SDVUNx: *sdv_un_ros* package

sdv-un-ros is a software metapackage developed for use on SDVUNx robots. It uses the tools provided by the ROS platform and allows the robotics software provided by this set of tools to be used together with the software necessary to adapt the robot to the platform.

ROS-Melodic version, is used in SDVUNx mobile robots: this is because the projects began their development in 2016 using the ROS-Kinetic version at that time. ROS-Melodic requires Ubuntu 18.04, a stable Linux distribution with excellent software tools.

The fundamental software parts are described below. For more details of each component, enter the respective links.

What is Ubuntu?

Ubuntu is an operating system based on Debian. It is part of the GNU/Linux family and contains mostly open source software. Its installation and use are free. Ubuntu is presented as an alternative to Windows operating systems and is widely accepted in the field of servers and mainframes. Ubuntu combines the advantages of open source software with the characteristics of a stable version operating system, that is, an operating system that does not change much once it is released in order to maintain compatibility with the software that was released with the system at the time. This is an advantage especially if you use software that is not updated very often, as in the case of the SDVUNx robots.

The version of Ubuntu used in the SDVUNx robots is Ubuntu 18.04 Server LTS. The server versions contain software tools focused on remote administration: this is very useful for the use of robots, since it can be considered as a remote system that does not have a screen, even inside the laboratory. Being an LTS version, the Ubuntu parent company is committed to supporting this operating system for at least 5 years: however, an extended support of 10 years has already been announced, which is very advantageous.

Using Ubuntu Server can be difficult, especially if you’re coming from a Windows-type environment. It is recommended that you meet the following knowledge requirements to start using Ubuntu software.

- Use of command terminal
- bash
- SSH
- Basic notions of wireless networks

What is ROS?

ROS (Robot Operating System) is a software platform that contains tools that allow the construction and use of programs for robotics. It is an open source project that leverages other open projects to build a robust environment.

ROS consists of two major versions: ROS1, which is used in SDVUNx robots, and ROS2, the new version that seeks to replace ROS1, promising greater compatibility and less dependency on the operating system. ROS1 is installed on specific versions of Ubuntu and was released every two years, at the same time as the LTS versions of Ubuntu. Since the development of the SDVUNx software started in 2016, when ROS2 did not exist yet, ROS1-Melodic is used.

The installation and use of ROS require several minimum prior knowledge:

- Use of command terminal in Linux
- bash
- Python
- Use of APT (*Advanced Packaging Tool*)

What is a node?

In ROS, programs and their communications are represented using graph theory: programs are represented as nodes and communication channels are represented as edges. A node is then a program that uses the ROS environment and can connect to other nodes via edges. Nodes can be programmed in multiple languages, the most used being Python and C++. In order to use the ROS features, you must import the ROS libraries.

The idea behind representing programs as nodes is to be able to convert all the complex tasks that a robot can do into simpler tasks that are easier to develop and maintain, also promoting code reuse. ROS contains many general-purpose nodes, intended to be used in any robotics implementation, making it easy to build software suites tailored to any project. A sample with nodes an topics is in Figure 67.

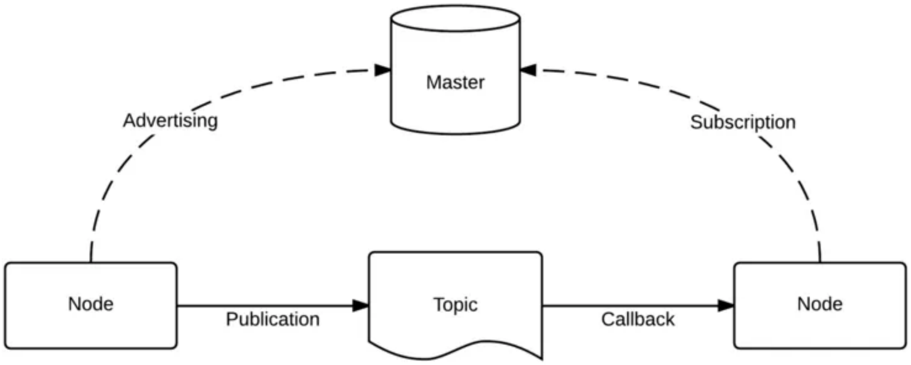

Fig. 67 ROS master node connected with two nodes that publish and listens messages through topics.

What is a topic?

In ROS, the communication channels between the nodes, that is, the edges, are called *topics*. These channels are characterized by having a well-defined message type, with a clear data structure. Any node can *subscribe* to a topic and receive or post messages.

ROS contains a nice set of standard messages to be used in topics. It is also possible to create new messages in case the ones provided by ROS are not enough. Understanding what the different nodes do and what topics they use to function is the most important part of using ROS.

What is it a package in ROS?

The nodes, services, messages and libraries that have a common purpose within ROS are grouped into *packages*. ROS distributes most of these tools grouped in packages such as *move-base* or *map-server*.

What is a metapackage in ROS?

Metapackages are collections of packages in ROS. They are often used to group the packages of a specific solution, for example, the packages used for a specific type of robot or a specific project. The *sdv_un_ros* metapackage contains packages with a clear distinction in their functions and purposes: the metapackage is intended for the SDVUN project.

What is the Navigation Stack?

It is a set of ROS software packages designed to be used in mobile robots. It contains nodes, messages, libraries, and services that are intended for pose estimation, path calculation, sensor reading, error correction, and robot motion. Although not a metapackage as such, the *Navigation Stack* is the de facto solution for any mobile robot with a certain degree of autonomy. *sdv_un_ros* depends on several of the packages included in the *Navigation Stack* and these are defined as dependencies in the metapackage declaration.

Package details

Related Links:

- SDVUNx: *sdv_un_ros* installation
  - Software Requirements
    - Installation
- SDVUNx: how to use *sdv_un_ros* packages
- SDVUNx: *sdv_un_ros* packages summary
  - SDVUNx: *sdv_msgs* package
    - Messages list
    - Dependencies
  - SDVUNx: *sdv_nav* package
    - Nodes
    - *launch* files
    - Important nodes
      - Navigation Stack
      - AMCL
      - HECTOR Mapping
      - Map server
    - SDVUNx Parameters
    - Package dependencies
  - SDVUNx: *sdv_serial* package
    - Nodes
    - *launch* files
    - *serial_sdv_node* topics
    - *sdv_serial_node* node flowchart
    - Package dependencies
  - SDVUNx: *sdv_process* package
    - External package *ros_coms*
    - Nodes