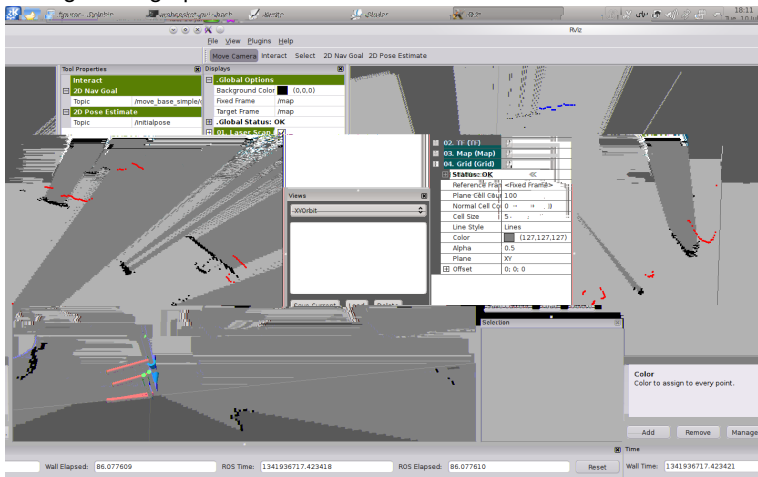
/tf:

- inspect /tf tree,
- compute transformations;

Visualization

rvi z:

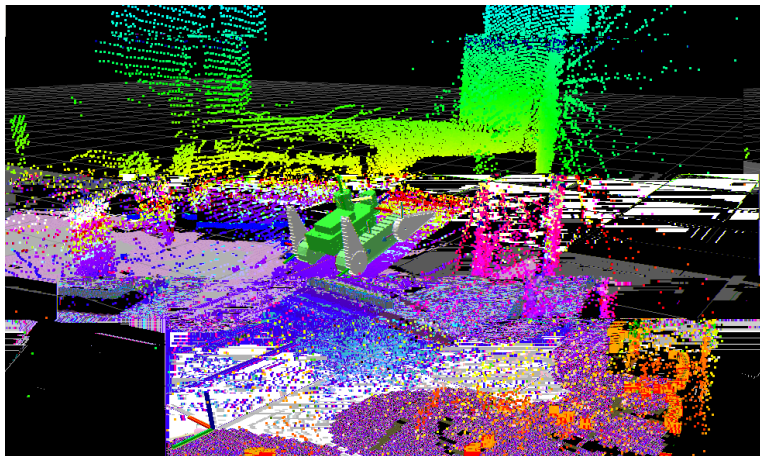
- configurable graphical interface



Visualization

rvi z:

- full 3D visualization



Recording

Recording messages:

- container: **bagfile**,
- rosbag: generic subscriber;

Replaying messages:

- rosbag: generic publisher,
- offline testing of perception pipeline.

Third party tools

Hardware drivers:

- plenty of common sensors,
- some actuators,
- some robots;

Third party tools

Hardware drivers:

- plenty of common sensors,
- some actuators,
- some robots;

Software stacks:

- mapping,
- navigation,
- 3D perception,
- simulation,
- ...

Third party tools

Hardware drivers:

- plenty of common sensors,
- some actuators,
- some robots;

Software stacks:

- mapping,
- navigation,
- 3D perception,
- simulation,
- ...

Important community.

Conclusion

ROS: not the first/only middleware for robotics

- generalist and strongly funded,
- developed by multiple experienced scientists,
- provides a comprehensive tool suit;

Conclusion

ROS: not the first/only middleware for robotics

- generalist and strongly funded,
- developed by multiple experienced scientists,
- provides a comprehensive tool suit;

Impact around the world:

- umbrella project providing a default choice,
- increasing code exchange between researchers,
- standardization of data types, additional conventions,
- strong set of development and monitoring tools.

The Challenges

Uncontrolled growth:

- 1000s of packages, varying level of quality, maturity, etc.
- How to find the reliable ones?
- How to just find the relevant ones?

The Challenges

Uncontrolled growth:

- 1000s of packages, varying level of quality, maturity, etc.
- How to find the reliable ones?
- How to just find the relevant ones?

Maintenance and support:

- How to guarantee continuing support?
- How to integrate community input at the core level?
- How to take hard decision (API changes, . . .)?
- Who specifies what? Is there continuity?

The Challenges

Uncontrolled growth:

- 1000s of packages, varying level of quality, maturity, etc.
- How to find the reliable ones?
- How to just find the relevant ones?

Maintenance and support:

- How to guarantee continuing support?
- How to integrate community input at the core level?
- How to take hard decision (API changes, . . .)?
- Who specifies what? Is there continuity?

ROS2.0:

- multi-robot, realtime, embedded, production-ready;
- DDS as communication backend,
- API change,
- ???

Thanks for your attention.