Table of Contents

- Sonar
- Lidar
- Localization and navigation
- Motor controller
- Joystick
- Social interaction
- Speech
- Speech recognition

Skylab

- Setup Skylab
- Python scripts
- Webserver
- Functions of the webserver
- Skylab servers
- ROS installation on Ubuntu VMs in Skylab
- DNS,DHCP, pfSense & Ubuntu

Radeffect App

- Radeffect App

Lessons learned

- Todo & Advice
- Lessons Learned

Archive

- Previous Groups
- Research Archive

Welcome

Project Willy

- History of Willy
- Project Willy
- Publicity
- Sponsors

Getting started

- Development Guide
- Driving Willy
- Documentation

Build of Willy

- Design history
- Requirements
- Design reference
- Physical build
- Hardware

Robotic Operating System

- Introduction to ROS
- ROS Tutorials
- Multi master

Architecture

- Software Architecture
- Hardware Architecture
- Skylab Architecture
- ROS topic design

Hardware nodes

- sensor node
- si node
- power node
- WillyWRT

Components

- ROS master
- New ROS master on Lubuntu
- Brain
- Sonar
- Lidar
- Localization and navigation
- Motor controller
- Joystick
- Social interaction
- Speech
- Speech recognition

Skylab

- Setup Skylab

1. Software Architecture

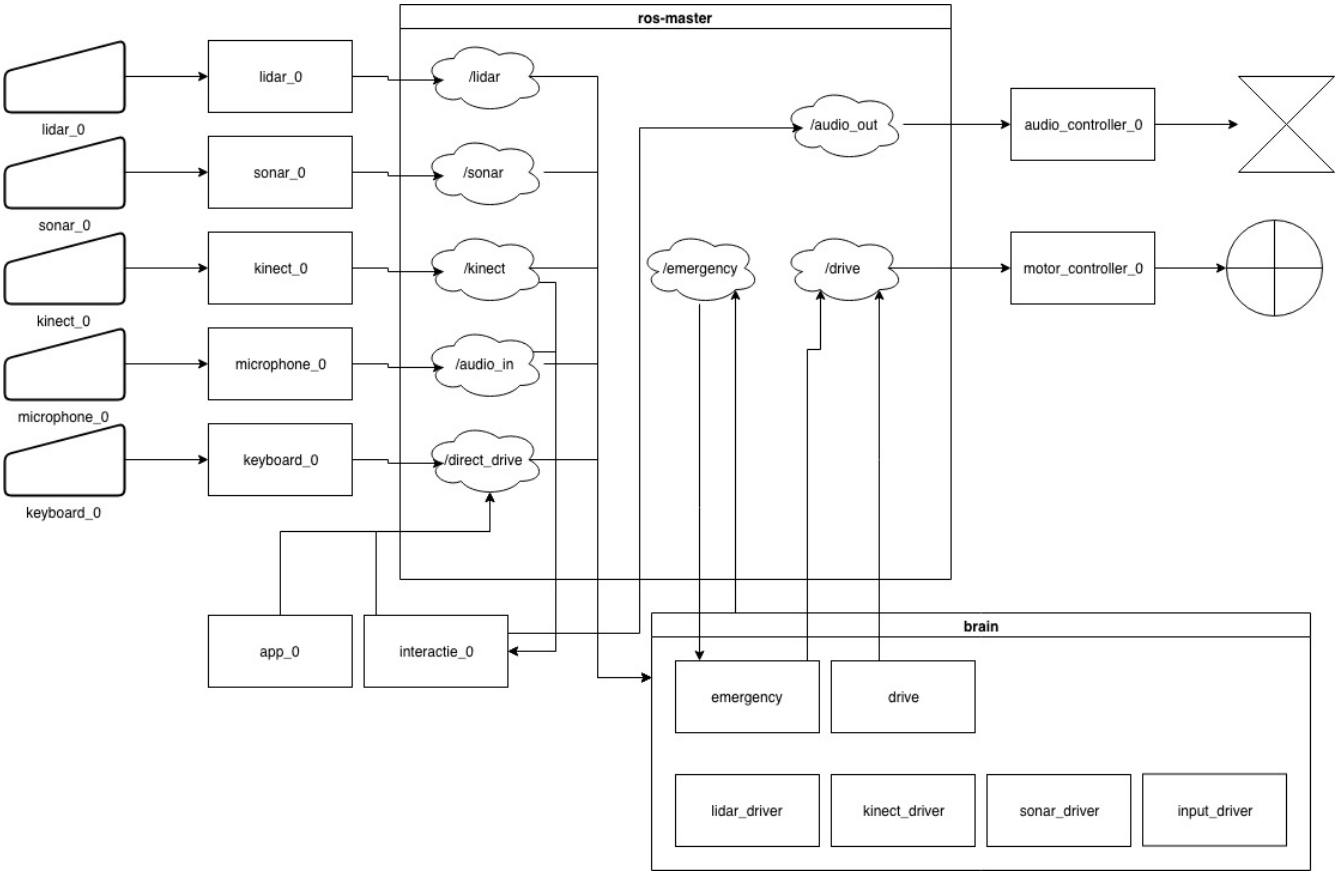
1.1. Standpoints

The software architecture needs to be compartmented for development puproses. To make sure the development doesn't compromise the system as a whole the following standpoints are defined to fortify the design.

- Each seperate function gets a separate repository
- Each seperate function has no dependency to another
- The GiT proces is applied for every repository
- Every function is running hardware independant
- Every function is running OS independant

1.2. Design

Embedding these standpoints into the Robotic Operating System (ROS) resulted in the following architure design.



1.3. ROS Master

A key feature of ROS is the topic communication. By separation of each function the ROS master, which facilitates the topics, is the key component. Every function communicate according a topic and therefore ROS master functions as a servicebus for each current and future feature.

Designing this architecture and keeping it up to date can be done by importing/edditing the xml files in https://www.draw.io/. Draw.io is free of charge to use and requires no client.

Welcome

Project Willy

- History of Willy
- Project Willy
- Publicity
- Sponsors

Getting started

- Development Guide
- Driving Willy
- Documentation

Build of Willy

- Design history
- Requirements
- Design reference
- Physical build
- Hardware

Robotic Operating System

- Introduction to ROS
- ROS Tutorials
- Multi master

Architecture

- Software Architecture
- Hardware Architecture
- Skylab Architecture
- ROS topic design

Hardware nodes

- sensor node
- si node
- power node
- WillyWRT

Components

- ROS master
- New ROS master on Lubuntu
- Brain
- Sonar
- Lidar
- Localization and navigation
- Motor controller
- Joystick
- Social interaction
- Speech
- Speech recognition

Skylab

- Setup Skylab
- Python scripts

2. Hardware Architecture

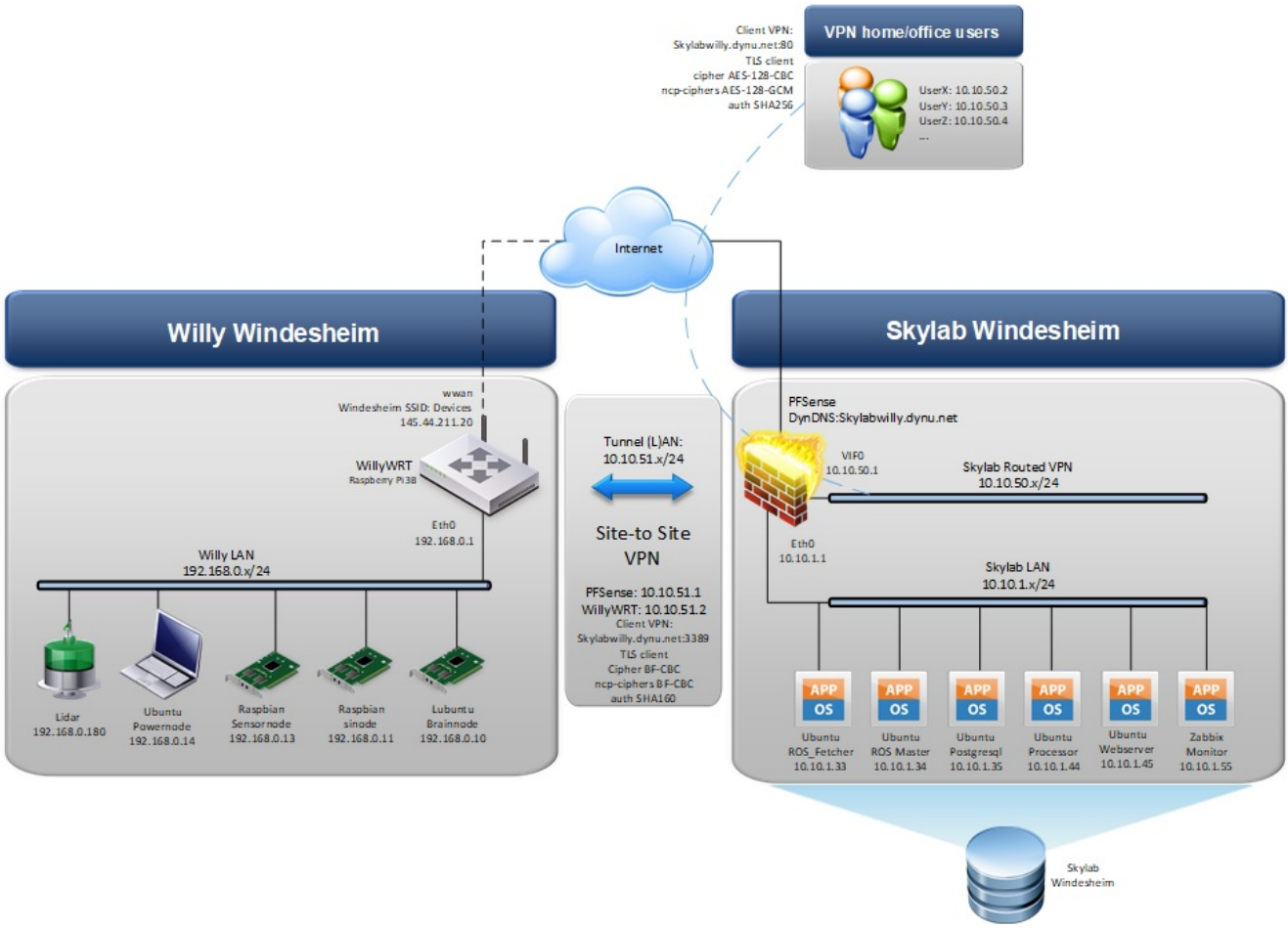
2.1. Standpoints

The hardware architecture is design with a set of standpoints just as the software architecture.

- Use low power hardware if possible to save power consumption
- Clustering similar functions on a separate hardware node
- Using multiple hardware nodes ensured non-disruptive node failure
- Functions that can be ran offsite - Skylab - should
- Variations of OS software is possible as long as topic communication is possible
- Variations of ROS software is possible as long as topic communication is possible
- Simultaneous development is possible
- Skylab communicates safely with Willy

2.2. Design

These standpoints resulted in the following hardware design.



2.3. Usability

By having multiple VPN instances, its possible - if all hardware is powered on - to connect through Skylabs to the individual hardware nodes on Willy even if not physically near him. Another option is to physically connect to WillyLAN and access either Skylab or Willy hardware nodes.

Designing this architecture and keeping it up to date can be done by editing the vsdx files in Microsoft Visio.

Welcome

Project Willy

- History of Willy
- Project Willy
- Publicity
- Sponsors

Getting started

- Development Guide
- Driving Willy
- Documentation

Build of Willy

- Design history
- Requirements
- Design reference
- Physical build
- Hardware

Robotic Operating System

- Introduction to ROS
- ROS Tutorials
- Multi master

Architecture

- Software Architecture
- Hardware Architecture
- Skylab Architecture
- ROS topic design

Hardware nodes

- sensor node
- si node
- power node
- WillyWRT

Components

- ROS master
- New ROS master on Lubuntu
- Brain
- Sonar
- Lidar
- Localization and navigation
- Motor controller
- Joystick
- Social interaction
- Speech
- Speech recognition

Skylab

- Setup Skylab

3. Skylab Architecture

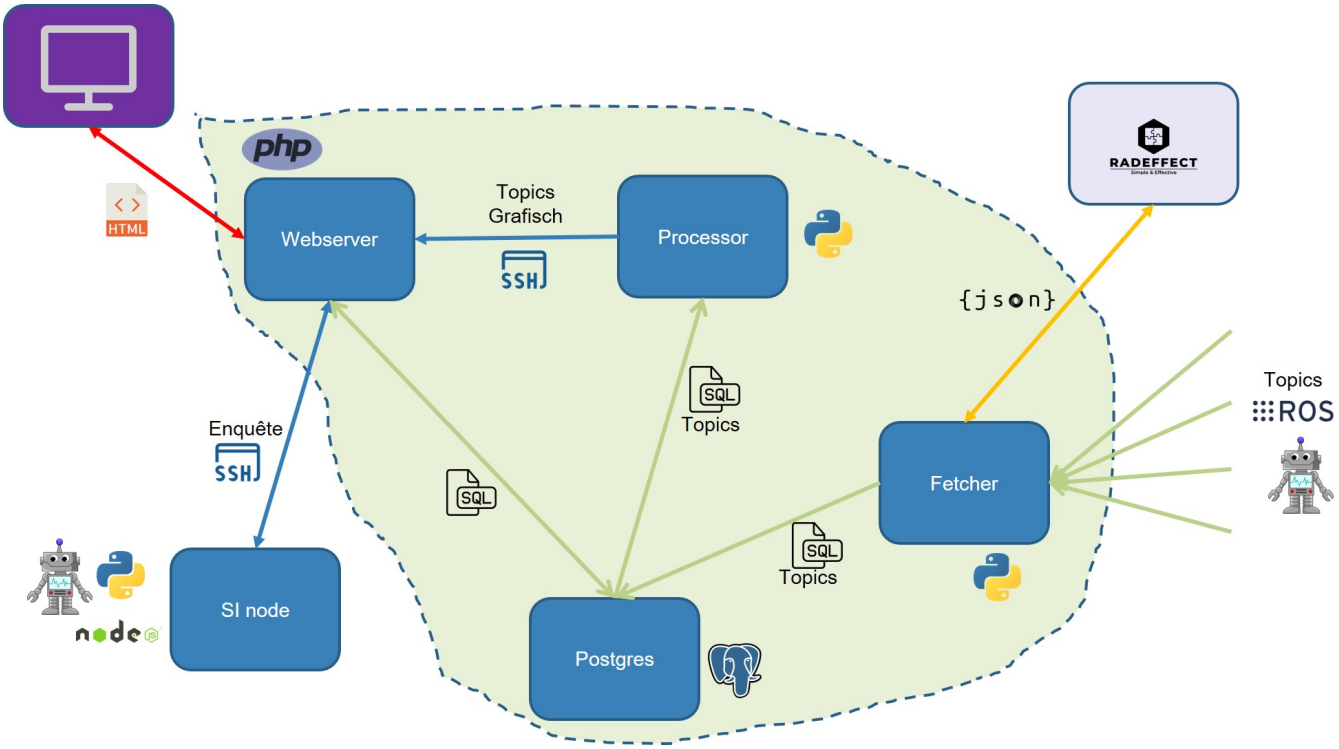
3.1. Standpoints

The Skylab architecture is design with a set of standpoints just as the hardware and software architecture.

- Skylab can give limited resources to a node, therefore modulairity is essential. All solutions in Skylab should be built with scale out possibility in mind.
- Skylab is the place for all not vital functions of Willy.
- Skylab is seen as a redundant environment with high enough up-time. There is no need for dual datacenter architecture.
- Variations of OS software is possible as long as topic communication is possible.
- Variations of ROS software is possible as long as topic communication is possible.
- Simultaneous development is possible.
- Skylab communicates safely with Willy.
- Skylab has limited port connection to the Internet, fi mail anf FTP is is not possible.
- Skylab supplies for ROS Topic fetcher, PostgreSQL and Webserver services. Additional nodes are used for all sorts of processing like Data Sciense and RADeffect connectivity.

3.2. Design

These standpoints resulted in the following hardware design.



3.3. Usability

By having multiple VPN instances, its possible - if all hardware is powered on - to connect through Skylabs to the individual hardware nodes on Willy even if not physically near him. Another option is to physically connect to WillyLAN and access either Skylab or Willy hardware nodes.

Designing this architecture and keeping it up to date can be done by editing the pptx file in Microsoft PowerPoint on the SharePoint site of Willy.

- ROS installation on Ubuntu VMs in Skylab
- DNS,DHCP, pfSense & Ubuntu

Radeffect App

- Radeffect App

Lessons learned

- Todo & Advice
- Lessons Learned

Archive

- Previous Groups
- Research Archive

4. ROS topics

This are the topics at the ROS master of Willy. There are several communication protocols defined in ROS, but in Willy only topics are used. The topics are shown in the next table.

Topic name	Purpose	Publisher	Subscriber	Message type	Message protocol
/willy/health	Is Willy healthy or is there an issue?	All nodes	All nodes	Int32	• 1 = healthy • 2 = non-critical issue • 3 = critical issue
/will/health_reason	If Willy is not healthy, the reason for that is published in clear text here	All nodes	All nodes	String	Clear text
/willy/activity	What is Willy doing?	Brain SI	All nodes	Int32	• 1 = cruising • 2 = conversation • 3 = enquête • 4 = driving with a defined target

Topic name	Purpose	Publisher	Subscriber	Message type	Message protocol
/interaction/is_active	If the social interaction is busy?	Social interaction speech recognition	All nodes	Int32	• 0 = not active • 1 = active
/interaction/clear_text	The unprocessed text what the person said	Social interaction speech recognition	All nodes	String	Who are you?
/move_action	Move actions for willy	Navigation brain	SI	String	• turn_around = Turn Willy around
/human_detect	Is there a human in front of Willy?	Power node	Brain	String	• 1 = id • 2 = accuracy • 3 = distance