



Model-based Design for Safety Critical Controller

Design with ROS and Gazebo

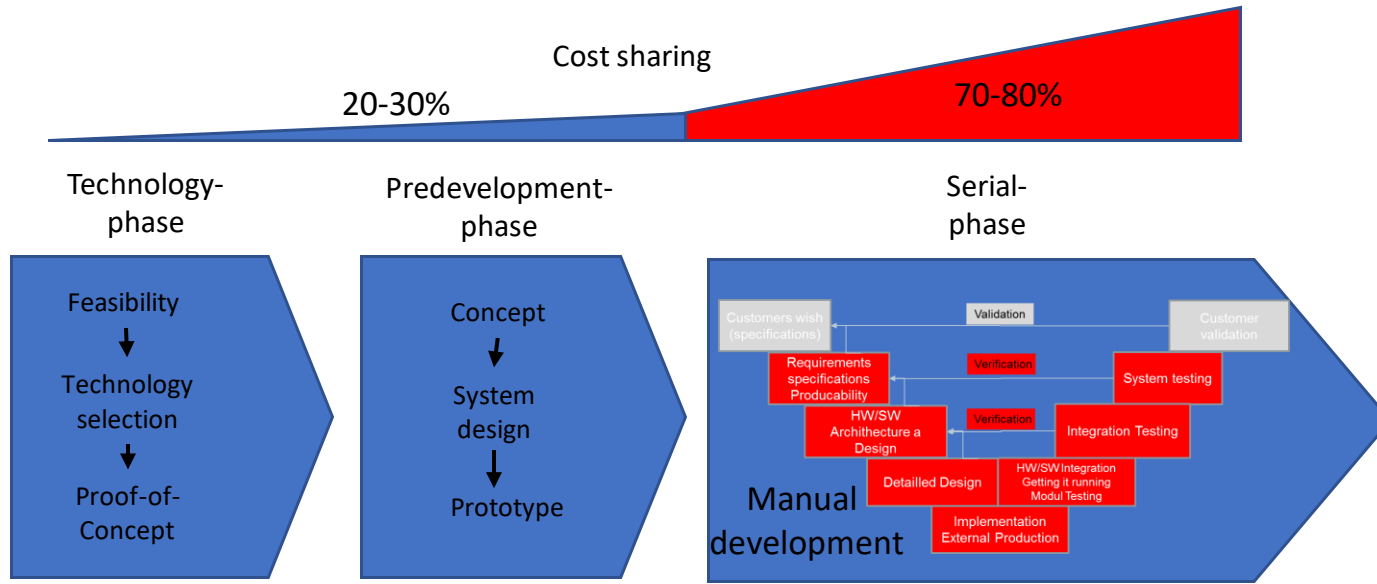
Michael Naderhirn, Mischa Köpf, Josef Mendler

Background (as I understand it)



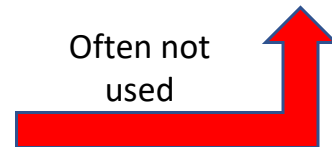
- Conformity means that a development was done according to safety critical standard
 - ISO 26262 – Automotive
 - DO-178C, DO-330-333 – Aeronautic
 - IEC62061 – Machinery
 - If “something bad” happens companies must show that the development was done conforming to the relevant safety critical standard
 - An independent auditing company helps a company through auditing that the development processes conform e. g. DO-178C
- Certification means that an authority certifies that a development is allowed to be used
 - UAV is allowed to fly into the civil aerospace (e. g. flies according the rules of the air)
 - Autonomous car is allowed to drive on the road (e. g. drives according the rules)
 - Conformity can be part of the certification
- Qualification means that a tool is qualified to be used for safety critical development
 - Documentation that a code generator generates correct code – e. g. tests show that the generated code produces the right and wrong results correctly
 - An independent auditing company audits the software used by company, the methods to test the software and the documentation

Product Development Cycle

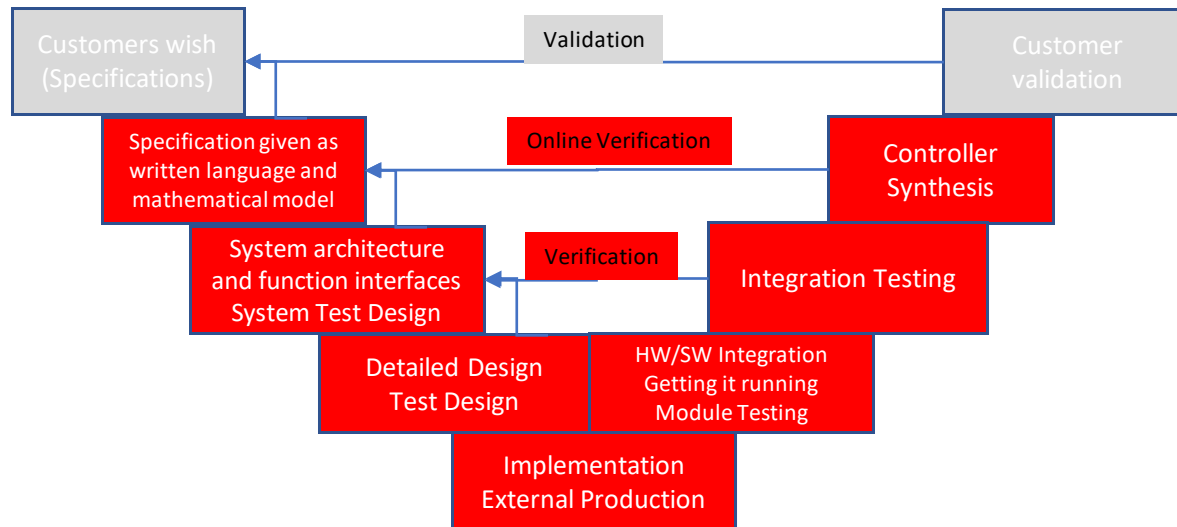


OpenModelica

Know How after
predevelopment-phase



Basic Design Steps



- Defining the system spec
- Hazard and risk analysis
- Determine Safety Integrity Level (SIL)
- Define controller structure and necessary redundancies

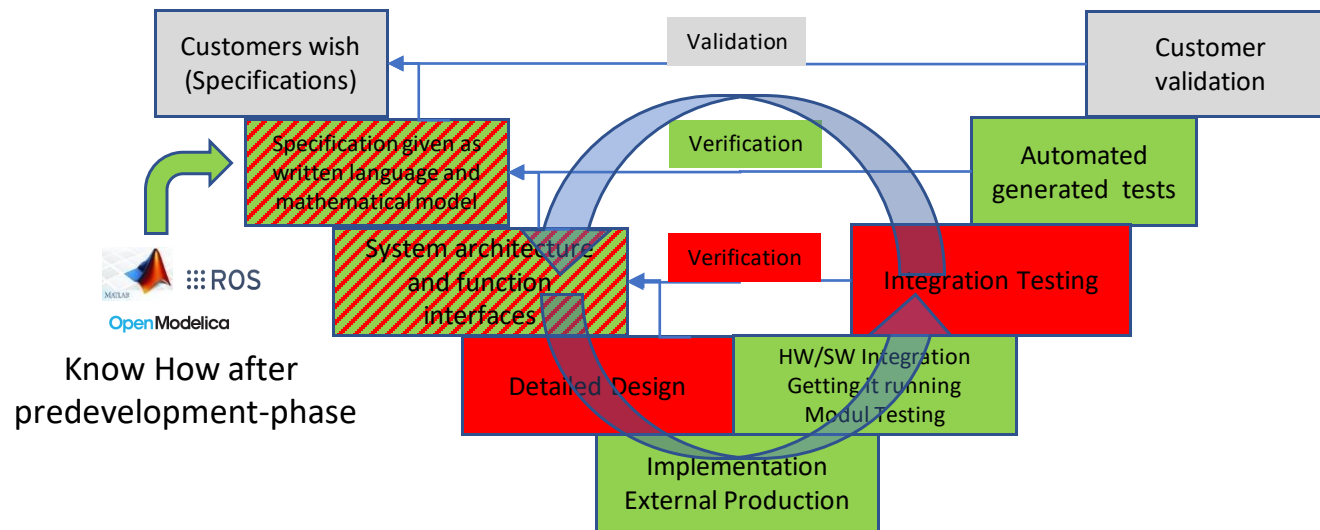
- Develop tests for the system, subsystem and modules
- Develop and test modules
- Integrate modules with subsystems and test
- Integrate subsystem with system and test
- Validate system with customer

How to get better?



- New mathematical methods to automatize development process
 - On- and Offline system verification
 - Verified deep neural networks
- Listen to customer (some important findings)
 - Development in 90% preferably done on Windows computers
 - ROS is used in app. 80-90% of prototyping of robotic solutions
 - SMEs have problems to afford commercial development tools – some stop robotic projects after prototyping
 - Robotics engineer needs to be a software architect
 - Available knowhow and packages should be reusable
 - > High degree of automation required

Incremental (Agile) Development

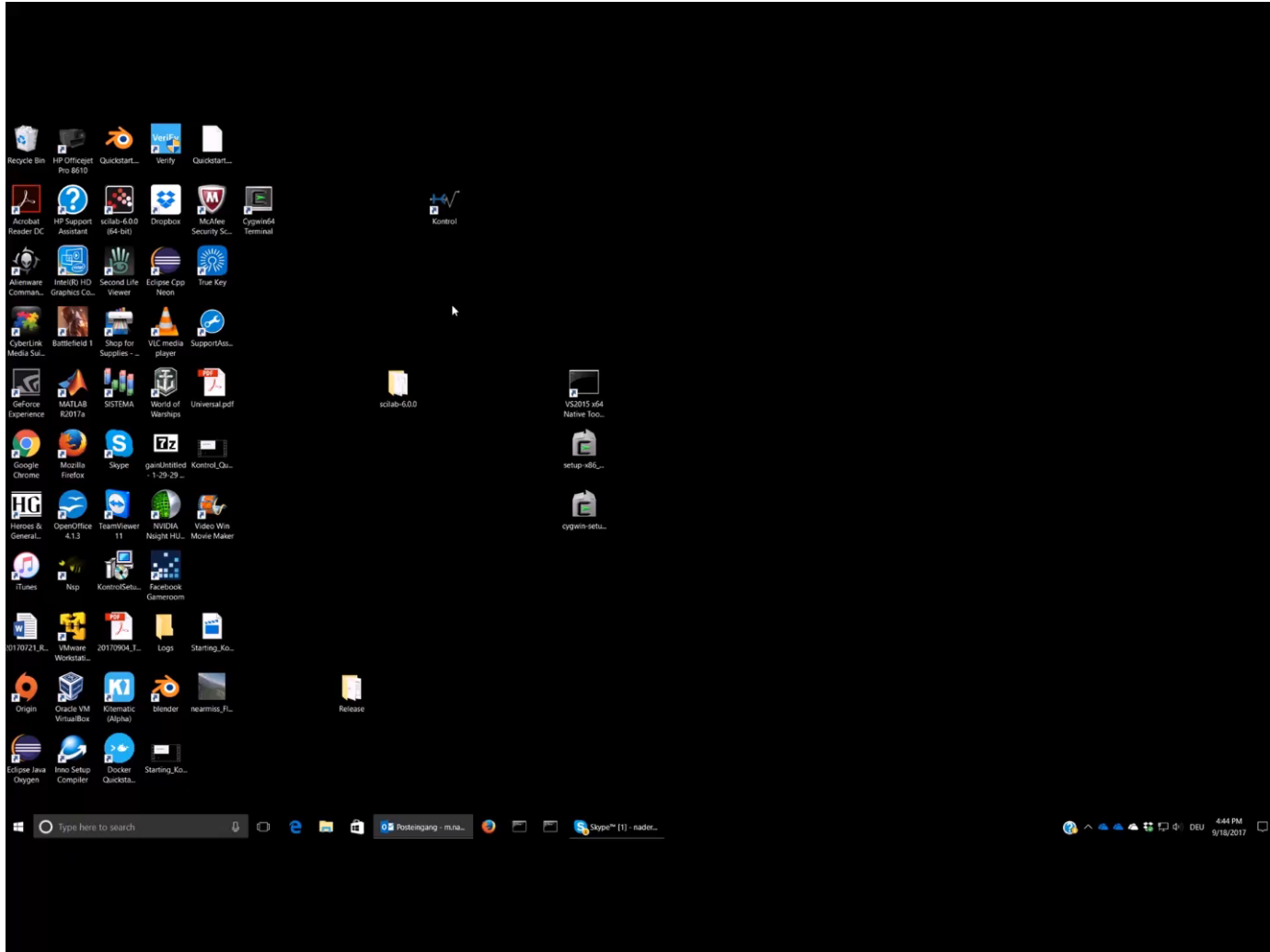


Agile development of mechatronic systems allows faster time2market

Our Solution - Kontrol

- Integrates
 - ROS / ROS2
 - Gazebo
- Imports ROS packages and prototypes automatically
- Model based design of ROS structure
- Software distribution to hardware and automatic configuration
- Manual coding or model based design
- Allows to add and edit Gazebo world
- Scilab connects to ROS network automatically from Windows
- Generates Code for ROS (and ROS2)
- Independent from simulation environment
- Future – Implements an incremental development process for safety critical controller design for mechatronic systems

Our Solution - Kontrol



Import of ROS Packages

The image shows a screenshot of a computer desktop with two windows open. The background window is the ROS.org website, displaying a list of ROS packages for the kinetic distribution. The foreground window is the ROS-Industrial GUI, showing a list of ROS packages and a description of the 'abb_driver' package.

ROS.org Website:

ROS.org
Documentation Browse Software News Download

fuerte groovy hydro indigo jade kinetic lunar melodic

packages stacks metapackages search

Browsing packages for kinetic

Name	Maintainers / Authors	Description
abb_driver	Shaun Edwards, Jeremy Zoss, Shaun Edwards	<p> ROS-Industrial nodes for interfacing with ABB robot controllers. </p> <p> This n
abb_irb1200_support	Levi Armstrong (Southwest Research Institute)	
abb_irb120_gazobo	Mark Culleton (Trinity College Dublin), Kevin Kelly (Trinity College Dublin)	
abb_irb120_moveit_config	Mark Culleton (Trinity College Dublin)	
abb_irb120_support	Mark Culleton (Trinity College Dublin), Levi Armstrong (SwRI)	
abb_irb120t_moveit_config	Mark Culleton (Trinity College Dublin)	
abb_irb1600_6_12_moveit_config	Koerthana Manivannan	
abb_irb1600_support	Austin Deric (Southwest Research Institute)	
abb_irb2400_moveit_config	Shaun Edwards (Southwest Research Institute)	
abb_irb2400_moveit_plugins	Shaun Edwards (Southwest Research Institute), Jeremy Zoss	
abb_irb2400_support	Shaun Edwards (Southwest Research Institute)	

ROS-Industrial GUI:

Control - abb_driver

Project Help

Visual Machines Software Architecture Editor Gazebo Editor Hardware Architecture Software Distribution Implementation

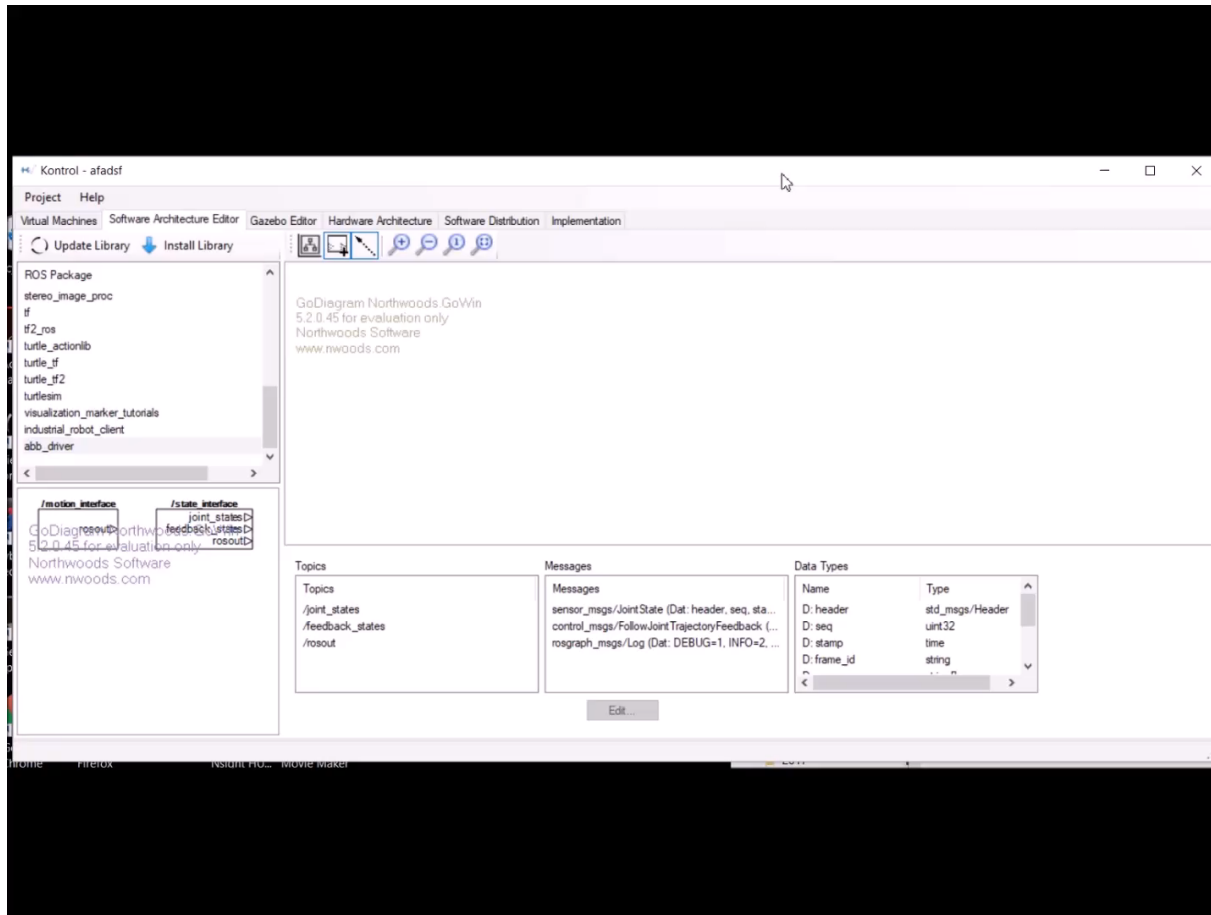
ROS Package

- abb_driver
- camera_calibration
- camera_calibration_parsers
- calculus_1d
- diagnostic_aggregator
- diagnostic_common_diagnostic
- gazebo_ign
- gazebo_ros
- image_proc
- image_proc_plugins
- image_transport
- image_transport_plugins
- image_transport_plugins
- image_transport_plugins
- image_transport_plugins
- image_transport_plugins

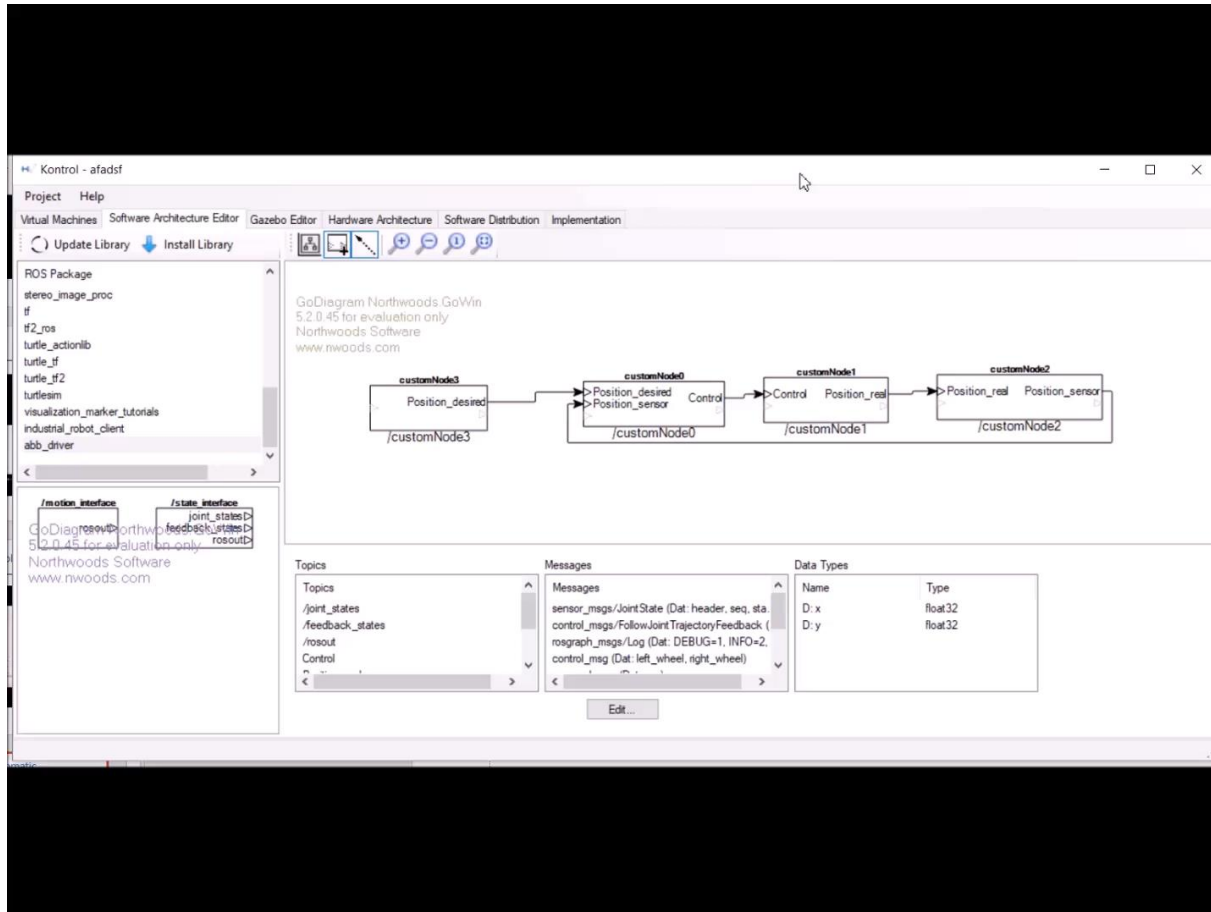
CamStudio
Open Source
Press the Stop Button to stop recording

5:31 PM
9/20/2017

Model-based Design of Controller Structure



Software Distribution and Automatic Configuration

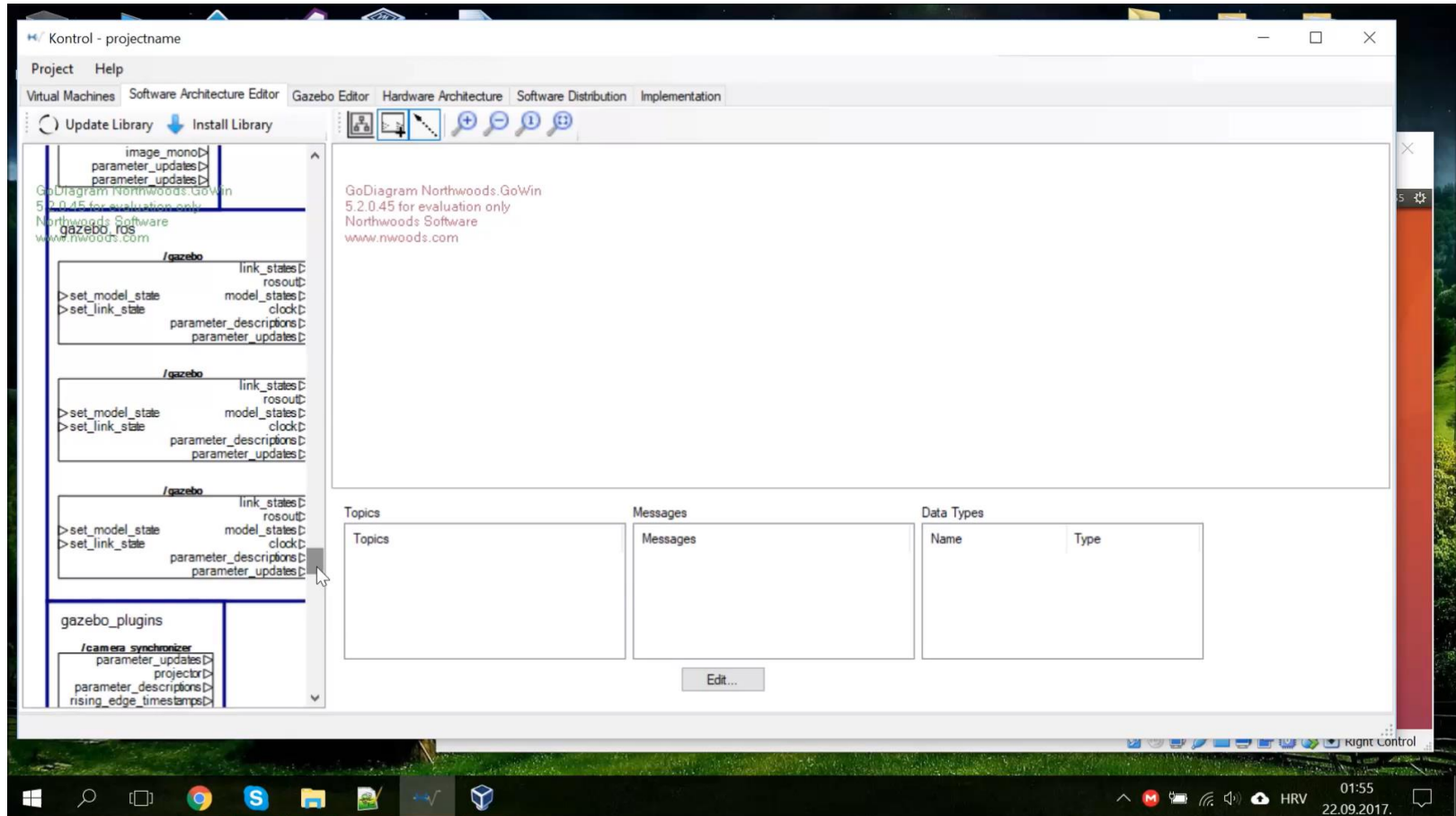


Manual Coding and Model-based design

The screenshot displays a ROS development environment with several windows:

- Terminal:** Shows the execution of a Makefile target `diffrential_drive_generate_messages` and the generation of ROS messages. The output includes the path `~/media/uf_vishared/rl/Scripts` and the command `diffrential_drive_generate_messages`.
- Project Explorer:** Shows the ROS package structure, including the `diffrential_drive` package and its sub-packages: `Position_desired`, `Position_sensor`, `Control`, `Position_real`, and `Position_sensor`.
- Diagram:** A block diagram showing the interconnections between the packages. The `Position_desired` package (CustomNode) sends data to the `Position_sensor` package (CustomNode), which then sends data to the `Control` package (CustomNode). The `Control` package (CustomNode) sends data to the `Position_real` package (CustomNode), which finally sends data to the `Position_sensor` package (CustomNode).
- Messages and Data Types:** A table showing the messages and data types for the packages. The messages are `control_msg` (Data: left_wheel, right_wheel), `pos_real_msg` (Data: x, y), and `pos_sens_msg` (Data: x, y). The data types are `Control` and `Position_sensor`.
- Network:** A network diagram showing the connections between the packages.
- Output:** A terminal window showing the output of the program, including the path `~/media/uf_vishared/rl/Scripts` and the command `diffrential_drive_generate_messages`.

Add and Edit Gazebo World



What we want? / What we are looking for?

- We are looking for Beta Testers
 - Register at kontrol.tech – Beta testing
 - Will start in Q4/2017

Contact



Kontrol GmbH
Dr. Michael Naderhirn
Marienhöhe 25
4391 Waldhausen
Austria
Email: m.naderhirn@kontrol.tech
+43-676-9760483

Kontrol GmbH
Dr. Josef Mendler
Marienhöhe 25
4391 Waldhausen
Austria
Email: j.mendler@kontrol.tech
+49-175-5738846