



DE LA RECHERCHE À L'INDUSTRIE

# Developing ROS2 systems with Papyrus for Robotics

SHARC Day, GDR Robotique 09/07/2021

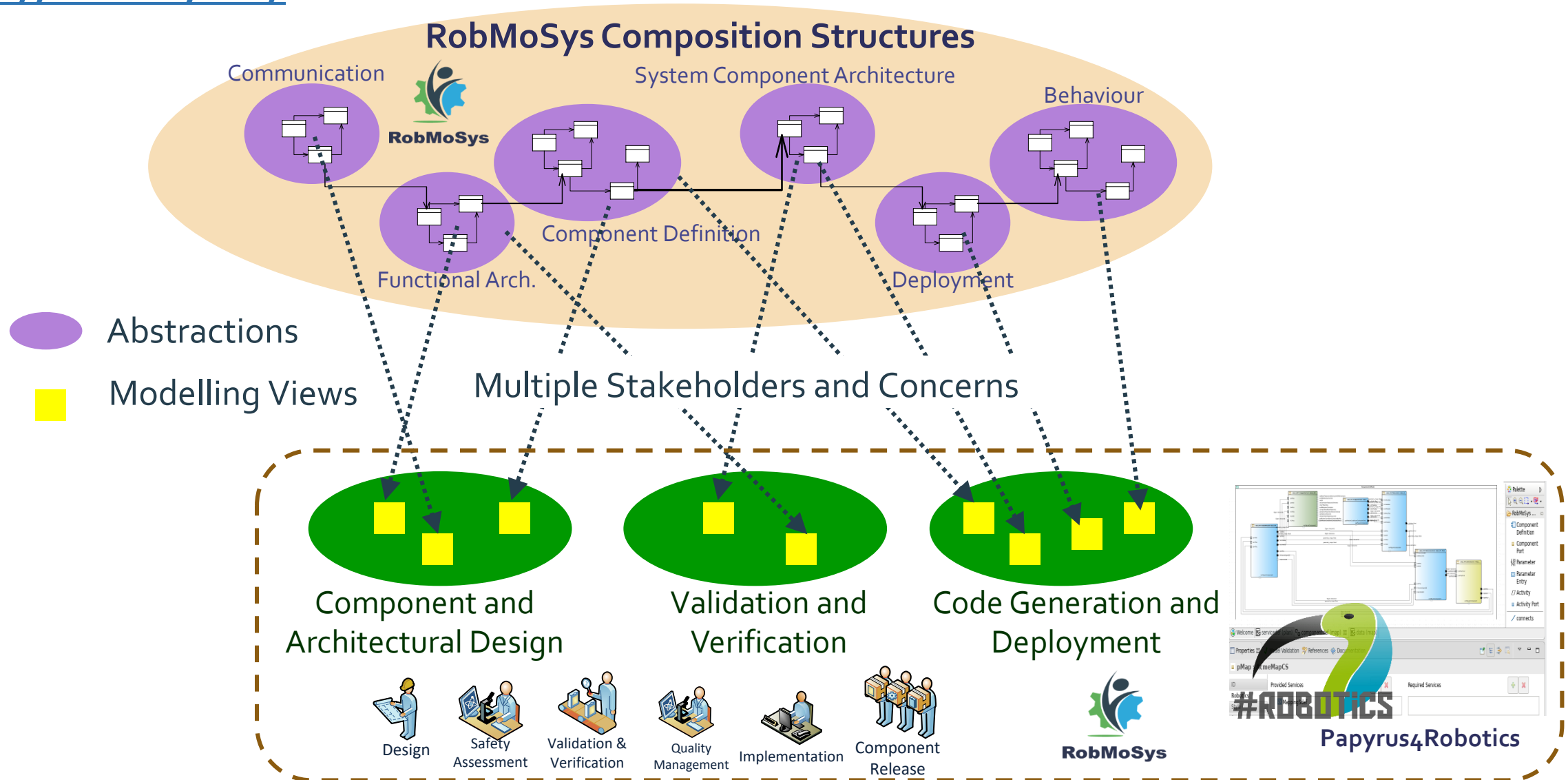
Matteo MORELLI, Ansgar RADERMACHER CEA-LIST/DILS/LSEA

**Robotics software engineering** is still in a “**craft**” **stage** compared to software engineering in more advanced domains such as **automotive** and **avionics**

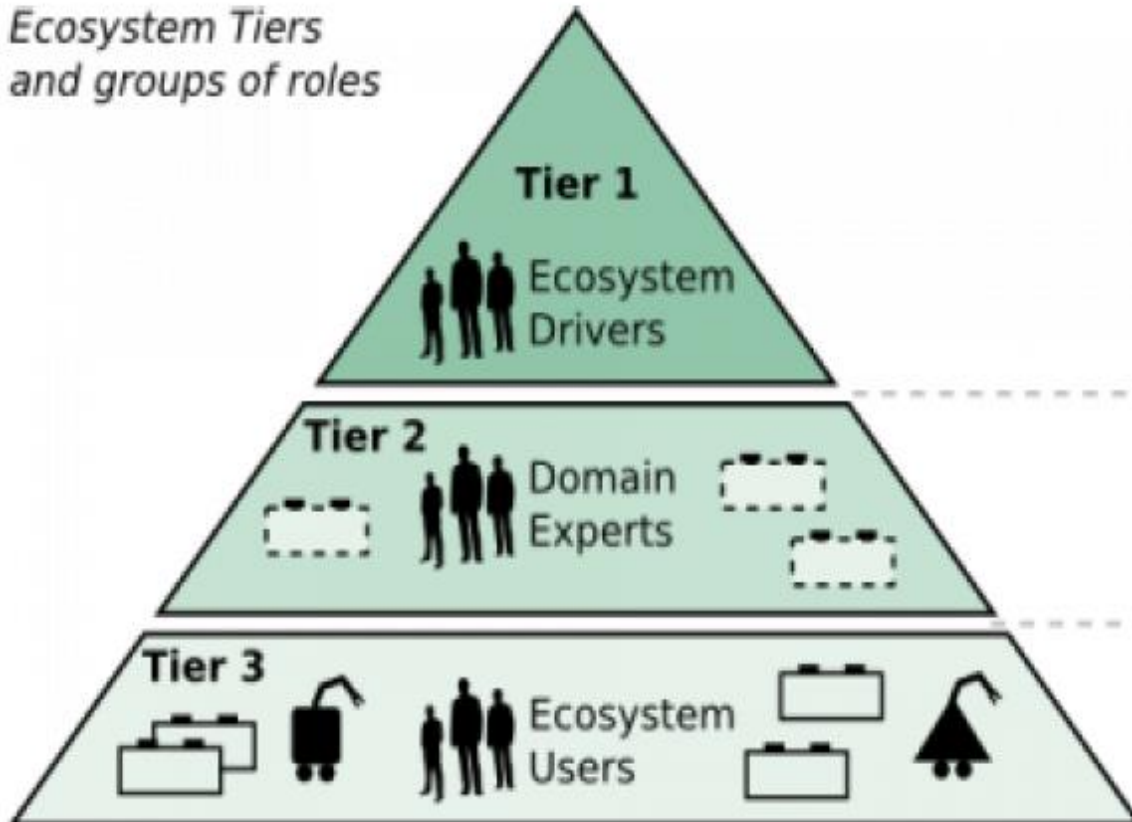
- ▶ Numerous and often incompatible component-based frameworks
- ▶ Code-centric software development approach
- ▶ No standard interfaces between roles involved in developing robotics systems
- ▶ Participants involved with heterogenous backgrounds and skills



<https://robmosys.eu/>



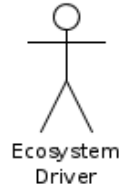
## Ecosystem Tiers and groups of roles



Roles on Composition Tiers:

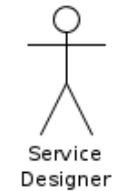
**Tier1**  
Defines the composition-structure, structures the ecosystem.

This is e.g. the RobMoSys consortium.



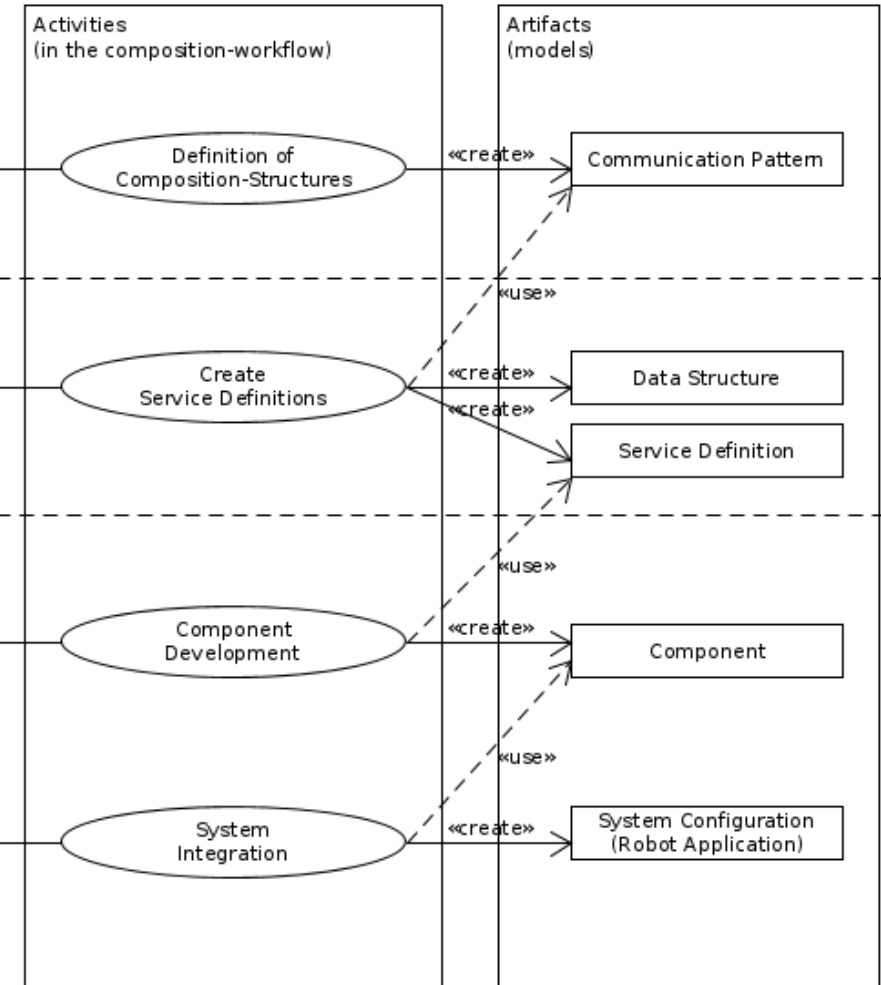
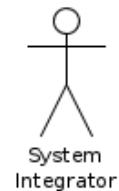
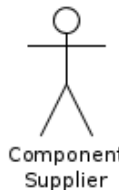
**Tier2**  
Structures the domains within robotics.

This is e.g. the manipulation domain



**Tier3**  
These are the users of the ecosystem. It is about providing and using content.

E.g. SMEs providing specific solutions as component or building concrete systems.



(list is not complete)

Papyrus for Robotics is a model-based development environment that supports the RobMoSys methodology

- ▶ **Modular and role-based design**  
provide dedicated views and abstractions for the different stakeholders in the robotics value chain
- ▶ **Code-generation**  
transform models of software architectures, platform descriptions and deployment specifications into code
- ▶ **Reverse engineering**  
build component and service models from existing systems
- ▶ **Behavior tree execution**  
enable modeling of reactive and composable robot behaviors
- ▶ **Safety analysis**  
perform dysfunctional analysis on component architectures



Reverse  
[youtu.be/fmAWepIiHd0](https://youtu.be/fmAWepIiHd0)



Service Designer  
Tier 2

Services Definition

Skill Definition



Skills Designer



Component Developer

Component Definition

Create ROS2 Build files  
[youtu.be/LZxr66gtHgA](https://youtu.be/LZxr66gtHgA)



System Builder

System Build

Task Modeling



Behavior Developer

Cohesive CDT integration  
[youtu.be/P61COtOuUm0](https://youtu.be/P61COtOuUm0)



Programmer

Generate component code (AMCL)  
[youtu.be/M2J9XxWcgEI](https://youtu.be/M2J9XxWcgEI)

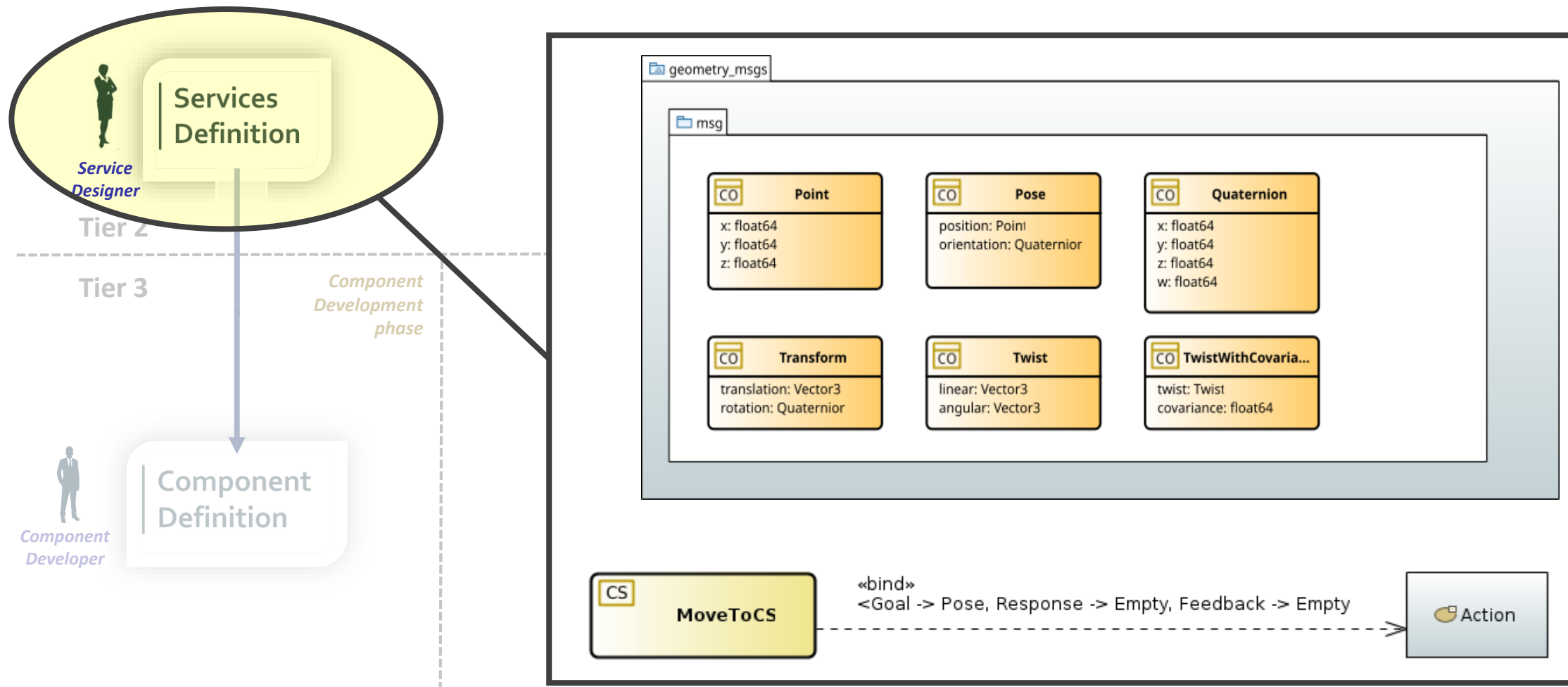
Deploy robotic behavior  
[youtu.be/Mim0cAr1WCs](https://youtu.be/Mim0cAr1WCs)

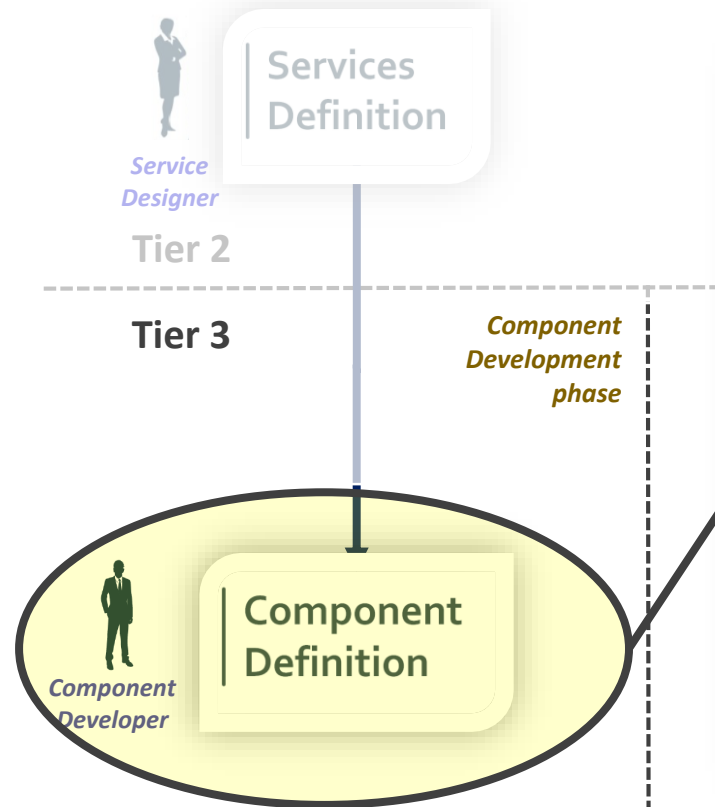
Link Hazards and Risks with Tasks  
[youtu.be/fNbgmT0NQYc?t=163](https://youtu.be/fNbgmT0NQYc?t=163)



Safety Engineer

## Ros2 data types and services (components' data flows / configuration&amp;coordination)





The screenshot displays a software development environment with two property windows and a central diagram:

- Top Property Window (NavigateToPose):**
  - Properties: ID, Ports, Style, Appearance, Rulers And Grid
  - Provided Services: MoveToCS
  - Required Services: <Undefined>
  - QoS: [Empty field]
  - Multiplicity: 1
- Bottom Property Window (cmd\_vel):**
  - Properties: ID, Ports, Style, Appearance, Rulers And Grid
  - Provided Services: P\_Twist
  - Required Services: <Undefined>
  - QoS: [Empty field]
  - Multiplicity: 1
- Central Diagram:**
  - Shows a component named **Navigation** containing an **Activity1**.
  - Inputs: laser, odom.
  - Output: cmd\_vel.
  - Arrows indicate connections from the **Navigation** component to the **NavigateToPose** property window and from the **cmd\_vel** output to the **cmd\_vel** property window.
- Right Panel:** A list of components and their relationships, including Component Definition, Component Port, Component Coordination Port, Parameter block, Activity, Activity Port, Periodic Task, connects, Link, Assertions, and Contract.





Component Developer

## Component Definition

code-generation



Programmer

binds data from component interface to internals (algorithms, etc.)

- `const auto goal = goal_handle->get_goal();`
- `res = do_navigate_algo ( goal->position.x, goal->position.y, ... );`
- `if (res == 1) { // algo OK } else {algo KO }`

The screenshot displays an IDE interface with two main panes. On the left is the 'Project Explorer' showing a tree view of a ROS package named 'NavigationComponent'. The tree includes folders for 'Includes', 'launch', 'models', 'src', and 'src-gen'. Under 'src', there are subfolders 'NavigationCompdef' and 'src-skel'. The 'NavigationCompdef' folder contains 'Navigation\_impl.cpp' and 'Navigation\_impl.h'. An arrow points from this file in the project explorer to the code editor on the right. The code editor shows the implementation of 'Navigation\_impl.cpp', featuring several ROS action methods: 'NavigateToPose\_goal', 'NavigateToPose\_cancel', 'NavigateToPose\_accepted', and 'odom\_handler'. The code uses ROS action libraries like 'rclcpp\_action' and 'mobilemanipulationservice'.

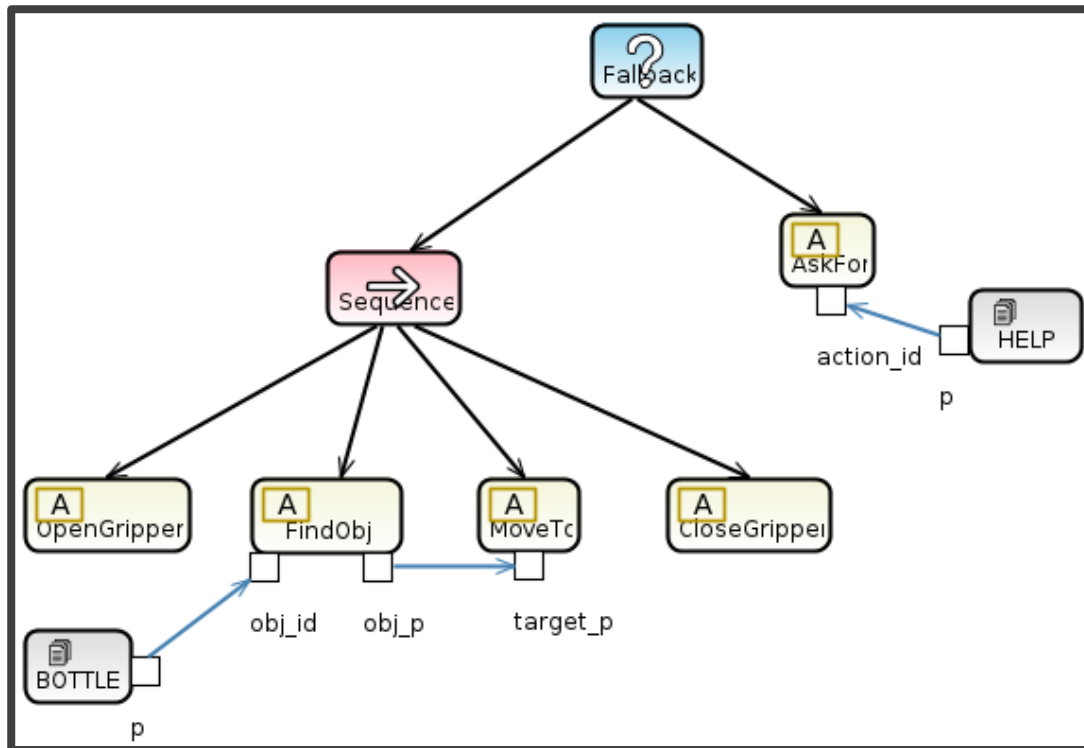
The screenshot shows the 'MobileManipulation' software interface. On the left, a list of skills includes 'MoveTo (in target p: Pose)'. The main area displays the configuration for the 'MoveTo (target\_p: Pose)' skill, which is associated with the 'Coordination serv' and 'MoveToCS' components. The 'Input Attributes' section contains 'target\_p: Pose'. The interface also shows 'Output Attributes' and 'Result Att' sections.

Skill  
Definition

Tier 2

Behavior  
Development  
phase

Tier 3

Task  
Modeling

code-generation

```
Open [v] [F] ~/devel/papyrus-robotics/roboticsrcp-202...torial/patrolsy... Save [≡] - □ ×
1 // Generated by Papyrus4Robotics
2 //
3
4 #include "geometry_msgs/msg/pose.hpp"
5 #include "mobilemanipulationservicedef/action/move_to_cs.hpp"
6 #include "nav2_behavior_tree/bt_action_node.hpp"
7
8 class MoveToAction : public
9   nav2_behavior_tree::BtActionNode<mobilemanipulationservicedef::action::MoveToCS>
10 {
11 public:
12   MoveToAction(
13     const std::string& name,
14     const std::string & action_name,
15     const BT::NodeConfiguration& conf)
16 :
17   nav2_behavior_tree::BtActionNode<mobilemanipulationservicedef::action::MoveToCS>(
18     action_name, conf)
19 {
20 }
21
22 void on_tick() override
23 {
24   geometry_msgs::msg::Pose target_p;
25   getInput("target_p", target_p);
26   goal_.position = target_p.position;
27   goal_.orientation = target_p.orientation;
28 }
29
30 // MoveTo has in/out parameters => must provide a providedPorts method
31 static BT::PortsList providedPorts()
32 {
33   return{
34     BT::InputPort<geometry_msgs::msg::Pose>("target_p")
35   };
36 }
37
38 #include "behaviortree_cpp_v3/bt_factory.h"
39 BT_REGISTER_NODES(factory)
40 {
41   BT::NodeBuilder builder =
42     [](const std::string & name, const BT::NodeConfiguration & config)
```

**MoveTo (target\_p : Pose)**

Coordination serv **MoveToCS**

Skills

ID

Style

Appearance

Rulers And Grid

Advanced

Input Attributes

Output Attributes

Result At

target\_p : Pose

## Code-generation

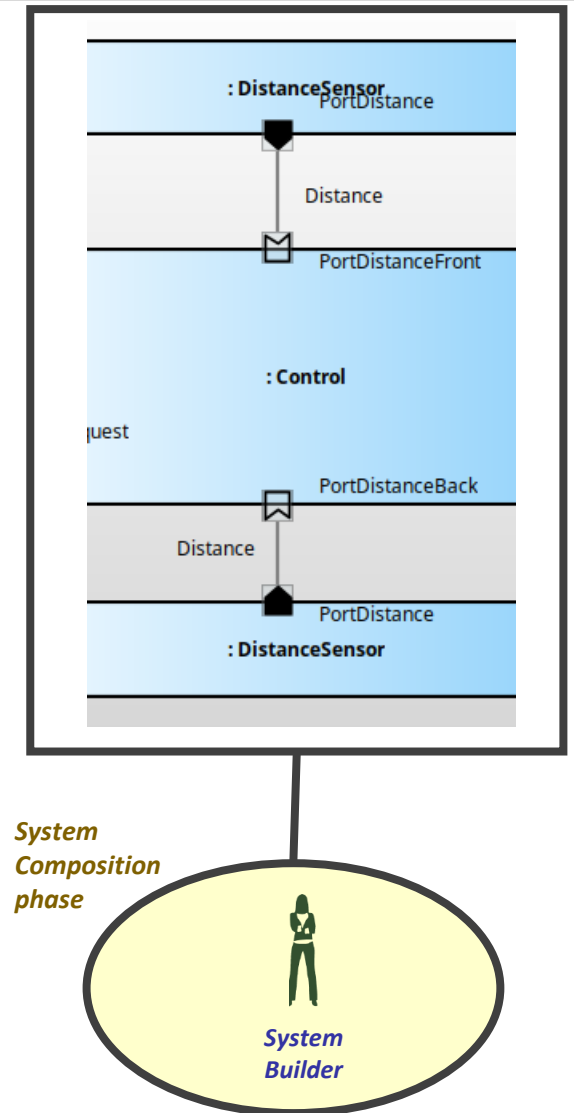
- ▶ **package.xml and CMakeLists.txt**  
configure Eclipse CDT to use colcon
- ▶ **launch script with re-mappings according to components' composition**  
launch scripts are also generated that activate components automatically
- ▶ **YAML files for parameter configuration**  
default value overridden per instance

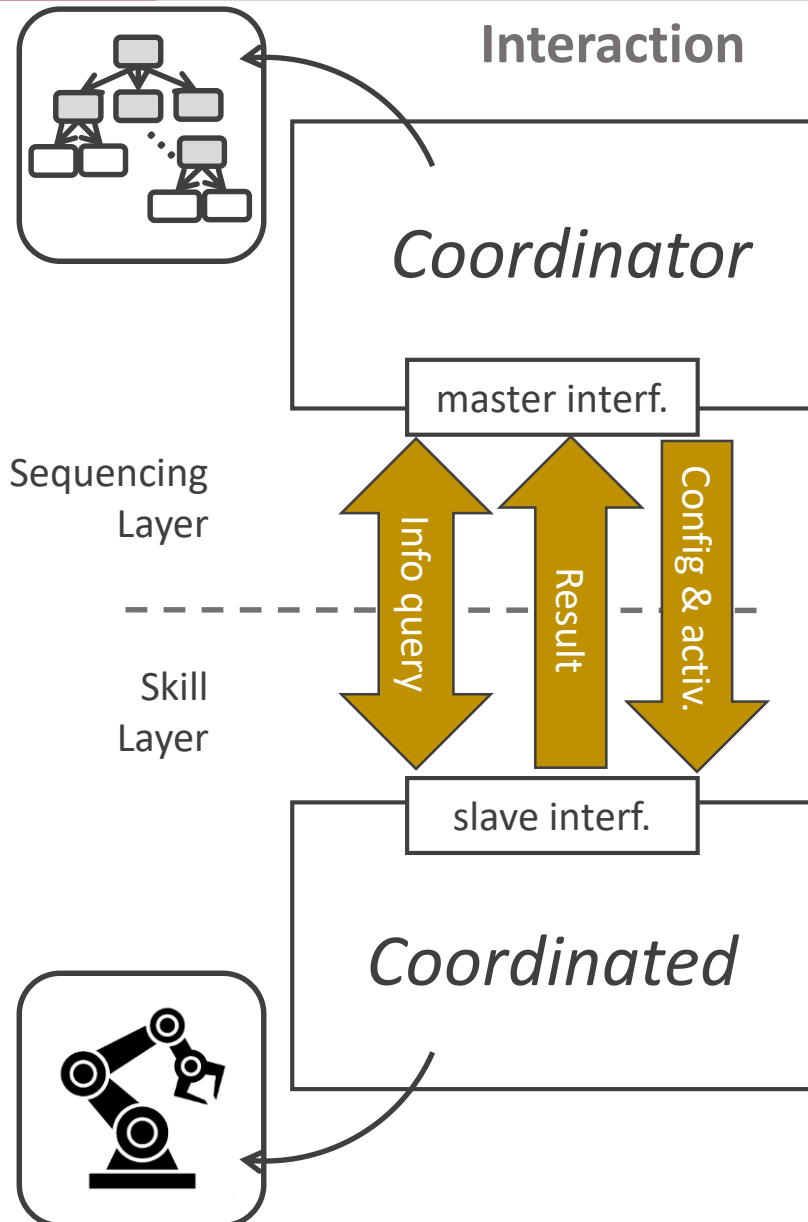
code-generation



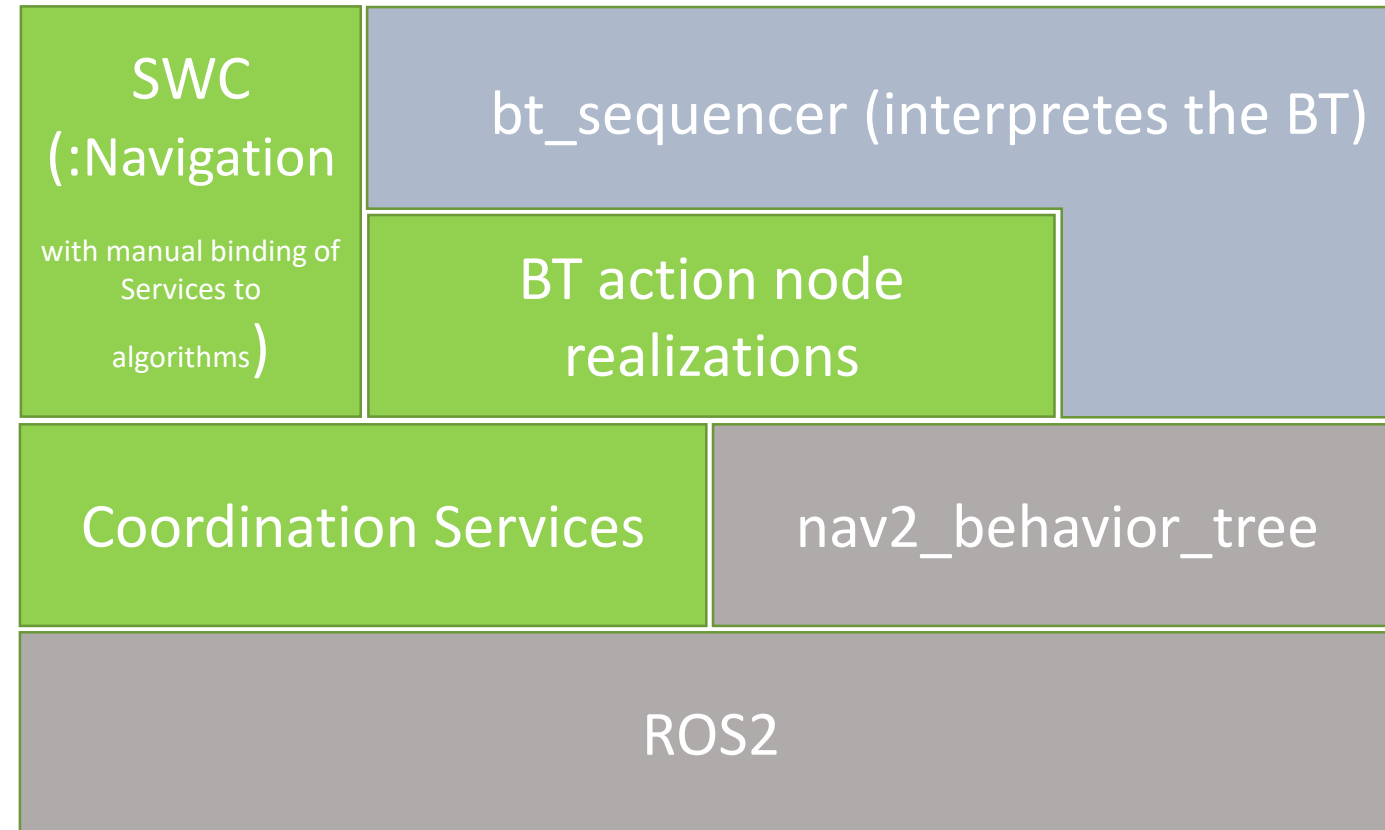
## Eclipse integration

- ▶ Can launch and debug (CDT debugger) a component from Eclipse  
<https://youtu.be/kWkUpKcJq48>
- ▶ Use of Eclipse launch configurations / console



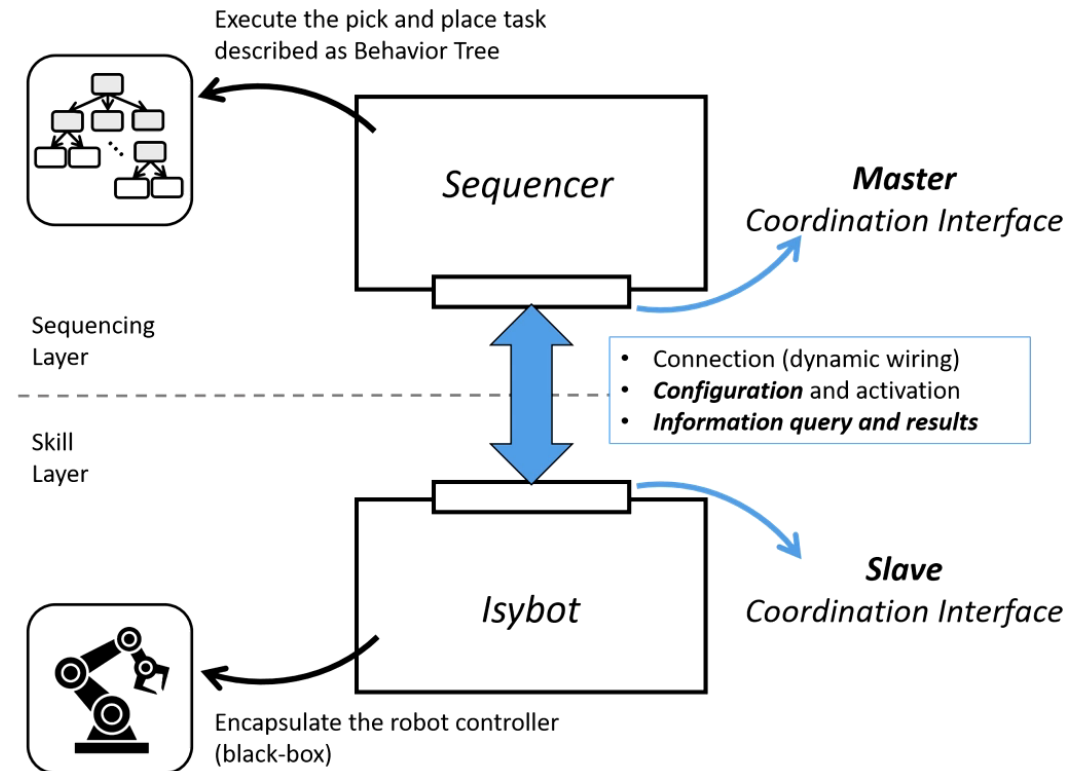


## Run-time architecture

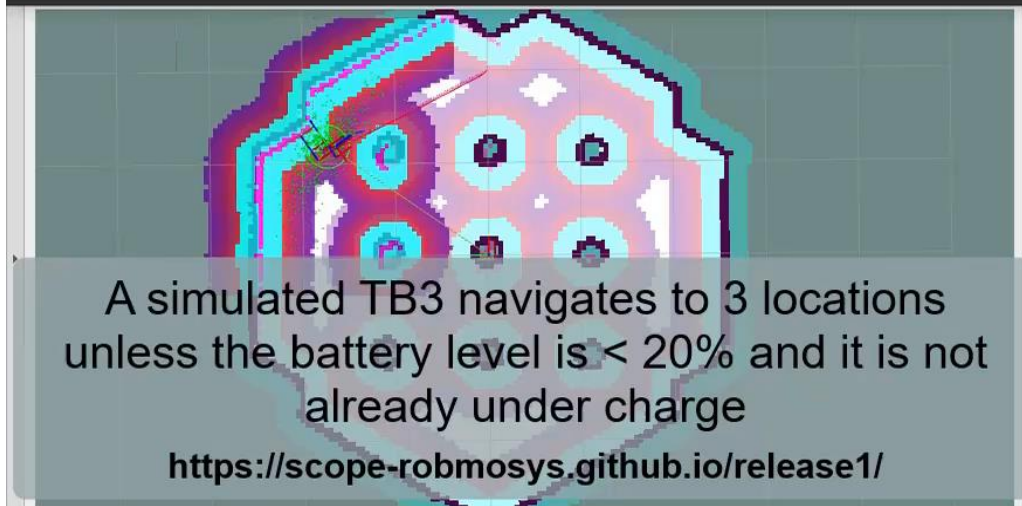


- Supporting modules (from standard ROS2)
- Supporting modules (customized from standard ROS2)
- Generated modules

- Context: simulation of the collaborative robot arm Isybot performing a pick and place task
- System composed of 2 components -- the **Sequencer** executes the **behavior tree** specification of pick-and-place task by opportune **configuration and activation of the Isybot component**
- The **coordination interface** conforms to the “Architectural Pattern for Component Coordination” [1] introduced in RobMoSys and is ***generated from models of the behavioral and system specification***



[1] [https://robmosys.eu/wiki/general\\_principles:architectural\\_patterns:component-coordination](https://robmosys.eu/wiki/general_principles:architectural_patterns:component-coordination)

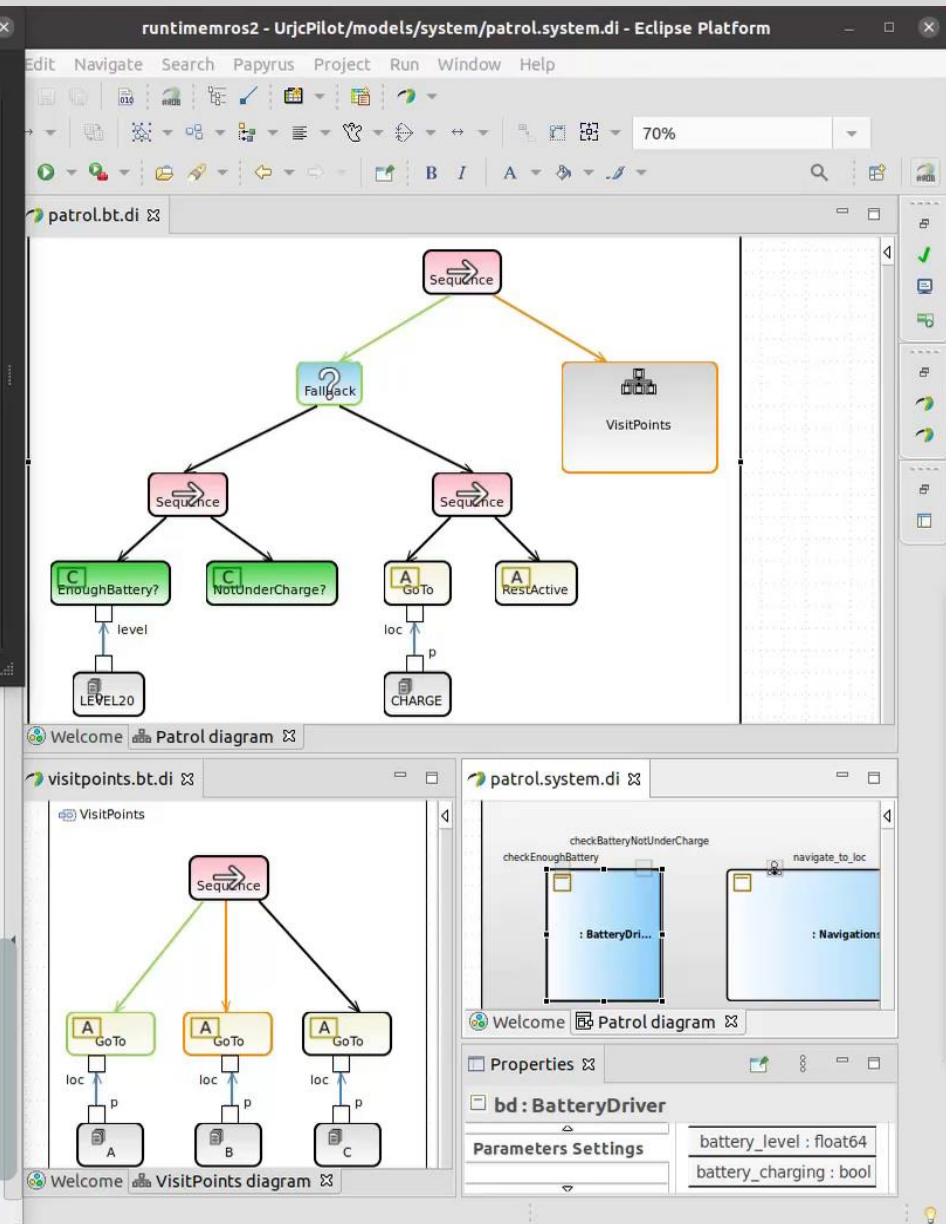


A simulated TB3 navigates to 3 locations unless the battery level is  $< 20\%$  and it is not already under charge

<https://scope-robmosys.github.io/release1/>

Reset Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click: Zoom. Shift: More options.

31 fps



## Goals

- ▶ **Reduce the effort spent in programming**  
component development effort and system integration effort (by generating artefacts from the models).
- ▶ **Improve the quality and safety of the obtained system**  
higher consistency due to generation, improved safety via HARA integration

$$M_{\text{CEA-M3}} = \frac{1}{|C|} \sum_c \frac{\text{Generated LoC} - \text{LoC due to modeling effort}}{\text{Total LoC}}$$

	Body files		Header files	Service files	Build files	
	<i>Generated</i>	<i>Manual</i>	<i>Generated</i>	<i>Generated</i>	<i>Generated</i>	<i>Manual</i>
HMI	166	28	225	10	190	7
Skill Server	422	192	381	45	199	7

Table 5.2: Generated and manually added LoC for the C++ header/body files, service and build files of components `AdaptiveHMI` and `RobotSkillServer`.

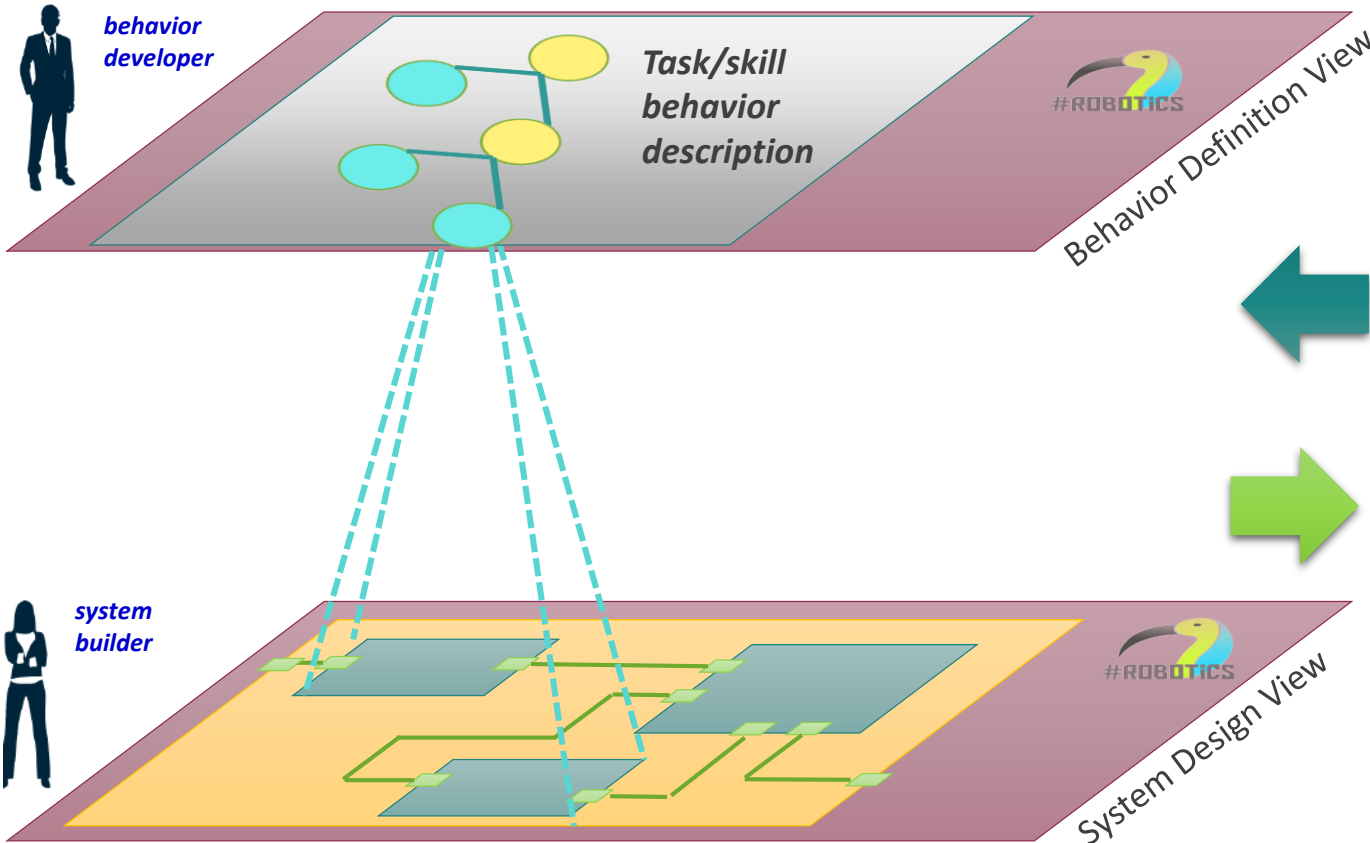


Reduction btw 35% and 67,5%  
(depending on the user background and expertise in programming)





risk = hazard x exposure



Hazard	Likelihood	Criticality	Exposure
tilt	M	LOM	2 M
rack surface	L	Limited pitch control -Induced divergence -LOM, LOC, LOW	1 M
hardover	L	Limited pitch control -Induced divergence -LOM, LOC, LOW	1 M
floating surface	M	Limited pitch control -LOM, LOC, LOW	1 H
oscillatory	L	Limited pitch control -Induced divergence -LOM, LOC, LOW	1 M
...	L	Slow actuator dynamics -Limited pitch control	2 M

Likelihood	Criticality					
	4	3	2R	2	1R	1
High	M	M	H	H	H	H
Medium	L	M	M	M	H	H
Low	L	L	M	M	M	M

Hazard Analysis and Risk Assessment view

- ▶ **Task-Based HARA is performed following ISO 10218-2:2011.** For each action in the behavior tree, we list all the relevant hazards and compute their risk index. The risk analysis table structure is extracted from ISO/TR 14121-2:2007.

**RobMoSys**  
Composable Models and Software  
For Robotics Systems

The RobMoSys  
Robotics Platform

Modular & Role-  
Based Design

Agile Risk  
Assessment

Compositional  
Safety Analysis

Robustness  
Simulation

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 732410

runtime-EclipseApplicationTableESF - CEARobotPrinterModel/pickandplacebehavior.di - Eclipse Platform

File Edit Navigate Search Papyrus Project Run Window Help

Model Explorer

- Task Definition
  - PrinterPaperPickAndPlace
    - Initialisation
    - PrepareToGrasp
    - GraspUnGrasp
      - «TreeRoot, Task» PrinterPaperPicl
      - «TreeRoot, Task» Initialisation
      - «TreeRoot, Task» PrepareToGrasp
      - «TreeRoot, Task» GraspUngrasp
        - «Uses» <Usage> Initialisation
        - «Uses» <Usage> PrepareToGrasp
        - «Uses» <Usage> GraspUnGrasp
        - «Uses» <Usage> MovementSkills
        - «Uses» <Usage> GraspingSkills
        - «Uses» <Usage> MovementSkills
        - «Uses» <Usage> GraspingSkills
        - «Uses» <Usage> MovementSkills
        - «Uses» <Usage> GraspingSkills
    - HazardAnalysis
      - GraspUnGraspHazardAnalysis
        - HazardAnalysisTable
          - «HazardAnalysis» moveme
          - «HazardAnalysis» unintent
          - «HazardAnalysis» unintent
          - «HazardAnalysis» end-effe
          - «HazardAnalysis» materi:
          - «HazardAnalysis» unexpect
          - «HazardAnalysis» unintent
          - «HazardAnalysis» unintent

	A	B	C
L	Task	Hazard	Origin
1	movements of ro...	moveTo (p : Pose3d)	movements of robot arm
2	unintended mov...	moveTo (p : Pose3d)	unintended movement
3	unintended mov...	openGripper ()	unintended movement
4	end-effector failu...	openGripper ()	end-effector failure (separation)
5	materials and pr...	openGripper ()	materials and products falling or ejection
6	unexpected relea...	moveTo (p : Pose3d)	unexpected release of potential energy fr.
7	unintended mov...	moveTo (p : Pose3d)	unintended movement of the gripper
8	unintended mov...	closeGripper ()	unintended movement of the gripper

GraspUnGraspHazardAnalysis

UML Name GraspUnGraspHazardAnalysis

Table Label

Appearance Is abstract  true  false

Paste Visibility public

Comments

Profile Protocol <Undefined>

GraspUnGraspHazardAnalysis

UML Name GraspUnGraspHazardAnalysis

Table Label


Appearance Is abstract  true  false

Paste Visibility public

Comments

Profile Protocol <Undefined>

Risk assessment is performed assessing operational hazard situations and mitigation measures.



**RobMoSys**  
Composable Models and Software  
For Robotics Systems

paperus  
#ROBOTICS

The RobMoSys  
Robotics Platform

Modular & Role-  
Based Design

Agile Risk  
Assessment

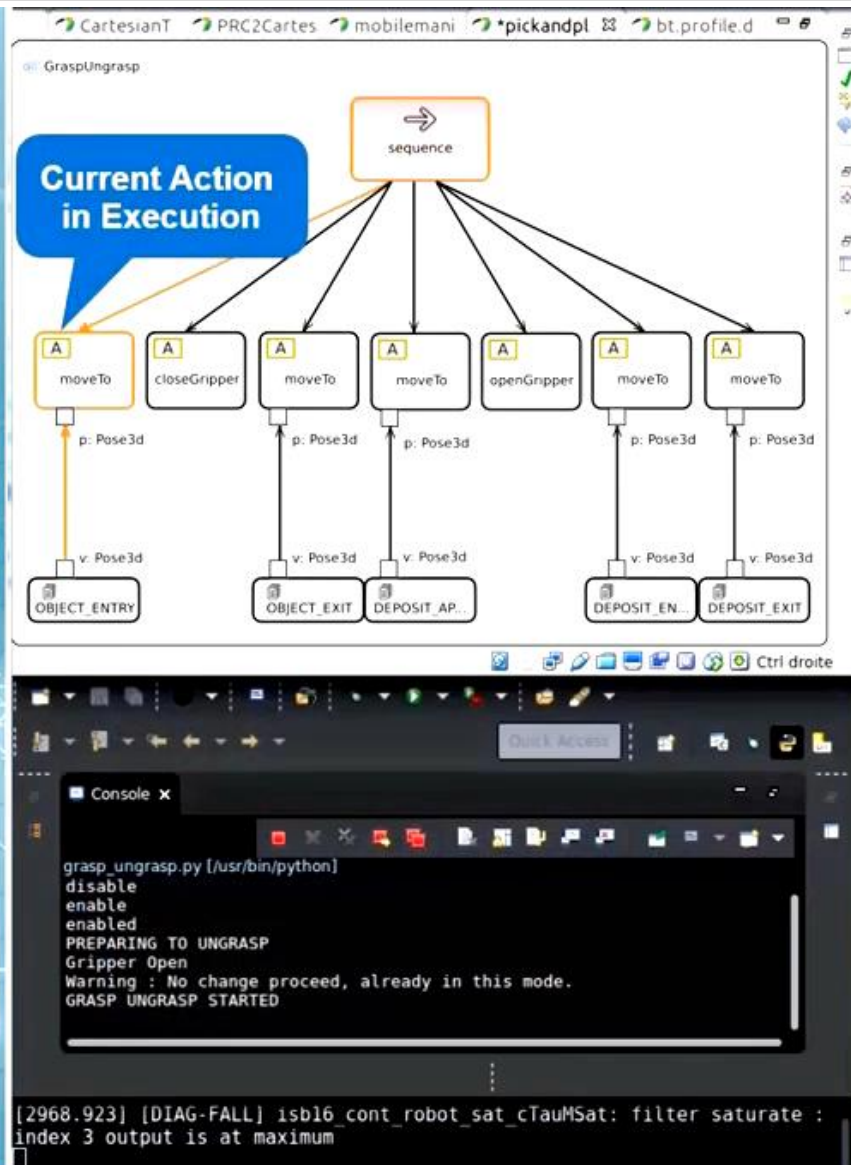
Compositional  
Safety Analysis

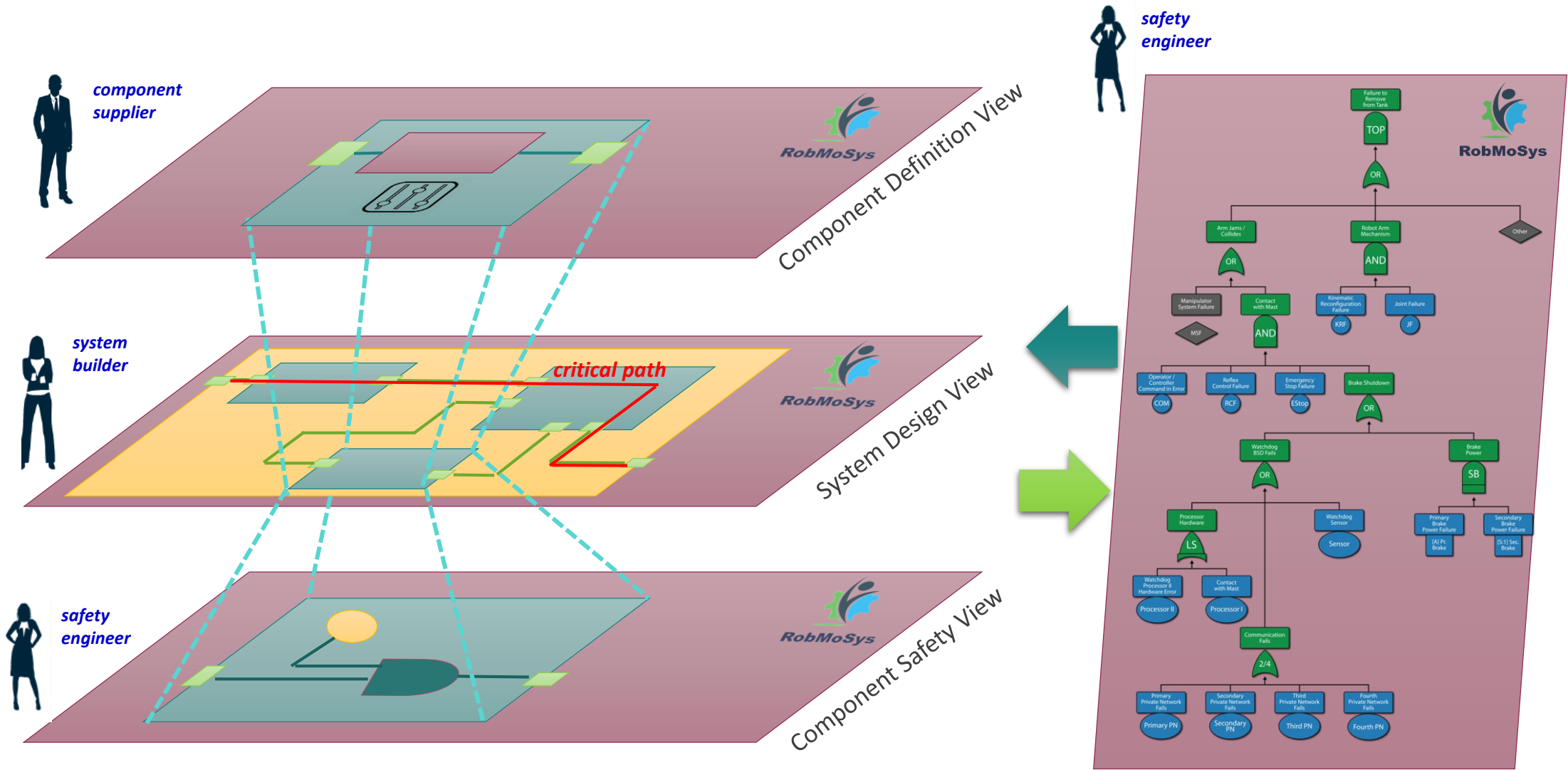
Robustness  
Simulation

eit Digital iCenter  
ALTERNATIONS PARIS

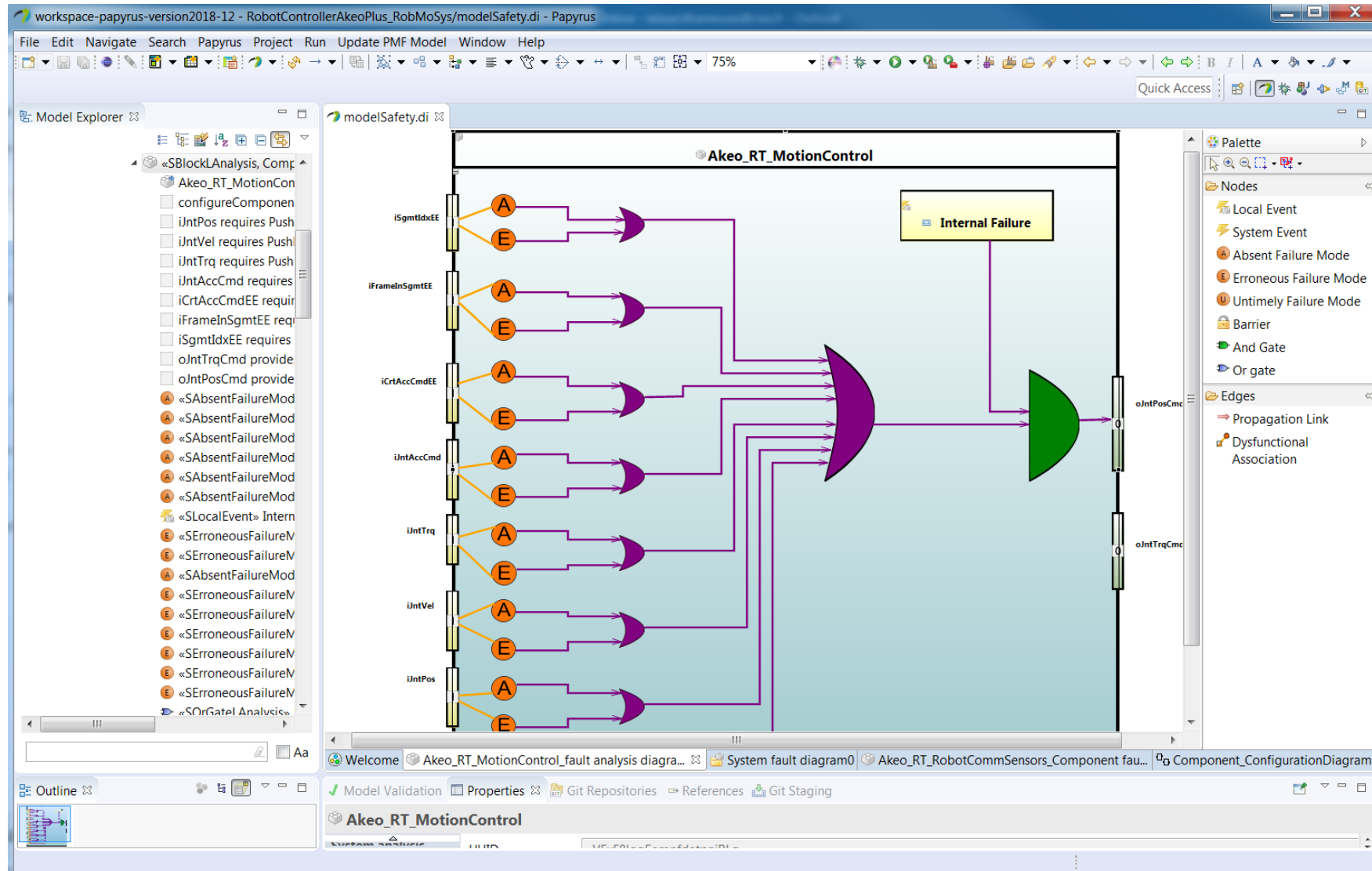
This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 732410

list  
cea tech

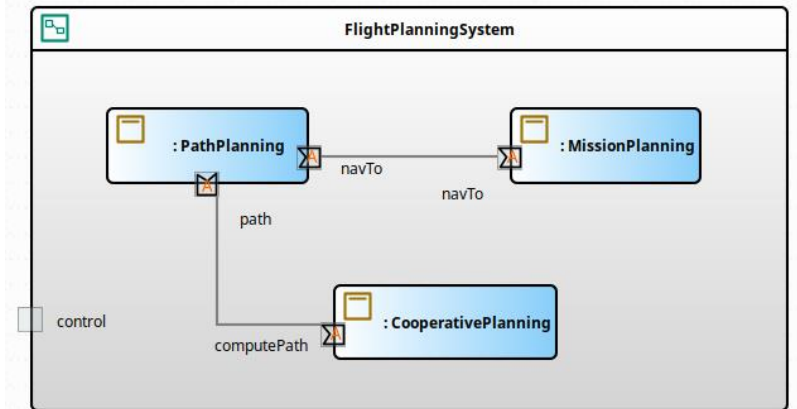




Fault Tree Analysis (FTA) View



- ▶ P4R used internally in several (European) projects  
⇒ Continues to be developed actively
- ▶ Support for composite components to model whole-part relationship
- ▶ Modeling and deployment of ROS2-based automated planning solutions
- ▶ Real-time support, model of computation & communication (MoCC)
  - ROS2 and pyCPA support



## 2 Planning System

