AVSIMULATION

Innovate > Simulate > Accelerate

SCANER™STUDIO TUTORIAL

- **INTRODUCTION**
- This tutorial is made to explain how to interface SCANeR™studio with ROS (Robot Operating System).
- Robot Operating System (ROS) is a set of open source computing tools for developing software for robotics.

PLAN

- **SCANeR™studio and ROS: environment and concepts.**
- **Create a ROS Workspace for SCANeR™studio.**
- **Create a catkin package for SCANeR™studio.**
- **The Code: SCANeR™ API and ROS.**
- **Building: SCANeR™ API and ROS.**
- **Running: SCANeR™ API and ROS.**

PLAN

- **SCANeR™studio and ROS: environment and concepts.**

SCANER™STUDIO TUTORIAL

Environment:

Prerequisite: The SCANeR™studio license has to be Cluster or Advanced (license with more than one machine).

- Windows environment:
  . Windows 10 64-bits.
  . Last version of SCANeR™studio 1.8 with its related Add-on Linux package (available on our official website).
- Linux environment:
  . Ubuntu 16.04.3 LTS (Xenial Xerus) amd64.
  . A network installation of SCANeR™studio 1.8 must be available. You can refer to the **README.Linux,** it comes with the Linux Add-on package
- ROS environment:
  . kinetic – 1.12.13

Note: This tutorial is made for a “Shared” installation of SCANeR™studio. If required, it is also possible to do the same when the installation of SCANeR™studio on Linux is a “Standalone” installation. In thus case, a Workstation license (1 machine) of SCAneR™studio is sufficient (see the README.Linux for the Standalone installation).

SCANER™STUDIO TUTORIAL

- **Concepts**

SCANeR™studio	ROS	Description
Modules	Nodes	A module/node is an executable that uses SCANeR™/ROS to communicate with other modules/nodes.
Messages	Messages	SCANeR™/ROS data type used when subscribing or publishing to a DataInterface/topic.
DataInterface	Topics	Modules/Nodes can publish messages on a DataInterface/topic as well as subscribe to a DataInterface/topic to receive messages.

PLAN

- **Create a ROS Workspace for SCANeR™studio.**

SCANER™STUDIO TUTORIAL

- **Note :**

1. You must have installed and configured a ROS environment (a tutorial is available at the following link: http://wiki.ros.org/kinetic/Installation/Ubuntu).

We recommend you to add the ‘source’ to your shell startup script (.bashrc):

```
source /opt/ros/kinetic/setup.bash
```

2. In addition to the network installation of SCANeR™studio and to be able to use the SCANeR™studio SDK, you need to get the following file:

sudo apt-get install cmake build-essential


```
sudo apt-get install cmake build-essential
```

SCANER™STUDIO TUTORIAL

1. Create and build a catkin workspace:

Execute the following command lines:

```
mkdir –p $STUDIO_PATH%/SCANeRstudio_1.8/APIs/samples/ScanerAPI/SampleROS/catkin_ws/src

cd $STUDIO_PATH/SCANeRstudio_1.8/APIs/samples/ScanerAPI/SampleROS/catkin_ws/

catkin_make
```

PLAN

- **Create a catkin package for SCANeR™studio.**

SCANER™STUDIO TUTORIAL

To Create a catkin Package, execute the following command lines:

```
cd $STUDIO_PATH/SCANeRstudio_1.8/APIs/samples/ScanerAPI/SampleROS/catkin_ws/src

catkin_create_pkg scanerapi_sample_ros_gateway_folder scanerapi_sample_ros_gateway std_msgs roscpp
```

This will create a “scanerapi_sample_ros_gateway_folder” which contains a “package.xml” and a “CMakeLists.txt”, which have been partially filled out with the information you gave.

PLAN

- **The Code: SCANeR™API and ROS.**

SCANER™ STUDIO TUTORIAL

1. Create a “src” directory :

Execute the following command lines:

```
cd $STUDIO_PATH/SCANeRstudio_1.8/APIs/samples/ScanerAPI/SampleROS/catkin_ws/src/scanerapi_sample_ros_gateway_folder

mkdir –p src
```

2. Copy the “scanerapi_sample_ros_gateway.cpp” file in the folder: $STUDIO_PATH/SCANeRstudio_1.8/APIs/samples/ScanerAPI/SampleROS/catkin_ws/src/scanerapi_sample_ros_gateway_folder/src (you will find the .cpp file in the .zip folder).

SCANER™STUDIO TUTORIAL

The contents of the .cpp file is the following:

```
#include "ros/ros.h"
#include "std_msgs/String.h"
#include "ScanerAPI/scanerAPI_DLL_C.h" //SCANeRTM API: C language Functions
#include "ScanerAPI/ScanerAPImessagesNetwork.h" //SCANeRTM API: Network utils
#include <sstream>



int main(int argc, char **argv)
{
ros::init(argc, argv, "scanerAPI_sample_ros_gateway");
ros::NodeHandle n;
Process_Init(argc, argv); //SCANeRTM API: Process initialization
ros::Publisher chatter_pub = n.advertise<std_msgs::String>("chatter", 100000);
DataInterface* vehicle_0 = Com_declareInputData(NETWORK_IVEHICLE_VEHICLEUPDATE, 0); //SCANeRTM API: Message to read
while (ros::ok())
{
Process_Wait(); //SCANeRTM API: Frequency synchronization
Process_Run(); //SCANeRTM API: Run the process
if (Process_GetState() == PS_RUNNING) //SCANeRTM API: The simulation is running
{
Com_updateInputs(UT_AllData); //SCANeRTM API: Update input data (data to read)
std_msgs::String msg;
std::stringstream ss;
ss << "Speed[0]:" << Com_getFloatData(vehicle_0, "speed[0]")*3.6; //SCANeRTM API: Read vehicle 0's speed on X;//SCANeRTM API: Read vehicle 0's speed.
msg.data = ss.str();
ROS_INFO("%s", msg.data.c_str());
chatter_pub.publish(msg);
ros::spinOnce();
}
}
Process_Close(); //SCANeRTM API: Clean way to stop the SCANeRTM module
return 0;
```

PLAN

- **Building: SCANeR™ API and ROS.**

SCANER™STUDIO TUTORIAL

1. Copy and replace the CMakeLists.txt (you will find the CMakeLists.txt in the .zip folder): (…\ScanerAPI\SampleROS\catkin_ws\src\scanerapi_sample_ros_gateway_folder\CMakeLists.txt):

```
###########
## Build ##
###########

## Specify additional locations of header files
## Your package locations should be listed before other locations
include_directories(
# include
../../../../../../include
  ${catkin_INCLUDE_DIRS}
)

link_directories(
../../../../../../../bin/Linux/ubuntu/16.04/lib
../../../../../../../bin/Linux/ubuntu/16.04/lib/external
)

## Declare a C++ library
# add_library(${PROJECT_NAME}
#   src/${PROJECT_NAME}/scaner_ros_gateway.cpp
# )

## Add cmake target dependencies of the library
## as an example, code may need to be generated before libraries
## either from message generation or dynamic reconfigure
# add_dependencies(${PROJECT_NAME} ${${PROJECT_NAME}_EXPORTED_TARGETS} ${catkin_EXPORTED_TARGETS})

## Declare a C++ executable
## With catkin_make all packages are built within a single CMake context
## The recommended prefix ensures that target names across packages don't collide
add_executable(${PROJECT_NAME}_node src/scanerapi_sample_ros_gateway.cpp)

## Rename C++ executable without prefix
## The above recommended prefix causes long target names, the following renames the
## target back to the shorter version for ease of user use
## e.g. "rosrun someones_pkg node" instead of "rosrun someones_pkg someones_pkg_node"
# set_target_properties(${PROJECT_NAME}_node PROPERTIES OUTPUT_NAME node PREFIX "")

## Add cmake target dependencies of the executable
## same as for the library above
add_dependencies(${PROJECT_NAME}_node ${${PROJECT_NAME}_EXPORTED_TARGETS} ${catkin_EXPORTED_TARGETS})

## Specify libraries to link a library or executable target against
target_link_libraries(${PROJECT_NAME}_node libScanerAPI.so
   ${catkin_LIBRARIES}
)
```

SCANER™STUDIO TUTORIAL

2. Run:

```
cd $STUDIO_PATH/SCANeRstudio_1.8/APIs/samples/ScanerAPI/SampleROS/catkin_ws

catkin_make
```

```
rigaut@rigaut-virtualbox:~/AVS/SCANeRstudio_1.8.58/APIs/samples/ScanerAPI/SampleROS/catkin_ws2$ catkin_make
Base path: /home/rigaut/AVS/SCANeRstudio_1.8.58/APIs/samples/ScanerAPI/SampleROS/catkin_ws2
Source space: /home/rigaut/AVS/SCANeRstudio_1.8.58/APIs/samples/ScanerAPI/SampleROS/catkin_ws2/src
Build space: /home/rigaut/AVS/SCANeRstudio_1.8.58/APIs/samples/ScanerAPI/SampleROS/catkin_ws2/build
Devel space: /home/rigaut/AVS/SCANeRstudio_1.8.58/APIs/samples/ScanerAPI/SampleROS/catkin_ws2/devel
Install space: /home/rigaut/AVS/SCANeRstudio_1.8.58/APIs/samples/ScanerAPI/SampleROS/catkin_ws2/install
####
#### Running command: "make cmake_check_build_system" in "/home/rigaut/AVS/SCANeRstudio_1.8.58/APIs/samples/ScanerAPI/SampleROS/catkin_ws2/build"
####
####
#### Running command: "make -j3 -l3" in "/home/rigaut/AVS/SCANeRstudio_1.8.58/APIs/samples/ScanerAPI/SampleROS/catkin_ws2/build"
####
[100%] Built target scanerapi_sample_ros_gateway_folder
rigaut@rigaut-virtualbox:~/AVS/SCANeRstudio_1.8.58/APIs/samples/ScanerAPI/SampleROS/catkin_ws2$
```

PLAN

- **Running: SCANeR™ API and ROS.**

SCANER™STUDIO TUTORIAL

Prerequisite:

- roscore must be running.
- You need to have a SCANeR™studio configuration with at least 2 machines (Windows: supervisor and Linux: SCANeR™ API/ROS).
- The SCANeR™studio Daemon must be running on both machines.

Note:

In this example and for all the SCANeR™ API projects, the values are sent with the SI units (here, the speed of the vehicle 0 will be sent in m/s).

SCANER™STUDIO TUTORIAL

To run the project:

1. Now, to be able to launch the ROS SCANeR™studioprocess, you need to set the environment variables of SCANeR™studio. To do that, go to your shared folder and source the setenv.sh script :

```
. ./setenv.sh SCANeRstudio_1.8
```

Note: We recommend adding this command line to the .bashrc file.

2. Source the new setup .*sh

```
source devel/setup.bash
```

SCANER™STUDIO TUTORIAL

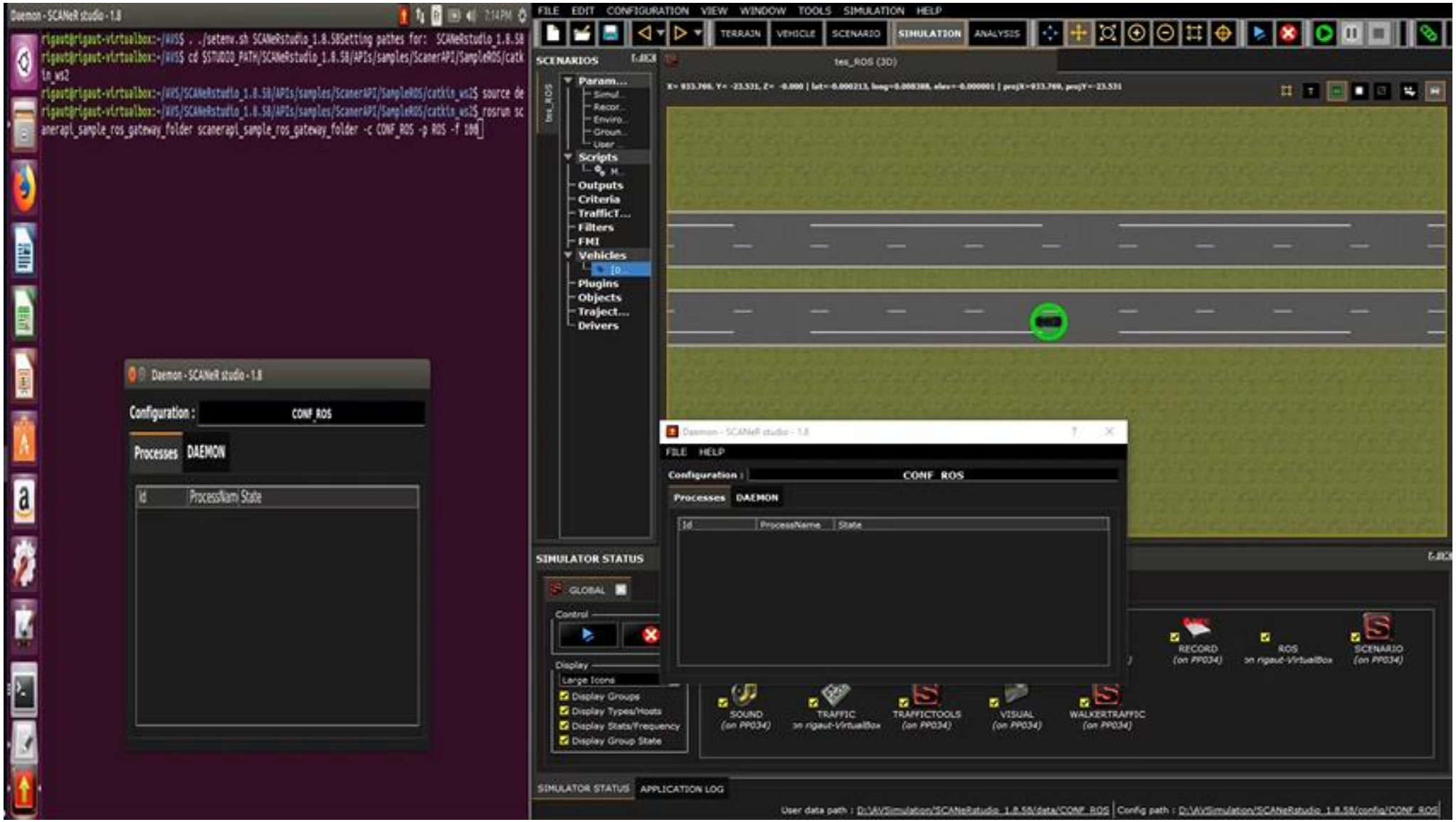
3. Execute the following command lines:

```
cd $STUDIO_PATH/SCANeRstudio_1.8/APIs/samples/ScanerAPI/SampleROS/catkin_ws/

rosrunscanerapi_sample_ros_gateway_folderscanerapi_sample_ros_gateway_folder_node–c
<YOUR_CONFIGURATION> –p ROS –f 100
```

The parameter–c is your SCANeR™studio configuration name.
The parameter -p is your Process name (must be available into your SCANeR™studio configuration).
The parameter –f is the Frequency of the process (optional, the default value is 100hz).

SCANER™STUDIO TUTORIAL



Europe Office

1, Cours de l’Ile Seguin
92650 Boulogne - Billancourt
+ 33 1 46 94 97 40

—

US Office

AdduXi Inc.
2791 Research Drive
Rochester Hills
MI 48309 USA

—

www.avsimulation.fr
support-scaner@avsimulation.fr

AVSIMULATION