A USE CASE IN MODEL-BASED ROBOT DEVELOPMENT USING AADL AND ROS

POLITECNICO
MILANO 1863

Gianluca Bardaro

Andrea Semprebon

Matteo Matteucci

THE ROBOTIC PLATFORM

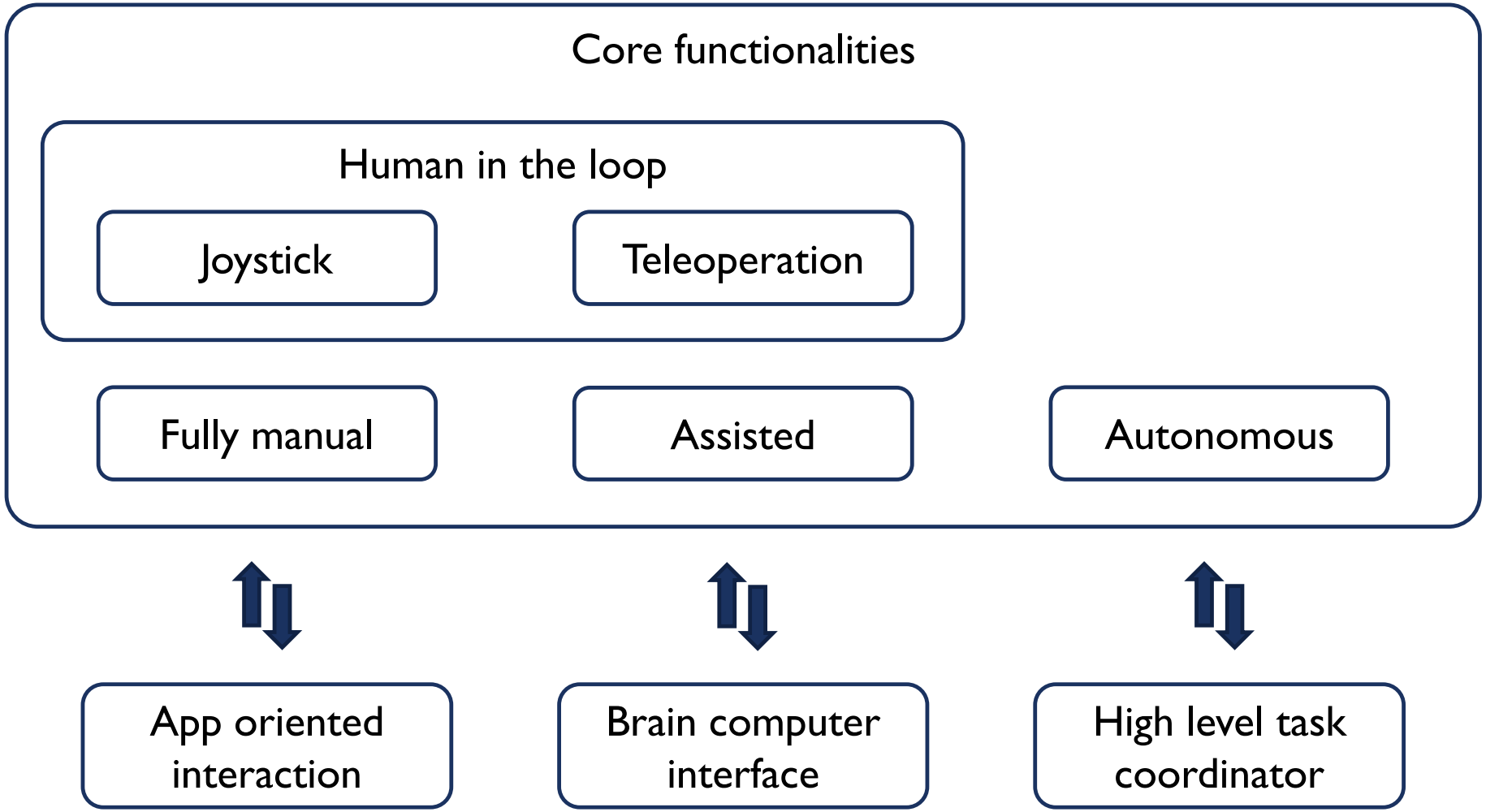
On board computer

Encoders

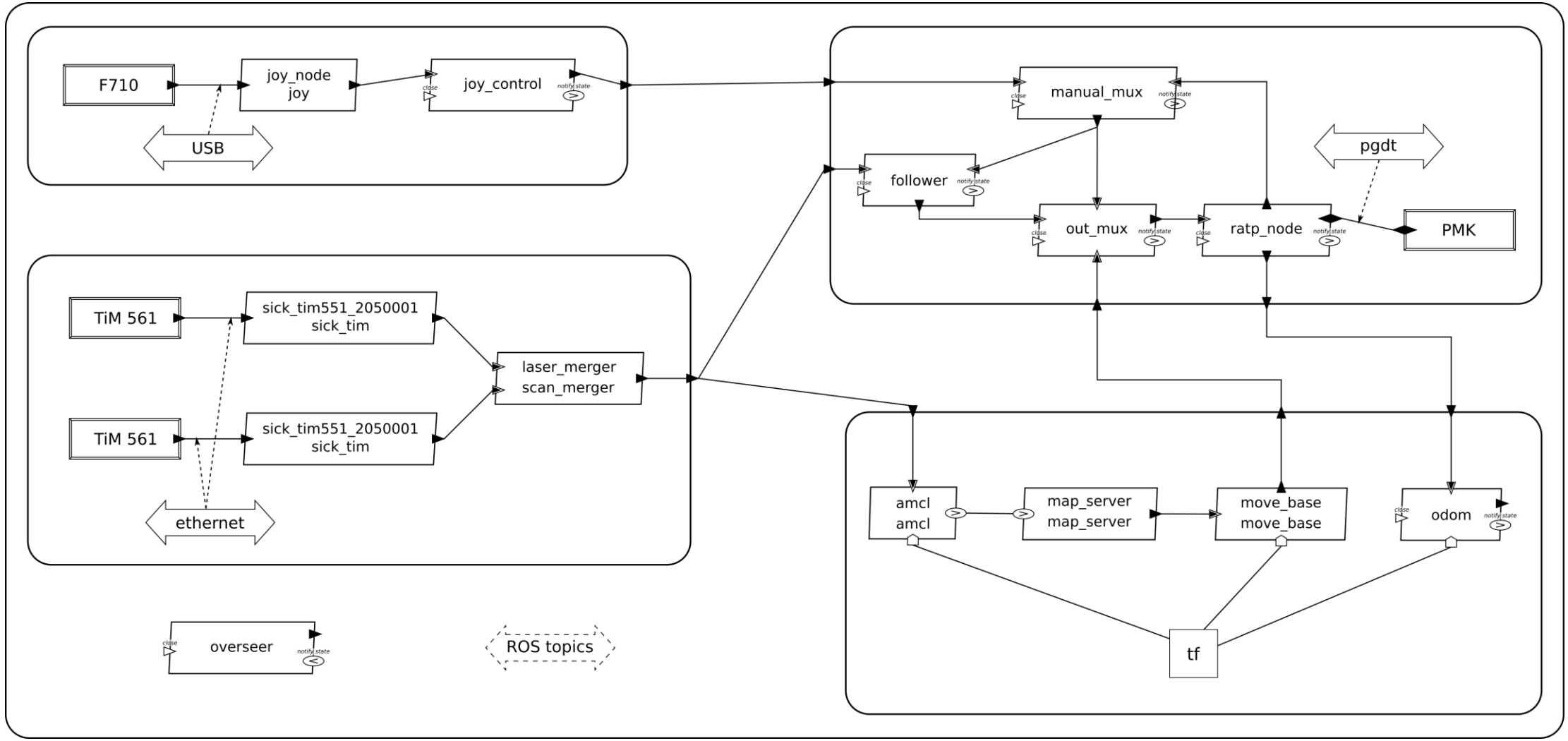
Joystick

Lasers

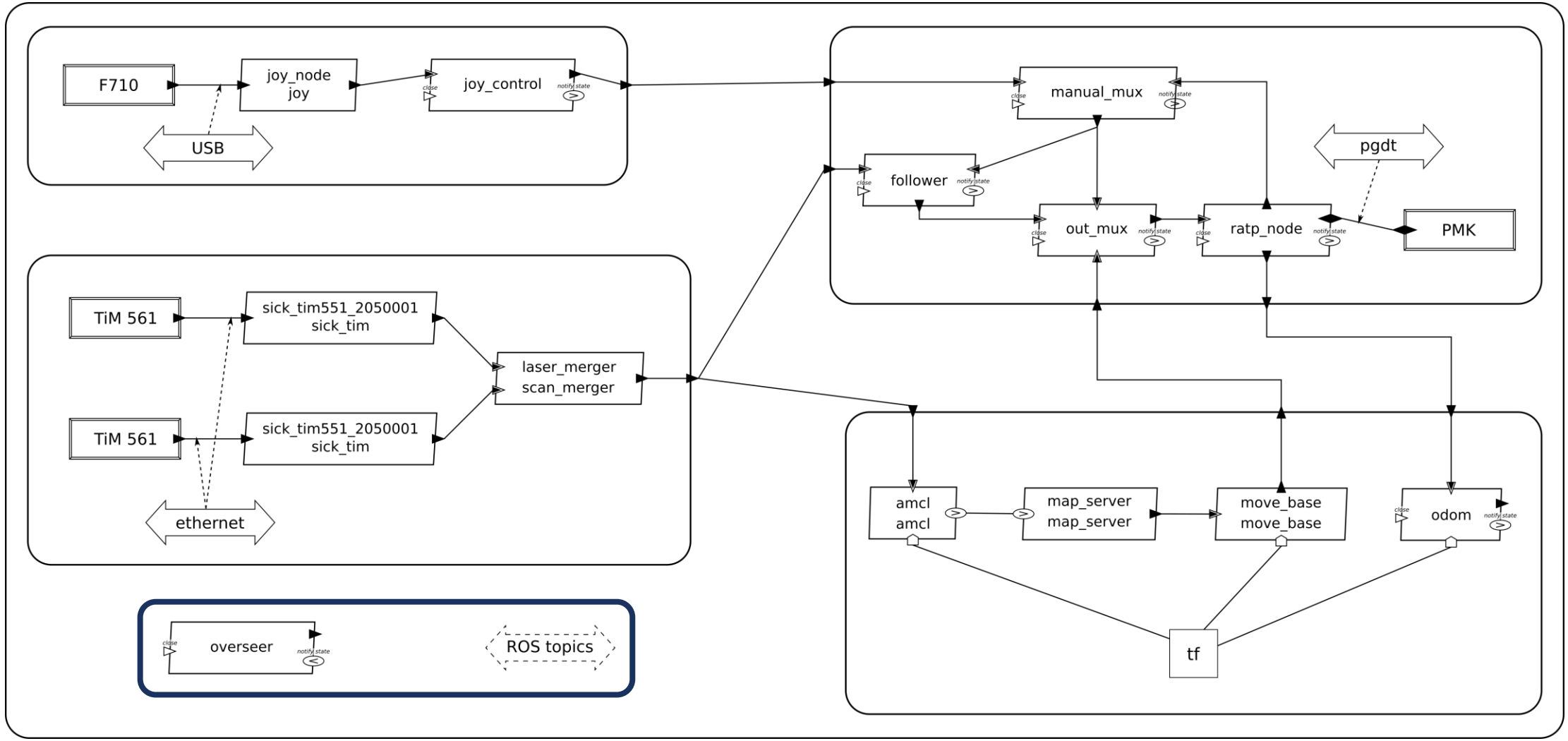
APPLICATION



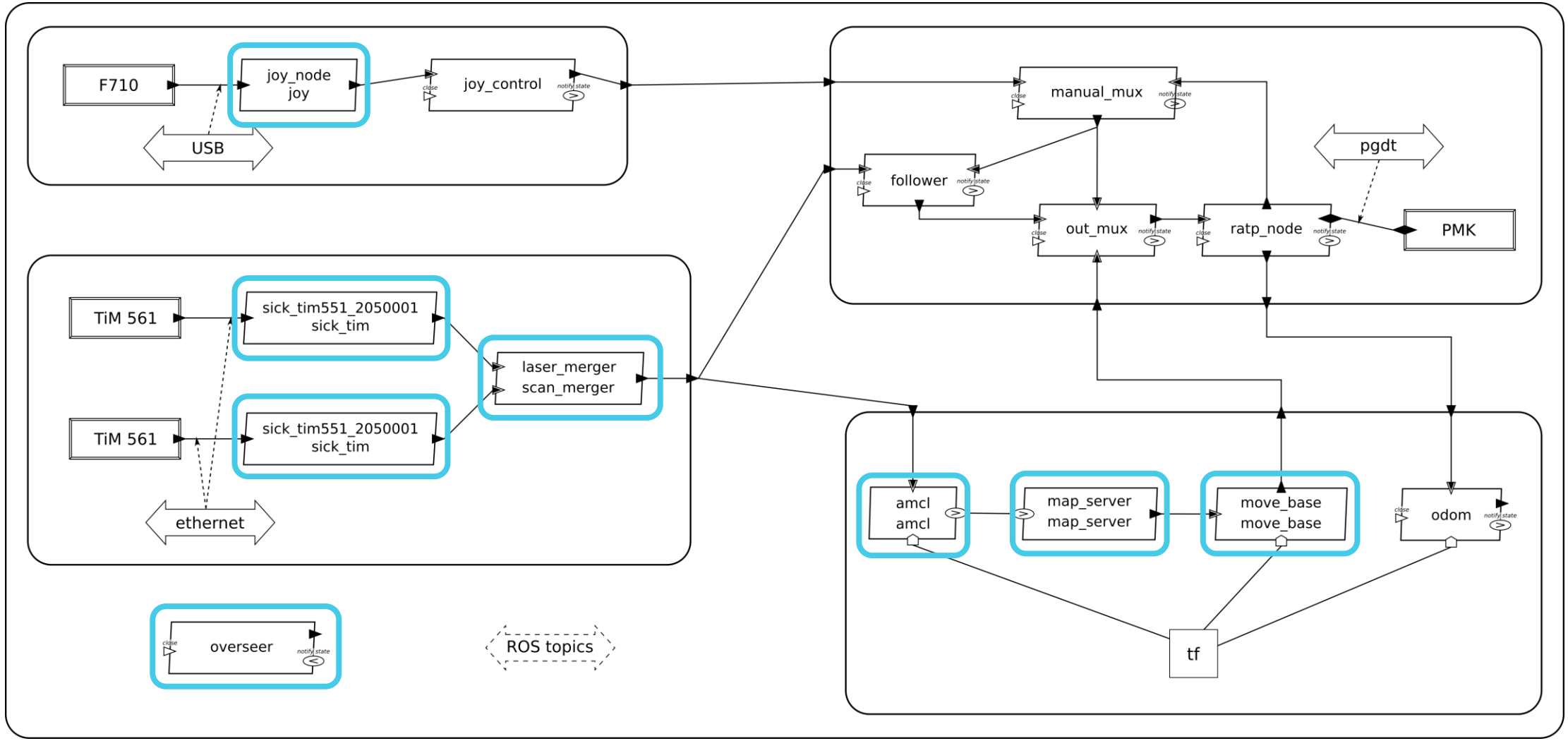
THE MODEL


F710
joy_node
joy
joy_control
close
notify state
USB
manual_mux
follower
pgdt
out_mux
ratp_node
PMK
TiM 561
sick_tim551_2050001
sick_tim
laser_merger
scan_merger
TiM 561
sick_tim551_2050001
sick_tim
ethernet
amcl
amcl
map_server
map_server
move_base
move_base
odom
tf
overseer
ROS topics

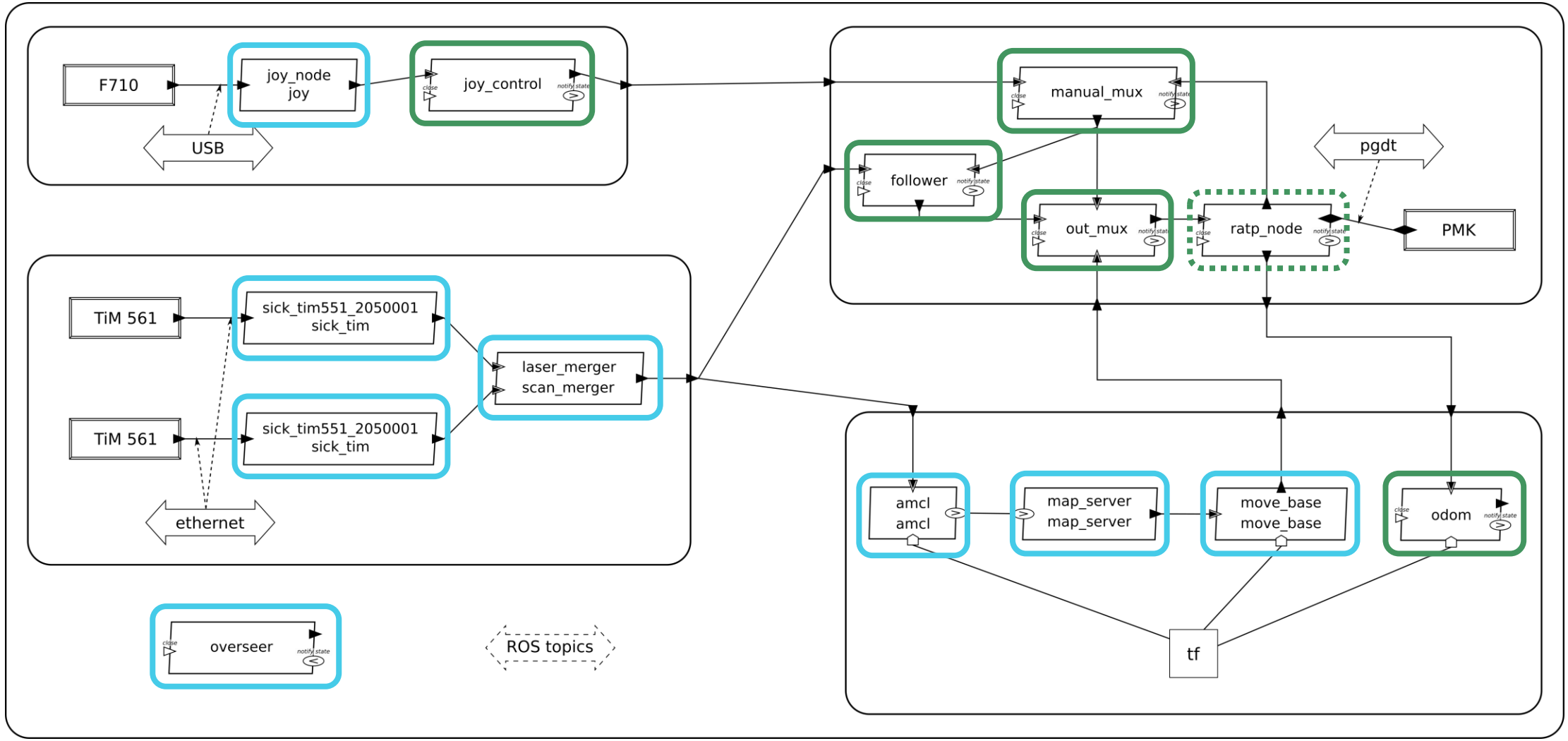
THE MODEL


F710
joy_node
joy
USB
joy_control
close
notify state
manual_mux
follower
out_mux
ratp_node
pgdt
PMK
TiM 561
sick_tim551_2050001
sick_tim
TiM 561
sick_tim551_2050001
sick_tim
laser_merger
scan_merger
ethernet
amcl
amcl
map_server
map_server
move_base
move_base
odom
tf
overseer
ROS topics

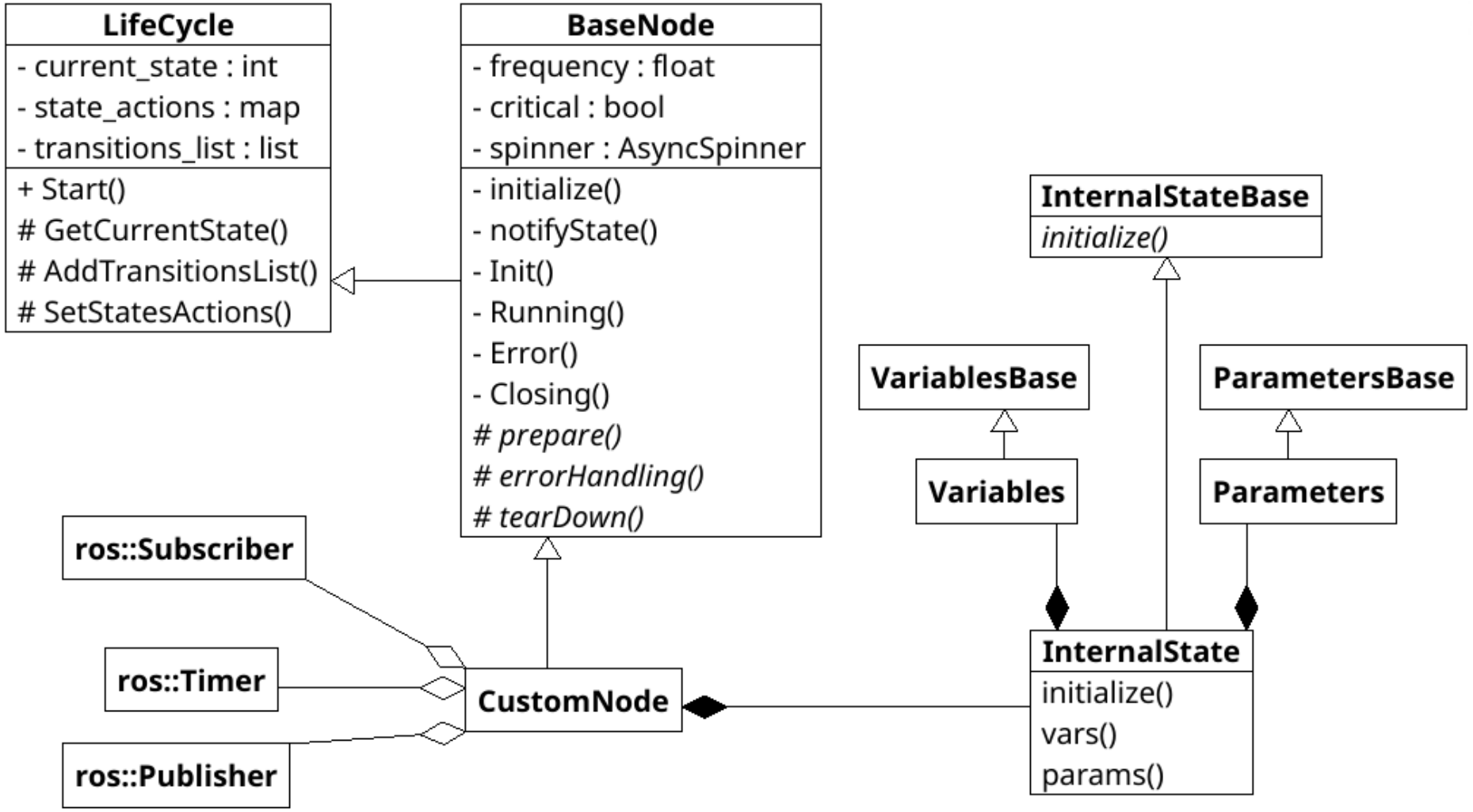
THE MODEL


F710
joy_node
joy
USB
joy_control
close
notify state
manual_mux
follower
out_mux
ratp_node
pgdt
PMK
TiM 561
sick_tim551_2050001
sick_tim
TiM 561
sick_tim551_2050001
sick_tim
laser_merger
scan_merger
ethernet
amcl
amcl
map_server
map_server
move_base
move_base
odom
tf
overseer
ROS topics

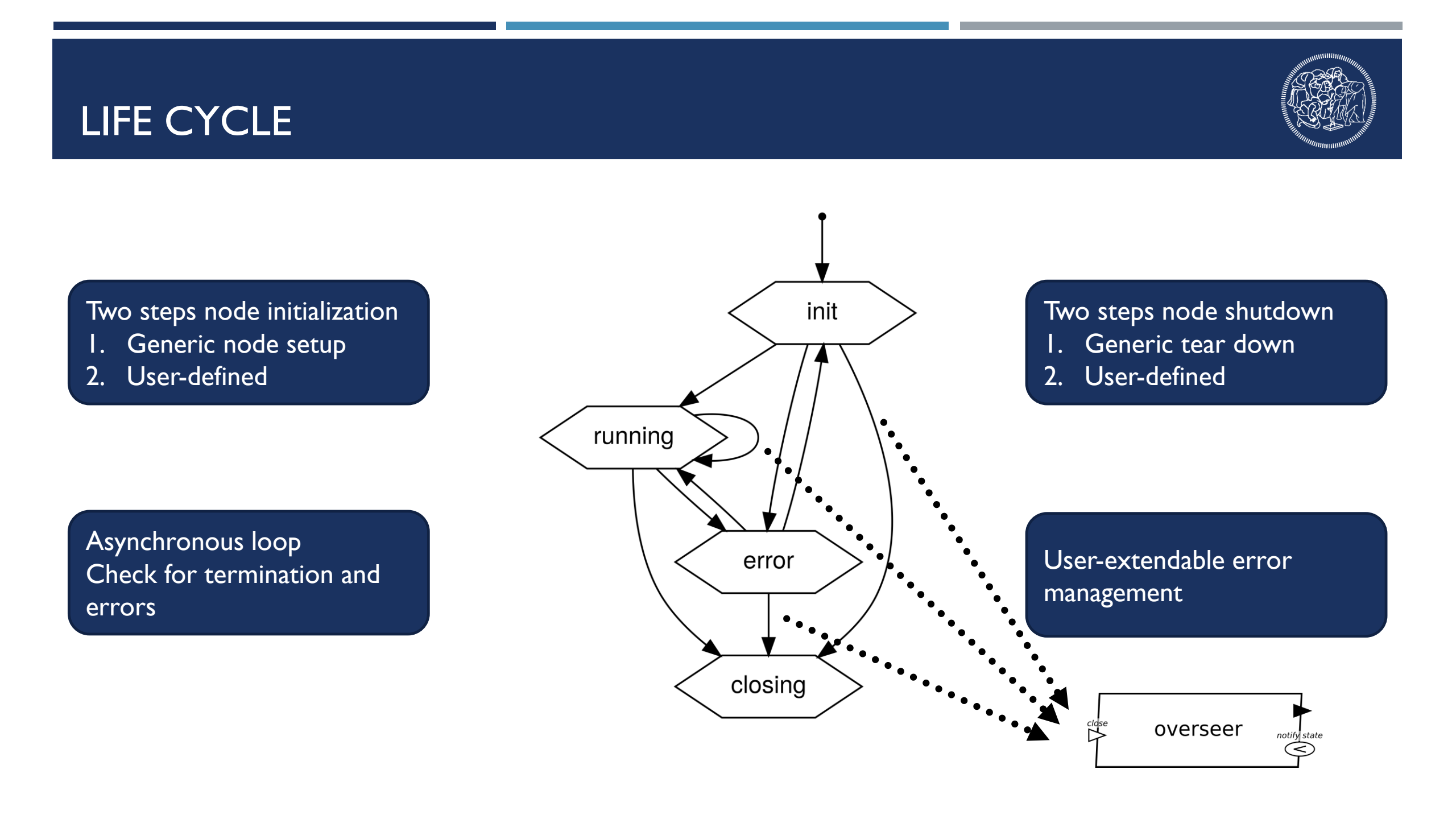
THE MODEL


F710
joy_node
joy
USB
joy_control
close
notify state
manual_mux
follower
out_mux
ratp_node
pgdt
PMK
TiM 561
sick_tim551_2050001
sick_tim
TiM 561
sick_tim551_2050001
sick_tim
laser_merger
scan_merger
ethernet
amcl
amcl
map_server
map_server
move_base
move_base
odom
tf
overseer
ROS topics

BASE NODE STRUCTURE



LIFE CYCLE



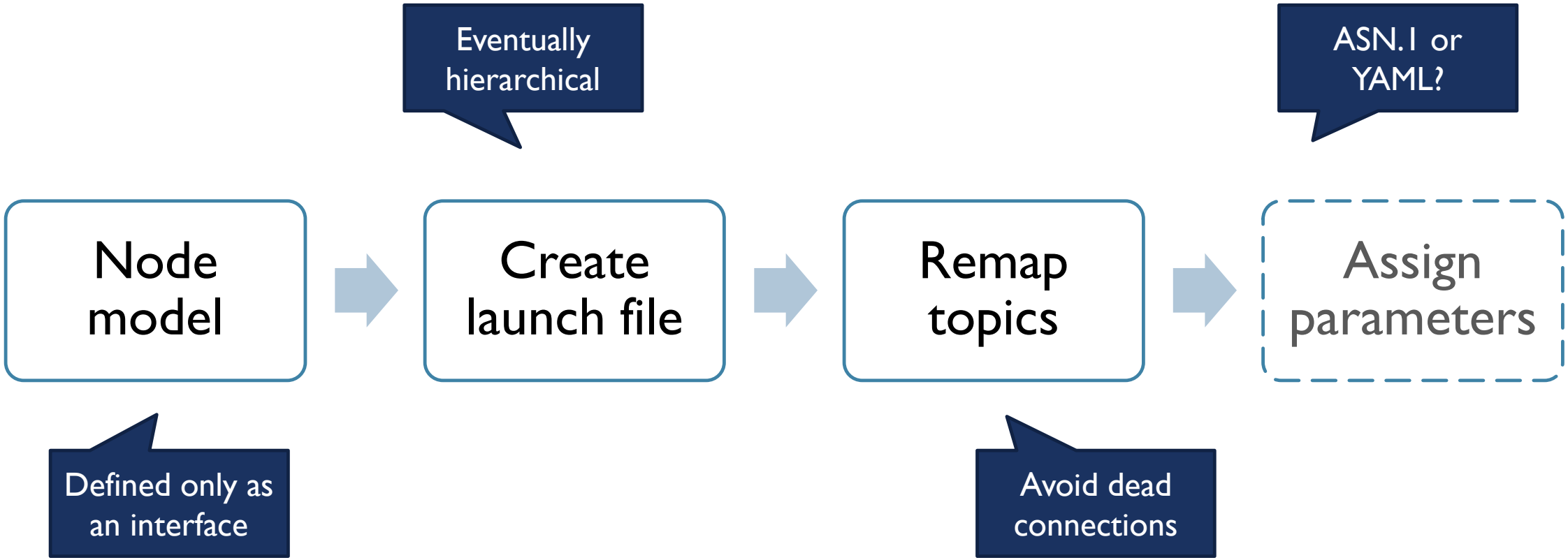
Two steps node initialization
1. Generic node setup
2. User-defined

Asynchronous loop
Check for termination and errors

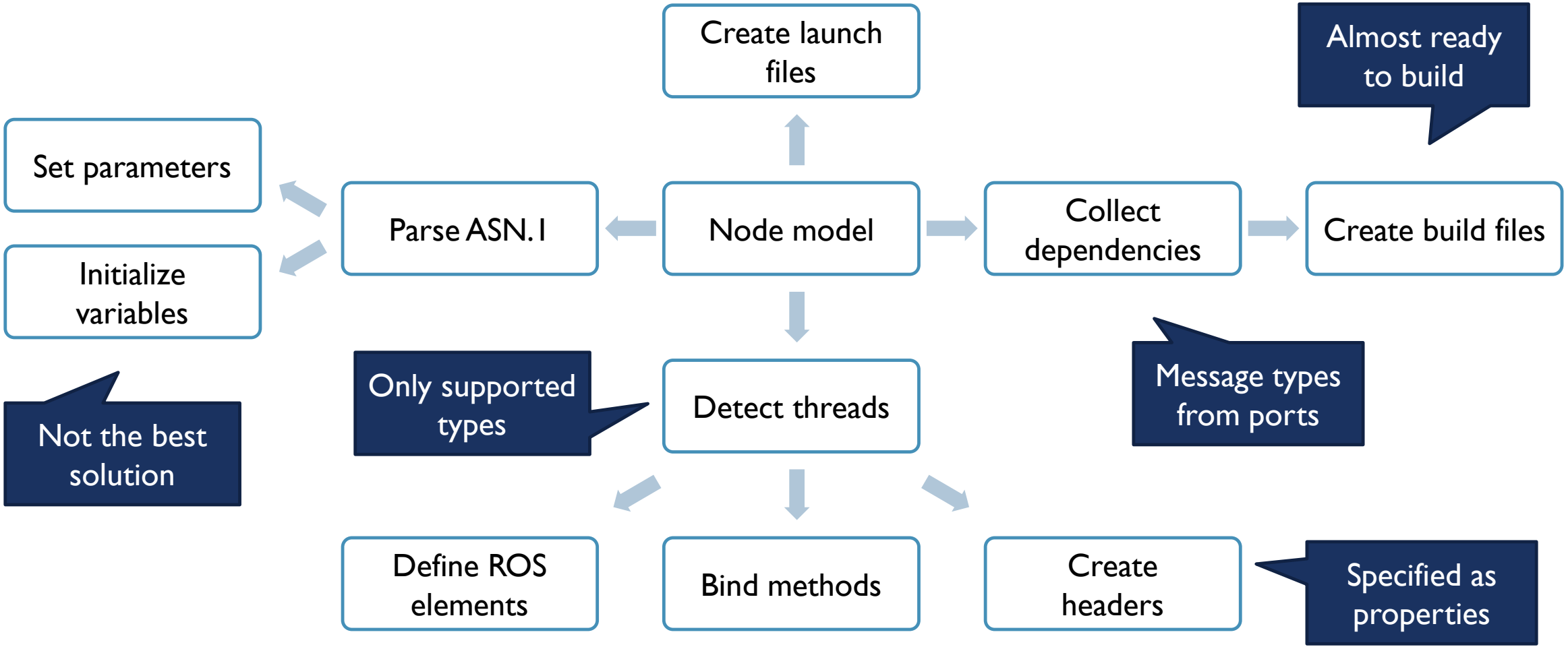
Two steps node shutdown
1. Generic tear down
2. User-defined

User-extendable error management

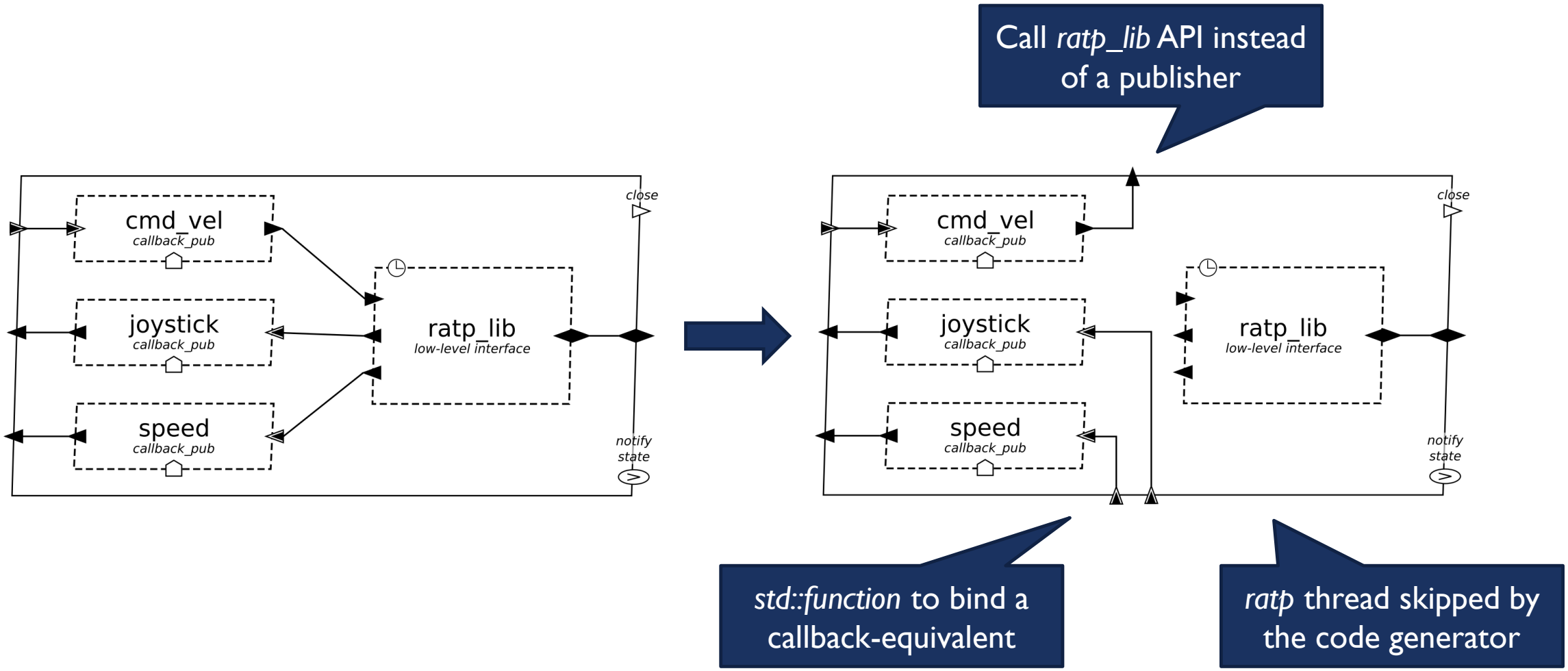
CODE GENERATION: EXISTING ROS NODES



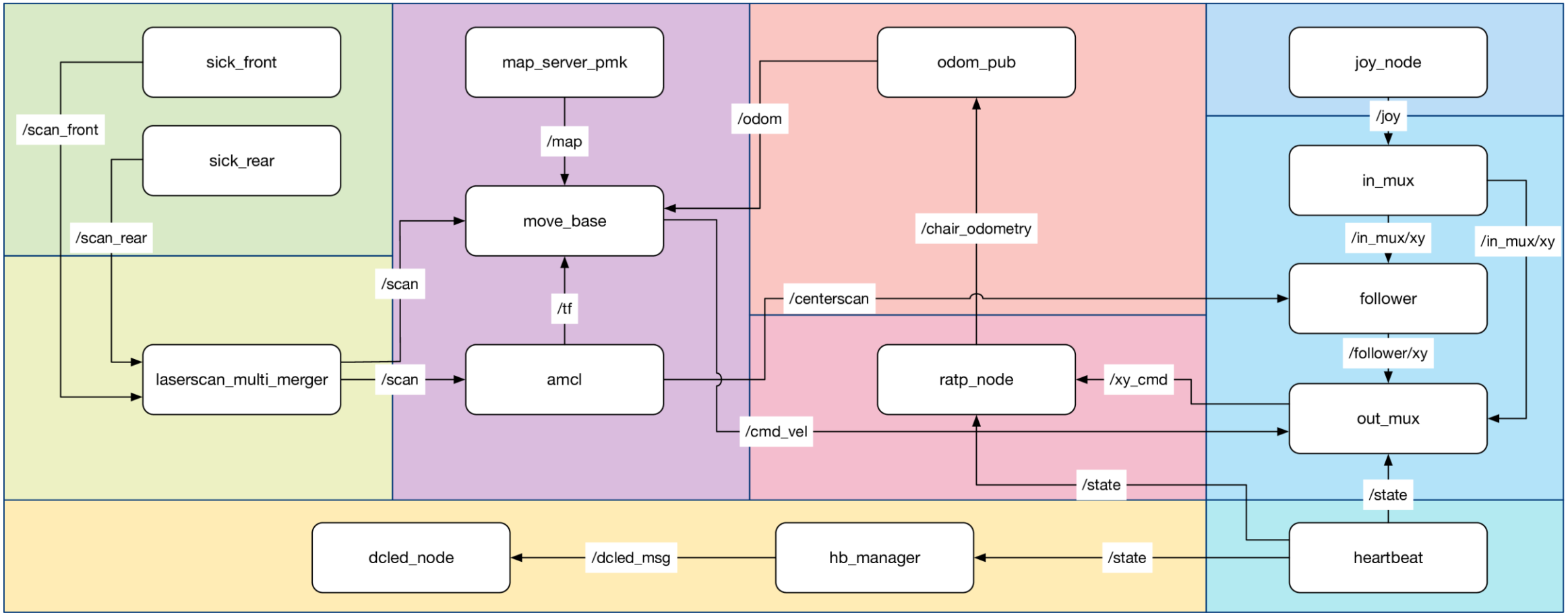
CODE GENERATION: CUSTOM NODES



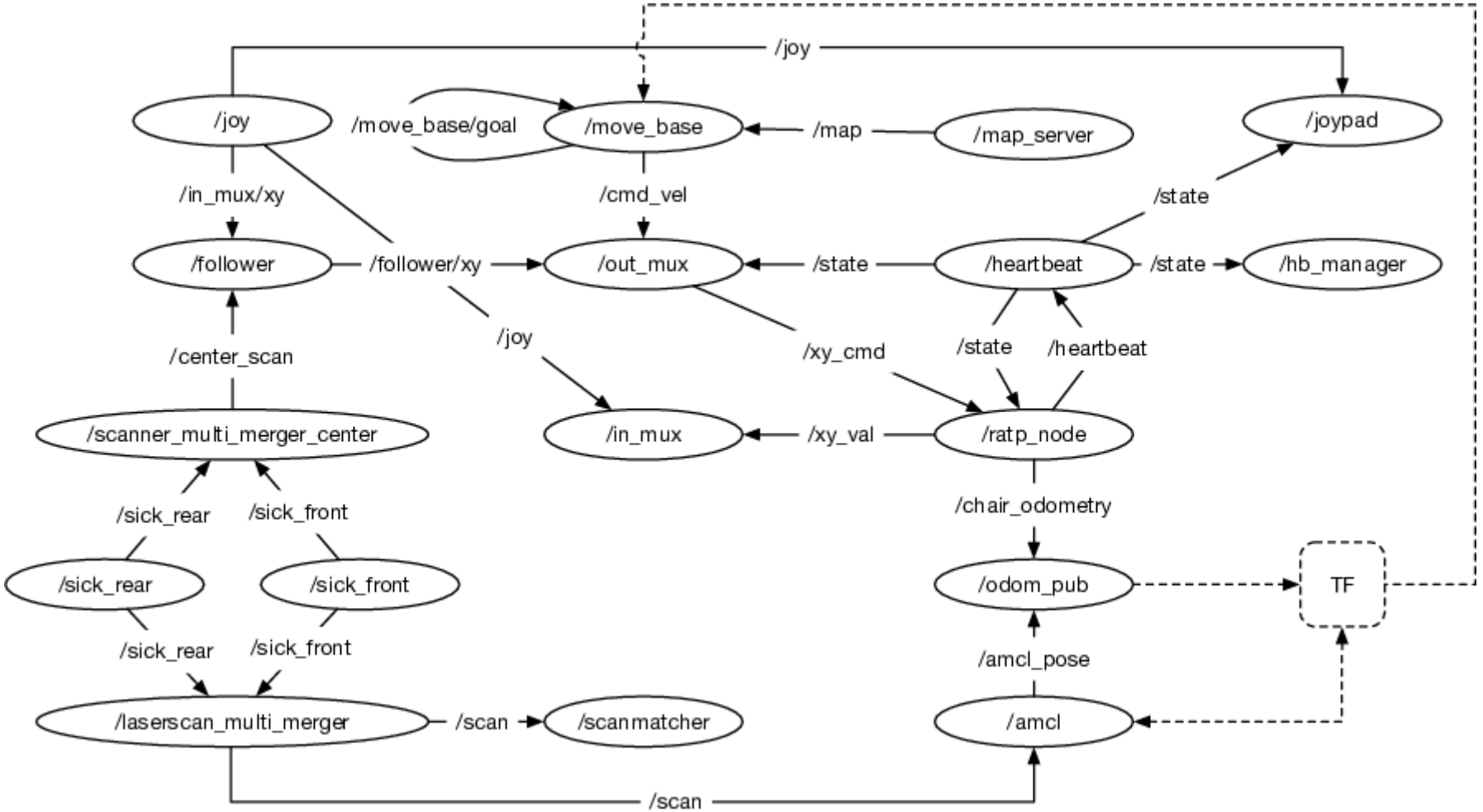
RATP NODE


Call *ratp_lib* API instead of a publisher
cmd_vel
callback_pub
joystick
callback_pub
speed
callback_pub
ratp_lib
low-level interface
close
notify state
cmd_vel
callback_pub
joystick
callback_pub
speed
callback_pub
ratp_lib
low-level interface
close
notify state
std::function to bind a callback-equivalent
ratp thread skipped by the code generator

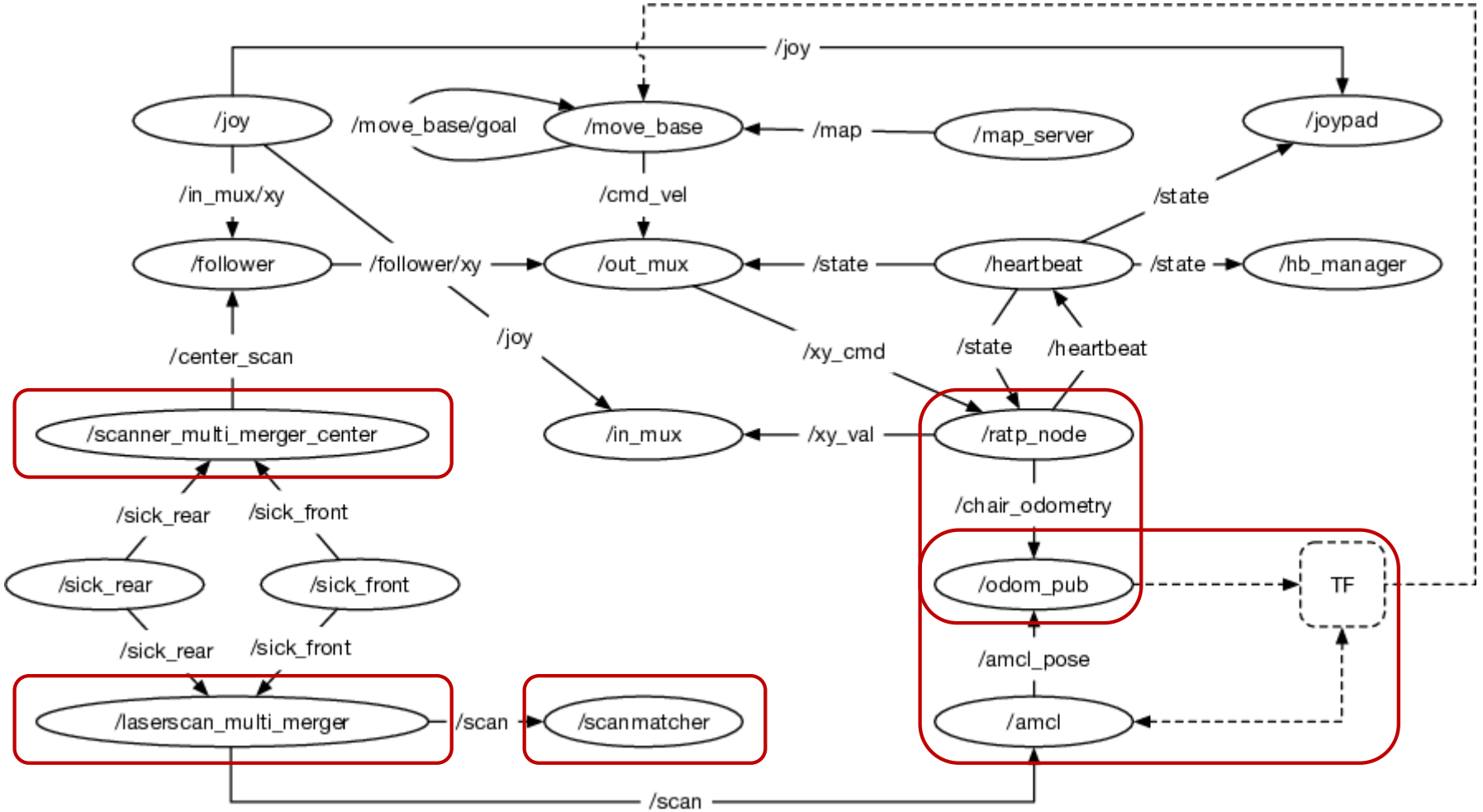
ARCHITECTURE COMPARISON



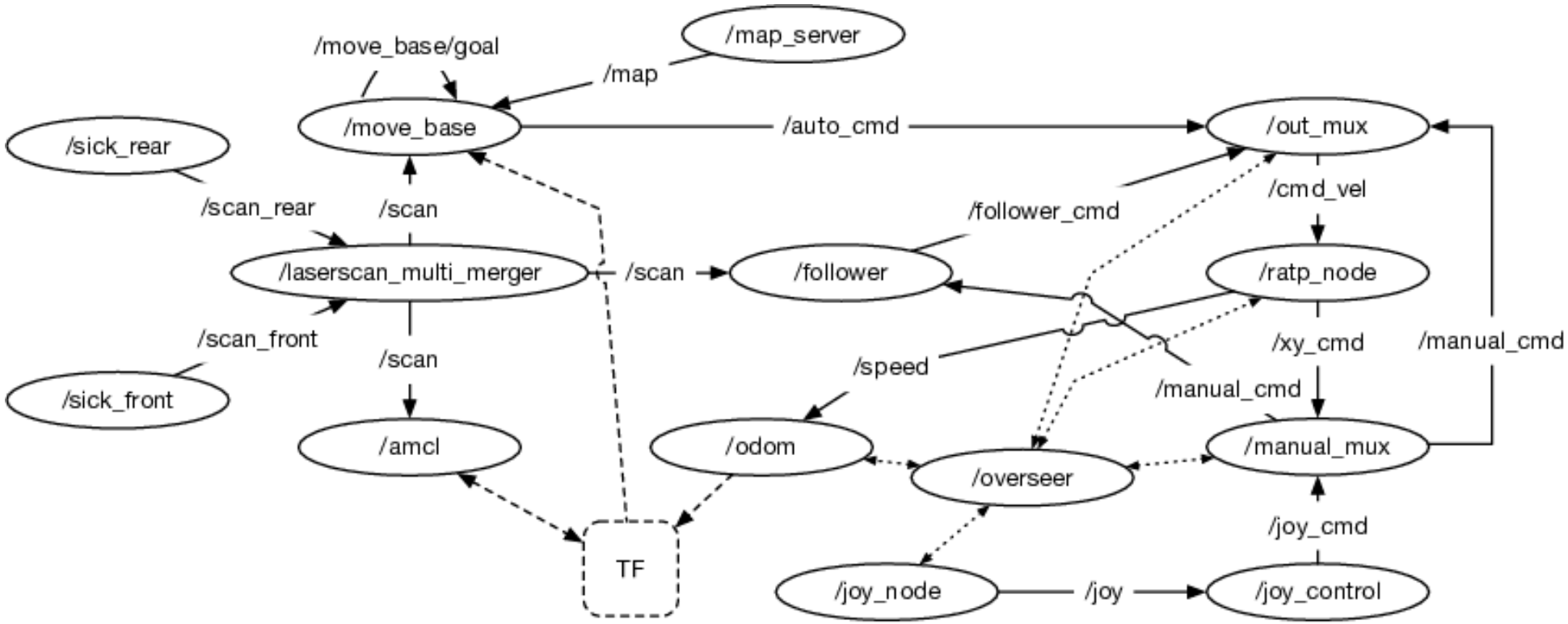
ARCHITECTURE COMPARISON



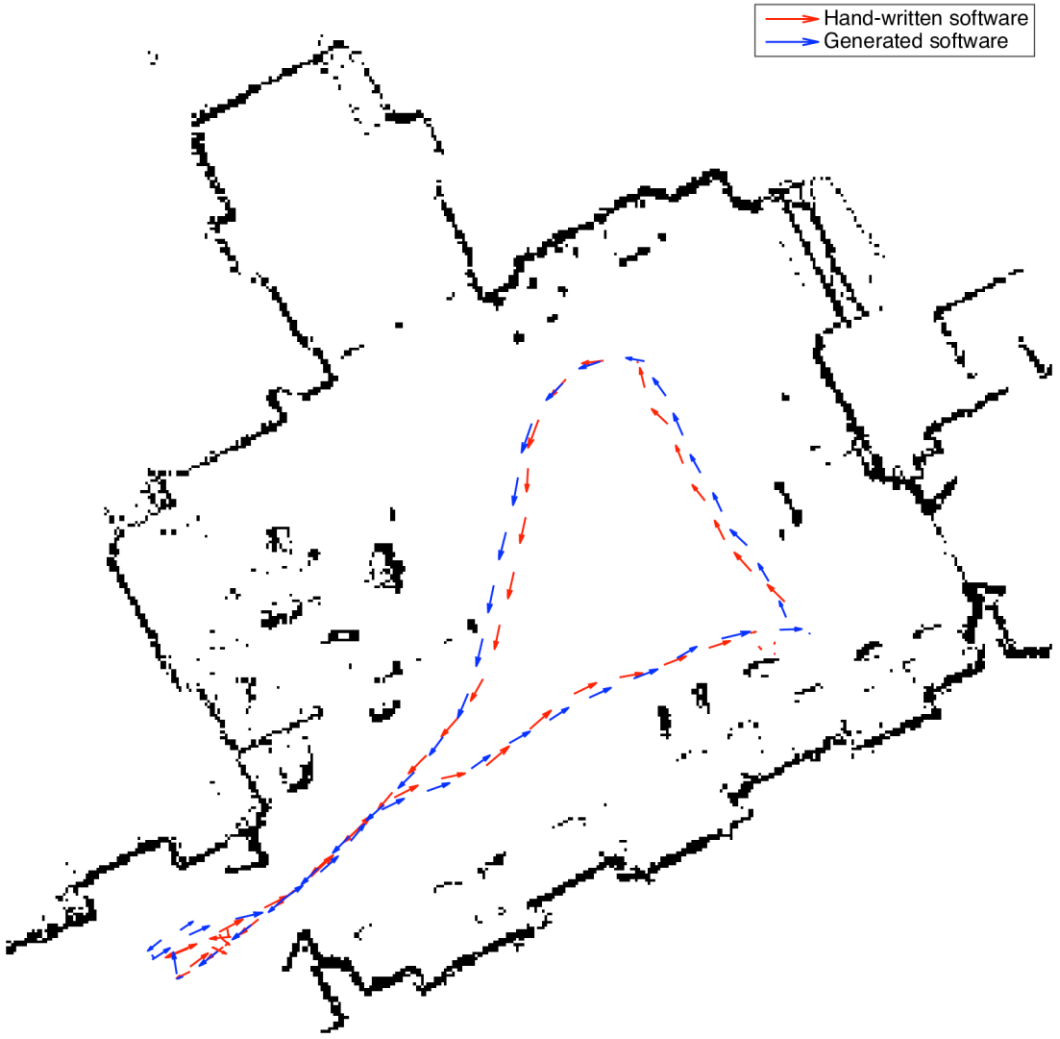
ARCHITECTURE COMPARISON



ARCHITECTURE COMPARISON



AT RUNTIME



CONCLUSIONS

This work is the answer to a practical problem:

"Simplify the development of our mobile platforms"

Getting there

"Push re-usability from components to designs"

Working on it

What we learnt:

It is about consistency and precision

"Modeling is (relatively) easy"

"Code generation is (definitely) hard"

Requires a deep vertical expertise

THANK YOU