Ref. Ares(2026)5995162 - 12/06/2026



SOPRANO
Socially-Acceptable and Trustworthy Human-Robot Teaming for Agile Industries

Project Number 101120990

D2.4 Customizable Task Execution II

Version 1.0
31 May 2026
Final

Public Distribution

FORTH and PROFACTOR

Project Partners: **AEGIS, ATB, CEA, Centro Ricerche Fiat, Circular Economy Foundation, E-TERRY, F6S, FORTH, Harokopio University of Athens, IFADO, Jade University, KUKA Assembly & Test, Netcompany-Intrasoft, NTUA, PROFACTOR, Technology Transfer Systems, The Open Group, University of Bremen, University of York**

Every effort has been made to ensure that all statements and information contained herein are accurate, however the SOPRANO Project Partners accept no liability for any error or omission in the same.

© 2026 Copyright in this document remains vested in the SOPRANO Project Partners.

PROJECT PARTNER CONTACT INFORMATION

AEGIS Nikos Kostalas Humboldtstrasse 25 38106 Braunschweig, Germany E-mail: n.kostalas@aegisresearch.eu	**ATB** Sebastian Scholze Wiener Strasse 1 28359 Bremen, Germany E-mail: scholze@atb-bremen.de
CEA Nikolaos Matragkas NanoInnov Bat. 862 PC 174 91191 Gif-sur-Yvette Cedex, France E-mail: nikolaos.matragkas@cea.fr	**Centro Ricerche Fiat** Felice Tauro Strada Torino 50 10043 ORBASSANO, Italy E-mail: felice.tauro@crf.it
Circular Economy Foundation Ioannis Basdekis Rue Breydel 34-36-40 1040 Brussels, Belgium E-mail: ioannis.basdekis@circular.foundation	**E-TERRY** Fabian Rösler Neuwerkstrasse 50 99084 Erfurt, Germany E-mail: fabian.roesler@e-terry.de
F6S Gergana Dimitrova 77 Lower Camden Street Dublin D02 XE80, Ireland E-mail: gergana@f6s.com	**FORTH** Manolis Lourakis N Plastira Str 100 70013 Heraklion, Greece E-mail: lourakis@ics.forth.gr
Harokopio University of Athens Iraklis Varlamis Eleftheriou Venizelou 70 176 71 Athens, Greece E-mail: varlamis@hua.gr	**IFADO** Felix Dreger Ardeystrasse 67 44139 Dortmund, Germany E-mail: dreger@ifado.de
Jade University George Athanassiou Friedrich Paffrath Strasse 101 26389 Wilhelmshaven, Germany E-mail: georgios.athanassiou@jade-hs.de	**KUKA Assembly & Test** Georgios Karantinakis Uhthoffstrasse 1 28757 Bremen, Germany E-mail: georgios.karantinakis@kuka.com
Netcompany-Intrasoft John Soldatos 2B Rue Nicolas Bové 1253 Luxembourg, Luxembourg E-mail: john.soldatos@netcompany-intrasoft.com	**NTUA** Maria Pateraki Heroon Polytechniou 9, Zographou Campus 157 80 Athens, Greece E-mail: mpateraki@mail.ntua.gr
PROFACTOR Sharath Akkaladevi Im Stadtgut D1 4407 Steyr-Gleink, Austria E-mail: sharath.akkaladevi@profactor.at	**Technology Transfer Systems** Antonio Avai Via Francesco d'Ovidio, 3 20131 Milan, Italy E-mail: avai@ttsnetwork.com
The Open Group Scott Hansen Rond Point Schuman 6, 7th Floor 1040 Brussels, Belgium E-mail: s.hansen@opengroup.org	**University of Bremen** Nico Hochgeschwender Bibliothekstrasse 1 28359 Bremen, Germany E-mail:ni_ho@uni-bremen.de
University of York Simos Gerasimou Deramore Lane York YO10 5GH, United Kingdom E-mail:simos.gerasimou@york.ac.uk	

DOCUMENT CONTROL

Version	Status	Date
0.1	Initial draft with outline	10 March 2026
0.5	Component descriptions	25 April 2026
0.7	Contributions by partners	2 May 2026
0.8	Internal review comments	10 May 2026
0.9	Final version for QA review	24 May 2026
1.0	Final QA for EC submission	31 May 2026

TABLE OF CONTENTS

1 Introduction **1**
1.1 Purpose of the deliverable *1*
1.2 Relation to other deliverables *1*
1.3 Structure of the deliverable *2*
2 Mapping of high-level plans to Robotic Capabilities [MRC] **3**
2.1 Technology overview and Component Advancements *3*
2.2 Methodology and implementation *4*
3 MH-MR Task Allocation [CTA] **9**
3.1 Technology overview and Component Advancements *10*
3.2 Methodology and implementation *11*
3.2.1 Action Server Interfacing 11
3.2.2 3.2.2 Multi-level task sequencing 12
3.2.3 Interaction with the end-user interface 13
3.2.4 Grouping, prioritizing, and resuming workflows 13
3.2.5 Dynamic Multi-Agent Coordination and Role Adaptation 14
3.2.6 Information rich back-end interface 14
4 MRC & CTA Available for Deployment by Third Parties **17**
5 MRC & CTA Deployment in the CRF Use Case **18**
5.1 Requirements *18*
5.2 Implementation *18*
6 MRC & CTA Deployment in the KUKA Use Case **22**
6.1 Requirements *22*
6.2 Implementation *22*
7 Conclusions and future work **23**
8 Updates Since D2.2 ..... **24**
8.1 Mapping of High-Level Plans to Robotic Capabilities [MRC] *24*
8.2 MH-MR Task Allocation [CTA] *24*

TABLE OF FIGURES

Figure 1. Complete architecture of MRC (Design Time), RCI (Run Time) and communication with CTA 4
Figure 2. Groot2 window showing Object Detection behaviour tree 5
Figure 3. Groot2 window showing unscrew_single behavior tree. 8
Figure 4. A Daisy Plan of robot tasks, where each petal represents a sequence of primitive actions. Smaller (nested) petals connected to specific primitive actions denote special actions triggered within the workflow. 15
Figure 5. The backend-interface enables the direct trigger of petals (workflows) and primitive robot actions to facilitate experimentation. 16

EXECUTIVE SUMMARY

This deliverable *D2.4 Customizable Task Execution II* outlines SOPRANO advancements in relation to *D2.2 Customizable Task Execution I*, aiming to enhance task execution flexibility in multi-robot and human-robot collaborative industrial environments. The current report focuses mainly on the SOPRANO components the Mapping of High-Level Plans to Robotic Capabilities (MRC), developed as part of Task 2.5, and the Multi-Human-Multi-Robot Task Allocation (CTA), developed as part of Task 2.6.

The MRC (Mapping of High-Level Plans to Robotic Capabilities) component provides the functionality required to translate abstract task descriptions into concrete, executable robot actions. It acts as the bridge between planning and execution by decomposing complex tasks into sequences of parameterized primitive actions that can be directly performed by robotic systems. MRC is closely integrated with the Robot Control Interface (RCI), which exposes the available robot capabilities and enables their standardized invocation across different platforms. Through this integration, it supports modular deployment and interoperability with diverse robotic systems and low-level control frameworks. Using ROS2 action services as the primary communication mechanism, it ensures reliable execution, continuous feedback, and robust handling of action outcomes, including error reporting and recovery. Practically, MRC and RCI enable the realization of high-level plans in real-world actions, ensuring that robotic capabilities are effectively aligned with environments, task requirements and operational constraints.

The CTA (Multi-Human Multi-Robot Task Allocation) component provides a flexible and scalable framework for coordinating robot actions in collaborative environments by combining high-level task planning with real-time adaptation. Through its two main modules, the Daisy Planner and the fHRI, it supports both mid-term task scheduling and short-term, context-aware adjustment of robot behaviour. The latest version of CTA, enables workflow generation, either from predefined configurations or runtime inputs (e.g., structured JSON messages), and can operate across multiple agents, assigning tasks based on availability, capabilities, and system state. It supports both detailed control of primitive action sequences and integration with higher-level, pre-existing robot behaviours, ensuring compatibility with diverse systems. The component also facilitates two-way communication with user interfaces, providing transparency on current and planned actions while allowing human intervention (e.g., pause/resume or action triggering). Additional capabilities include workflow grouping and prioritization for repeated task cycles, coordination across multiple workstations, and support for dynamic role allocation of robots. The above features support CTA in achieving efficient, safe, and human-centric coordination of multi-agent systems, easily adapted to the operational requirements of the individual use cases.

The MRC and CTA components have been successfully deployed and validated in two distinct SOPRANO use cases, demonstrating their adaptability across different industrial contexts. In the CRF use case, they support human–robot collaboration in automotive disassembly processes, coordinating mobile (FELICE) and stationary (UR10) robots to assist workers through tool provisioning and task execution. In the KUKA use case, they are applied to robot-supported building renovation, where workflows are dynamically generated from BIM-derived task descriptions and executed by the robotic drilling platform. These deployments highlight the ability of MRC and CTA to operate both in structured, repetitive environments and in more dynamic, data-driven scenarios, confirming their flexibility and readiness for real-world applications.

1 INTRODUCTION

1.1 PURPOSE OF THE DELIVERABLE

The purpose of this deliverable is to present the advancements achieved in adaptable Multi-Human Multi-Robot (MH–MR) task execution mechanisms within the SOPRANO framework, with particular emphasis on improvements that enhance their adaptability and facilitate deployment across diverse industrial applications. Special attention is given to enabling seamless interaction with other SOPRANO components through updated communication protocols and integration strategies.

The document details the technological developments introduced during the reporting period, and summarizes the experimental activities conducted to assess system flexibility, robustness, and adaptability in real-world industrial scenarios involving both human–robot and multi-robot collaboration. It further describes the progress made in transforming expert's knowledge and high-level task plans into executable robotic processes, highlighting the design, implementation, and validation of two core components: the Mapping of High-Level Plans to Robotic Capabilities (MRC) and the Multi-Human Multi-Robot Task Allocation (CTA). The MRC component focuses on translating abstract task descriptions into concrete robot actions, while the CTA component is responsible for task allocation, scheduling, and coordination among multiple agents in collaborative environments.

Overall, this deliverable describes the steps taken for advancing MRC and CTA to support the SOPRANO system toward real-world deployment, demonstrating increased maturity, robustness, and interoperability. In particular, it establishes that both the MRC and CTA components have reached Technology Readiness Level 6 (TRL6), having been validated in operationally relevant environments. This confirms their capability to support flexible, scalable, and reliable deployment in industrial scenarios with varying requirements.

1.2 RELATION TO OTHER DELIVERABLES

The current deliverable D2.4 builds upon the foundations established in deliverables *D2.2 Customizable Task Execution I* and *D5.3 Use Cases Evaluation I*, which primarily focused on the technical development of key components and their integration into the project's early prototypes. The progress presented in the current document reflects the maturation of these initial developments, extending them through iterative refinement, enhanced system integration, and validation under increasingly realistic operational conditions. Many of the advancements reported here have been directly informed and shaped by insights gained during real-world experimentation activities, part of which have already been documented in *D5.3 Use Cases Evaluation I* and *D5.4 Analysis of MH-MR Collaboration in Early Prototype Evaluations*. These experimental activities have provided valuable feedback on system robustness, usability, and human–multi-robot collaboration performance, enabling the consortium to align technical developments more closely with practical deployment requirements.

With respect to forthcoming SOPRANO deliverables, the work summarized in the current document constitutes a critical technological step for the next phase of real-world experimentation and integrated system validation. In particular, the presented developments provide essential background technologies, methodological framework, and integration maturity required for the activities that will be further consolidated and evaluated in *D5.5 Integrated Technology II* and *D5.6 Use Cases*

Evaluation II. As such, this deliverable plays a pivotal role in ensuring continuity between component-level innovation and fully integrated demonstrations in realistic operational environments.

1.3 STRUCTURE OF THE DELIVERABLE

The deliverable is organized into seven sections, each addressing a key aspect of the development, integration, and validation of the MRC and CTA components within the SOPRANO framework.

- Section 1: The current section outlines the objectives and scope of the deliverable, positioning it within the overall SOPRANO project. It introduces the concept of adaptable MH–MR task execution and presents the roles of the MRC and CTA components.
- Section 2: This section focuses on the advancements of the MRC component, describing a Behaviour Tree–based approach for mapping high-level plans to executable robot actions. It covers the design of Behaviour Trees, the runtime implementation within the RCI, and the ability to control different robot platforms while ensuring execution reliability and flexibility.
- Section 3: This section presents the design, functionality, and recent improvements of the CTA component. It highlights key features such as dynamic task allocation, workflow management, human–robot interaction mechanisms, and multi-agent coordination.
- Section 4: This section outlines how the components have been made available to third-party projects. It discusses their adoption by open-call subprojects and highlights their flexibility and reusability across different application domains.
- Section 5: This section describes the application of MRC and CTA in the CRF use case. It presents the adaptations required to support scenarios involving two robotic platforms with different characteristics, along with key scenarios and validation experiments.
- Section 6: This section describes the application of MRC and CTA in the KUKA use case. It focuses on adaptations for building renovation scenarios, including the integration of existing and newly developed robot behaviours.
- Section 7: The final section presents the conclusions, summarizing the main achievements of the deliverable. It also outlines future work, focusing on extended MH–MR testing and further refinement based on real-world deployment feedback.

2 MAPPING OF HIGH-LEVEL PLANS TO ROBOTIC CAPABILITIES [MRC]

2.1 TECHNOLOGY OVERVIEW AND COMPONENT ADVANCEMENTS

The Mapping of High-Level Plans to Robotic Capabilities (MRC) component constitutes a core element of the SOPRANO framework, enabling the transformation of abstract, high-level task plans into executable robotic actions. Building upon the foundations established in deliverable *D2.2 Customizable Task Execution I*, where the MRC component was introduced as a primarily design-time module leveraging Behaviour Trees (BTs) for hierarchical task decomposition, the current iteration extends its capabilities towards a more integrated, runtime-aware, and use-case validated system.

In D2.2, the MRC component focused on the development of BT-based task decomposition mechanisms and their validation in simulation environments, primarily within the CRF use case. The component demonstrated the ability to decompose complex tasks into modular and reusable actions and to map these actions to robotic capabilities, with initial integration interfaces defined towards runtime components such as the Robot Capabilities Implementation (RCI). MRC acts as a design-time component creating behaviour trees for individual robots to be executed, while RCI with its Behaviour Tree Run Time Layer (BTRL) i.e. Dynamic Behaviour Tree Engine (DBTE) utilizes these behaviour trees and interacts with low-level interfaces to showcase the Robot Capability. Therefore, CTA interacts with the run time component RCI while all the heavy lifting of designing behaviour trees is done by MRC.

Compared to D2.2, significant advancements have been achieved, namely:

- First, the integration between design-time planning (MRC) and runtime execution has been substantially strengthened. While D2.2 primarily defined this interaction conceptually, the current implementation establishes tighter coupling with WP4 components, particularly the RCI and execution frameworks reported in D4.1(Modular, Reconfigurable and Flexible Tools I) and D4.2 (Modular, Reconfigurable and Flexible Tools II). This enables the direct operationalization of BT-based task plans into executable robotic behaviours across different platforms.
- Second, the MRC component has evolved from a simulation-based validation stage to deployment within real-world use case scenarios, including both the CRF and KUKA use cases. This transition introduces support for heterogeneous robotic systems, requiring enhanced capability mapping, parameterization, and interoperability with perception, and robot proprioception. As a result, the MRC now supports more complex workflows, including multi-step disassembly processes and autonomous navigation-assisted operations.
- Third, the adaptability and robustness of the BT-based execution framework have been improved. Enhancements include better handling of dynamic environmental changes, improved error recovery strategies through fullback mechanisms, and tighter integration with real-time feedback from perception and robot proprioception. These improvements allow the system to react more effectively to uncertainties (to a certain extent locally at the MRC/RCI level) observed during early prototype evaluations, such as variations in human behaviour, and environmental changes identified in D5.4 (Analysis of MH-MR Collaboration in Early Prototype Evaluations).

Overall, the MRC component has progressed from a design-time, simulation-driven framework in D2.2 towards a more mature, integrated, and experimentally validated solution capable of supporting customizable task execution in real multi-human multi-robot (MH-MR) environments. These advancements represent a key step towards achieving higher Technology Readiness Levels and enabling scalable deployment across diverse industrial use cases (CRF and KUKA).

2.2 METHODOLOGY AND IMPLEMENTATION

Here we detail the design time component MRC under Task 2.5 (Adapted and Customizable Task Execution) while the run time component RCI has been described in D4.2 under Task 4.6 (Robot Capabilities Implementation). The link between MRC and RCI is visually outlined in Figure 1.

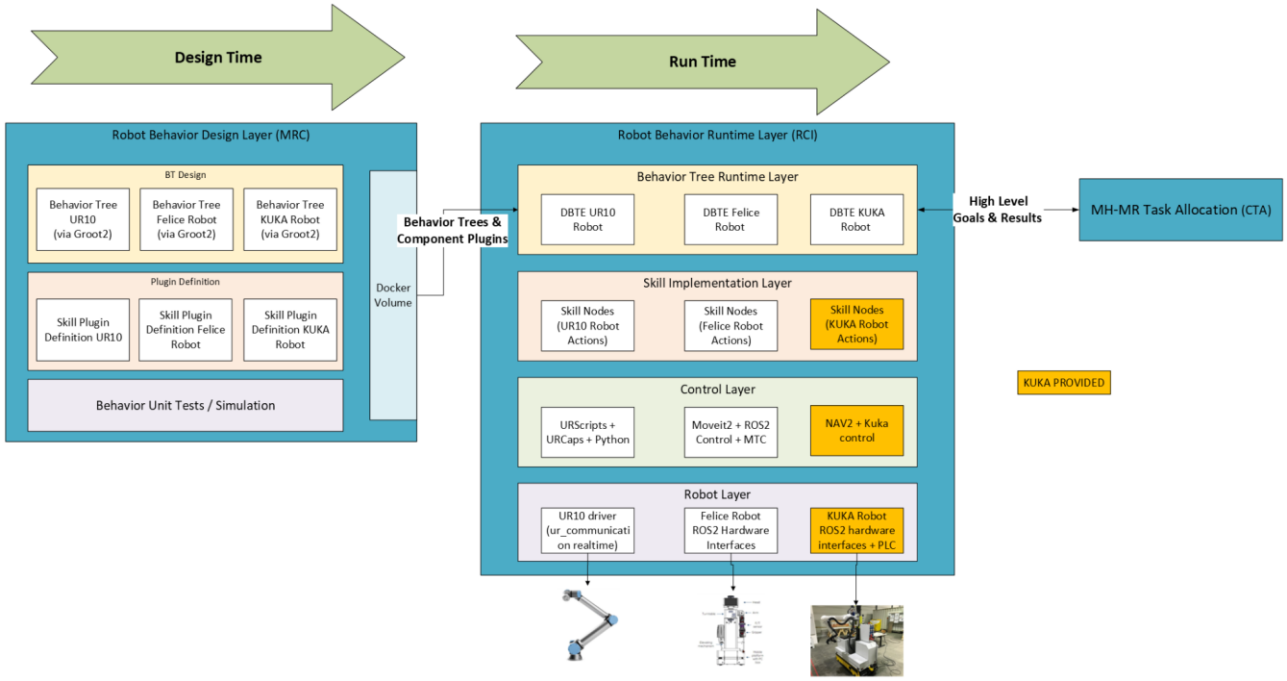


Figure 1. Complete architecture of MRC (Design Time), RCI (Run Time) and communication with CTA

MRC component can be divided into three layers:

- BT Design, this layer is the most crucial layer in defining how the high-level action can be broken down to robotic executable action such that it could easily be visualized and edited. This layer uses BehaviorTree.cpp[1], modified BehaviorTree.ROS2[2] and Groot2[3] to provide a design tool for behaviour trees designing and programming.

This part allows one to use Groot2 to create initial robot behaviours like navigation with recovery or tool pickup involving scanning the tool, then planning pickup and finally grasping the tool and coming back while handling deviations if any. Groot2 also allows adding new action nodes, condition

[1] https://github.com/BehaviorTree/BehaviorTree.CPP
[2] https://github.com/BehaviorTree/BehaviorTree.ROS2
[3] https://www.behaviortree.dev/groot/

nodes etc. to incorporate atomic skills (Move Arm, Operate Gripper etc.) and utility nodes (PrintString, PrintPose, DoTransform etc.) in the behaviour tree. The combination of the atomic skill nodes and utility nodes actually forms the usable MRC behaviour tree.

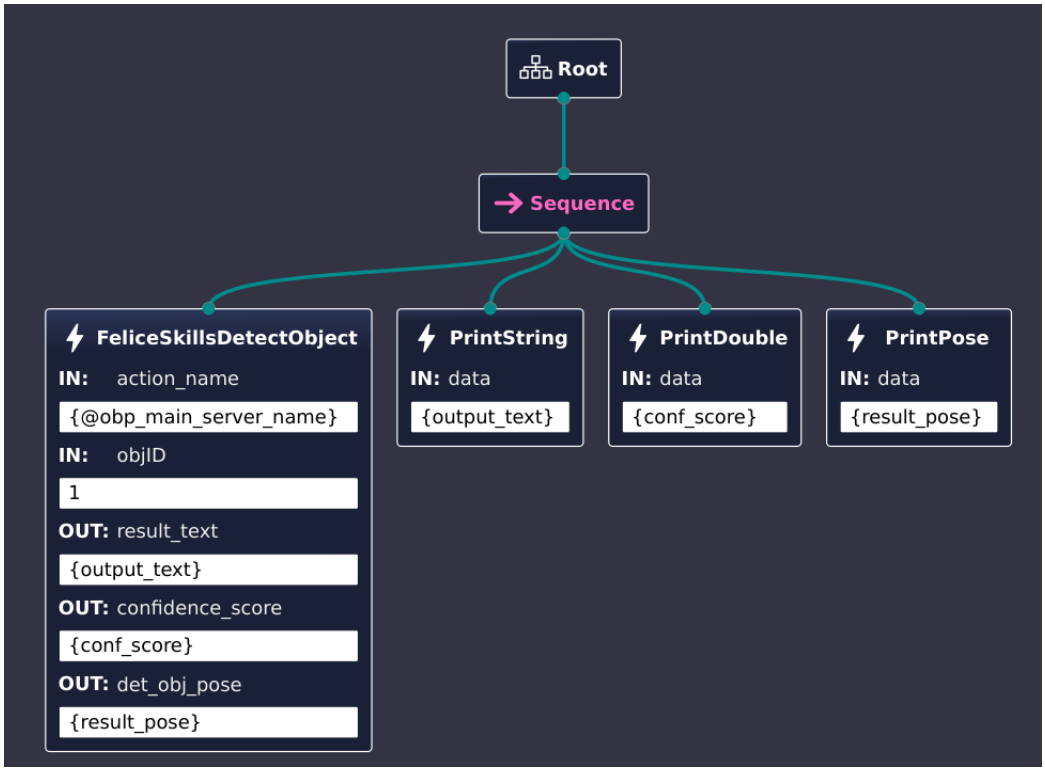


Figure 2. Groot2 window showing Object Detection behaviour tree

Figure 2 shows an object detection behaviour tree combining atomic skill of object detection with utility node to show a clean editable behaviour tree. Most importantly, the initial behaviour trees and node definition in Groot2 allows seamless addition or deletion of behaviour trees for the user of this component.

- Plugin Definition, this layer defines how the atomic skill should be invoked i.e when the behaviour tree runs within RCI what should happen when an atomic skill is being called. In SOPRANO ROS2 is used as communication between components and the atomic skills are wrapped around ROS2 action servers or services. The plugin definition defines ROS2 action server clients or service server clients as behaviour tree nodes such that these could be effectively placed inside a behaviour tree.
- Behaviour Unit Tests/simulation and docker volume, this layer provides sanity to the designed behaviour trees by performing unit tests for the design combined with plugins. It does so by using simulation (fake controllers, gazebo, URSim) or demo atomic skill action/service servers to test the complete design flow. Once satisfied, the MRC component is ready for integration, and the mrc_robot_bhvrs ROS2 package is available through docker volume which is loaded by the RCI's run time layer BTRL(Behaviour Tree Run Time Layer).

In order to allow error handling throughout the application stack, all behaviour tree plugins and actions share a common error handling convention. Errors have to be indicated with one of the following error codes:

Error Code	Label	Description
0	NO_ERRORS	everything is fine
1	ARM_EXT_FAILED	Failed to extend or position arm
2	REACH_FAILED	Could not reach target pose
3	PATH_PLANNING_ERROR	No valid motion plan found
4	TORQUE_LIMIT_EXCEEDED	Torque exceeded limit (jam, resistance)
5	COLLISION_RISK	Path blocked or unsafe
6	JOINT_LIMIT_ERROR	Exceeded joint limits
10	OBJ_LOC_FAILED	Object localization failed
11	NO_OBJECT_DETECTED	No object detected
12	VISION_TIMEOUT	Vision module timed out
13	CAMERA_ERROR	Camera or image acquisition failed
20	GRIPPER_FAILED	Gripper failed to open/close
21	TOOL_MISSING	Required tool not attached
22	TOOL_ACTUATION_FAILED	Tool command failed
30	UNSCREW_FAILED	Unscrewing action failed
31	INSERTION_FAILED	Insertion task failed
32	PICK_FAILED	Pick-up failed
33	PLACE_FAILED	Place action failed
40	HARDWARE_COMM_ERROR	Lost communication with hardware
41	INVALID_INPUT	Malformed or invalid command
42	TIMEOUT	Generic operation timeout
43	SAFETY_TRIGGERED	Emergency stop or safety system active
44	TASK_CANCELLED	Task was externally cancelled
50	OBJECT_LOCATION_ERROR	Object location out of allowed area
100	UNKNOWN	Unclassified error

The following text provides a summary of the plugins and behavior trees developed for each respective robot within the CRF and KUKA use cases.

FELICE Robot:

For FELICE Robot following plugins have been defined which integrate different modules to the behaviour tree engine:

1) *FeliceSkillsAddObjectRelatedCollisionScene*: This interacts with the action server handling removal and addition of collision scene, such that respective collision scene for different tool ids could be added to plan a collision free path for scanning the tool.
2) *FeliceSkillsDetectObject*: This interacts with the OBPs action server and retrieves 6D pose and score for object detection.
3) *FeliceSkillsMoveArm*: This interacts with the Felice Robot Arm action server and can make the arm move for joint goal or cartesian goal.
4) *FeliceSkillsNavigate*: This interacts with Felice Robot's Nav2 action server sending navigation goals to the robot.
5) *FeliceSkillsOperateGripper*: This interacts with Felice Robot's gripper action server thus opening, closing or opening the gripper half.
6) *FeliceSkillsPickPlaceAction*: This interacts with the Moveit2 Task Constructor based pipeline enabling planning of pickup and place given the tool and tool stand position.
7) *FeliceSkillsRemoveCollisionScene*: This interacts with the action server handling removal and addition of collision scenes, such that all the collision objects in the scene could be deleted.
8) *KbGetPoiClient*: This interacts with the KB service server communicating with the KB server (http) to retrieve points of interest like tool stand positions etc.
9) *GetPosesFromQueue*: This is a utility node responsible for returning scan poses from a queue of scan poses. This node is useful to move the arm to multiple scanning positions such that tools are detected every time.

Similarly, behaviour trees are provided for all the FELICE Robot Capabilities namely; Pickup, Place, Navigate, Handover, Release, Takeback and Takebackclose.

UR10:

The UR10-specific behaviour tree plugins provide a range of actions for the atomic skills, as well as utility methods for data manipulation and geometric transformations. The implementation is separated into 4 plugins:

1. *Robot Command* executes a URScript program. Different commands can be passed, such as *scan_full*, *scan*, *screw*, and *home*.
2. *Object Detection* triggers a small wrapper of the OBP module.
3. *Job Action* provides options for writing and archiving workflow data.
4. *Pose Transformation* queries the internal scene graph to apply geometric transformations to a robot pose.

The behaviour trees designed using the MRC component are also responsible for handling and propagating errors. For this purpose, every action has an error code output that is set to the corresponding value in case an error occurs.

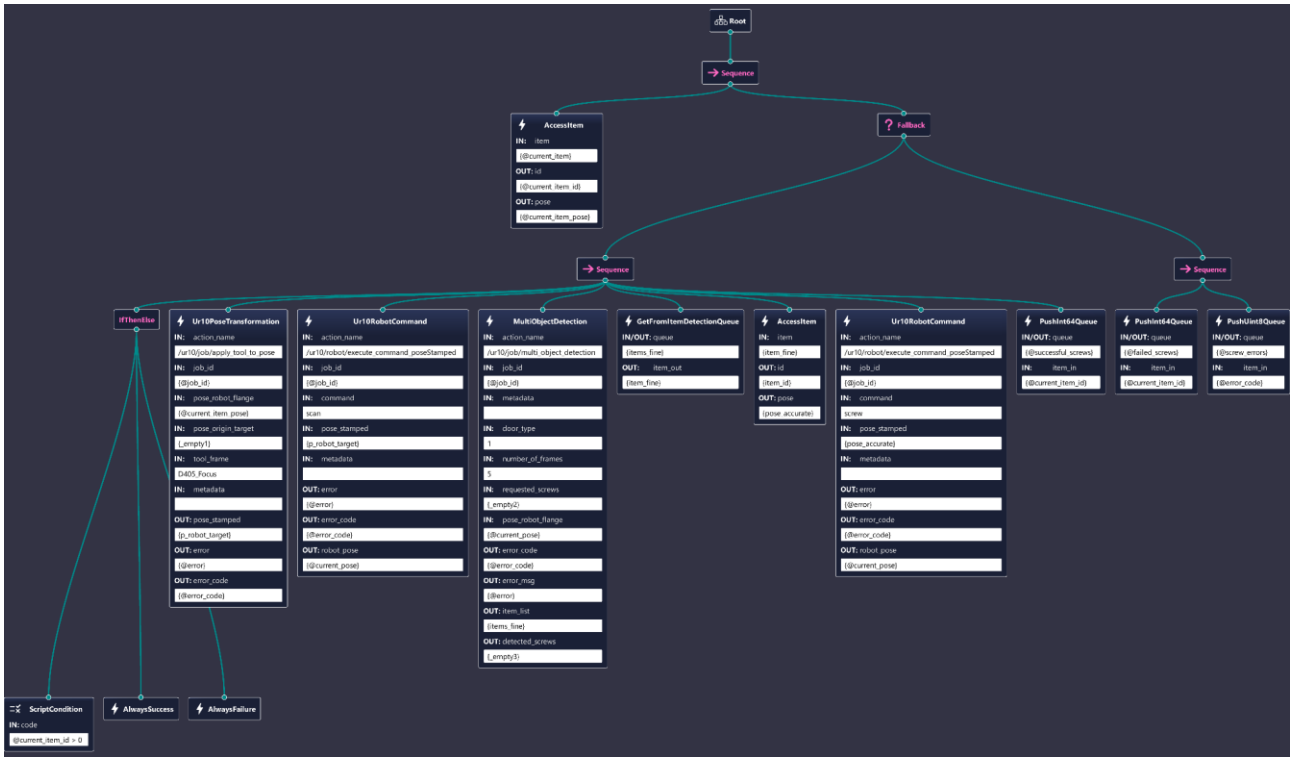


Figure 3. Groot2 window showing unscrew_single behavior tree.

KUKA robot:

For the KUKA Robot following plugins have been developed

1) *CrawlerNavRequest*: This interacts with the KUKA robot crawler's nav2 action server sending navigation goals to the crawler.
2) *KukaRobotDriller*: This interacts with the KUKA robot driller's action server sending drilling goals, where and how many holes to be drilled.

For the KUKA robot the MRC provides behaviour trees namely; navigate_with_recovery, drill_with_recovery, drill_holes (navigate to wall and start drilling), hmi_drilling (navigate to wall and start drilling based upon drilling data saved through HMI)

3 MH-MR TASK ALLOCATION [CTA]

The integration of human-robot collaboration across diverse and complex landscapes necessitates a flexible, dynamic architecture capable of fostering seamless interaction. Within the SOPRANO framework, this challenge is addressed by the Multi-Human-Multi-Robot (MH–MR) Collaborative Task Assignment (CTA) component. This component serves as the primary engine for the coordination of human and robot behaviour. It is designed to link robotic task execution with ongoing human labour, efficiently coordinating human-robot collaboration.

By building on evolving environmental cues extracted by other SOPRANO components and accounting for application requirements, the CTA aims at aligning robotic behaviours with human needs and intentions shaped by the operational context. It acts as a centralized supervisor of human–robot collaboration and the primary decision-maker for robotic actions. The CTA integrates expert knowledge on the operational characteristics and requirements of each use case, translating them into decision-making procedures. This prior knowledge is encoded as action sequences, priorities, constraints, and MH–MR interaction points, tailored to the needs of SOPRANO use cases. Combined with continuous monitoring of the work environment, this enables successful task execution while enhancing the fluency of human–robot collaboration.

The platform optimizes task distribution by evaluating robot availability and operational skills, expected execution times, and known task implementation constraints, ensuring efficient and feasible task execution, in accordance to the evolving operational requirements. The assignment logic is designed for transparency and ease of debugging, structured around three core criteria:

- *Capability Alignment*: Tasks are delegated only to robotic units that possess the specific technical specifications required for the job.
- *Dynamic Adaptation*: The system leverages information from the environment, allowing it to reassign tasks as operational requirements change or as the robot moves to support a different worker.
- *Safety & Constraint Enforcement*: By integrating safety constraints and considering risk alarms, CTA enforces safe task execution, in accordance with the currently observed environment state.

The component is fully configurable, allowing production managers to declare the skills of the operating robots, define complex workflows, and monitor task allocation and execution through an information-rich graphical user interface.

Ultimately, the SOPRANO CTA component establishes a flexible, scalable, and robust yet explainable environment in which multiple humans and robots can collaborate efficiently. Within this framework, CTA is responsible for managing robot actions and is not intended to prescribe or control the behaviour of human workers. Instead, human operators perform their tasks according to established work practices and procedures, while CTA observes and interprets their activity in order to guide robot behaviour accordingly.

Robot actions are therefore planned and adapted to complement human workflows, ensuring that collaboration remains intuitive, non-intrusive, and aligned with existing processes. By dynamically assigning tasks based on agent capabilities and system state, CTA ensures that the appropriate agent performs the appropriate task at the appropriate time, effectively supporting the execution of the defined use-case scenarios.

3.1 TECHNOLOGY OVERVIEW AND COMPONENT ADVANCEMENTS

As summarized in D2.2, the CTA component is divided into two main parts: the Daisy Planner (DP), responsible for high-level decision-making and aligning robot actions with domain-specific job sequencing, and the Human–Robot Interaction module, which handles short-term decision-making to enhance robot responsiveness in human–robot collaboration.

In short, the Daisy Planner organizes work into tasks, each represented by an action-flow that starts and ends in a "Ready" state for the assigned agent. Tasks are depicted as petals within the Daisy Plan, which encodes expert knowledge on action sequencing, prerequisites, and dependencies between parallel tasks executed by different agents, modelled as constraints. At runtime, the planner assigns tasks to collaborating agents based on performance and expected completion time with the aim to improve team productivity. It monitors execution to ensure timely coordination between agents, enforcing synchronization of robot activities with human actions. To support this, the DP operates on a mid-scale time frame in the range of minutes, avoiding the consideration of long task sequences in human–robot interaction. The Daisy Planner is designed to keep all available agents actively engaged and easily adapts to teams of varying sizes and compositions. By focusing on short-term planning and selecting tasks for the next available agent, it flexibly responds to the current environment and overall progress. In SOPRANO, the Daisy Planner is constantly enriched with new features to enhance robustness, usability, and adaptability across use cases, in order to support the monitoring, and coordinated execution of MH–MR scenarios.

The fluent Human–Robot Interaction (fHRI) module operates on a short, dynamic time scale to enhance the quality and efficiency of human–robot collaboration. It grounds high-level commands from the DP module in the current environmental context, ensuring actions are precisely specified and executable. Using real-time data from SOPRANO sensors and software, fHRI continuously monitors robot actions alongside human activity and environmental changes. It enables smooth interaction and introduces on-the-fly adjustments, such as adapting movements to improve human ergonomics, or increase collaboration fluency.

The CTA component outputs commands to the RCI component for execution and shares updates via the Kafka context broker. It also receives feedback from RCI on execution issues and can propose dynamic adjustments, such as repositioning or modifying movements. Once actions are completed, RCI notifies the CTA modules, allowing the system to proceed with the next planned step.

CTA Architectural design advancement. The Daisy Planner (DP) and the fHRI modules exhibit distinct operational characteristics. The DP implements the core functionalities for coordinating multi-agent collaboration, relying on a task-scheduling logic that is generic and can be abstracted from the specific requirements of individual use cases. In contrast, the fHRI module is inherently concerned with application-specific details that directly influence the situated fluency of human–robot interaction.

Despite these differences, in the previous reporting period the implementations of DP and fHRI remained largely coupled. This resulted in multiple full-scale versions of the CTA, each tailored to a specific application domain. Such an approach limited reusability and reduced the potential for adoption by third parties beyond the SOPRANO project.

To address these limitations and support deployment across diverse application domains, the architecture has been revised to explicitly separate DP from fHRI. The DP now retains responsibility

for general-purpose task scheduling and decision-making related to the triggering of primitive robot actions. In parallel, application-specific functionalities are encapsulated within the fHRI and implemented as a library of independent ROS2 action services (see below). These services can be flexibly integrated into the task execution flow through the CTA configuration file.

This architectural decoupling improves modularity, enhances maintainability, and significantly simplifies integration for third-party developers, particularly those involved in SOPRANO open-call projects, by enabling the reuse of the core CTA functionality without requiring modifications to application-specific components.

3.2 METHODOLOGY AND IMPLEMENTATION

During the second SOPRANO reporting period, the implementation and execution of experiments across the different use-case scenarios provided valuable insights into the operational behavior of the CTA component under realistic conditions. These activities not only validated core functionalities but also revealed opportunities for improvement, particularly in terms of adaptability, responsiveness, and coordination efficiency within human–robot teams. As a result, they guided a series of refinements in the functional characteristics of the CTA, aiming to better align its performance with the evolving requirements of real-world deployments.

At the same time, the continuous evolution of the broader SOPRANO ecosystem, driven by the parallel development and maturation of its individual components, introduced additional complexity in system integration. Changes in communication protocols and processing capabilities across components necessitated ongoing adjustments to the interfaces and internal workflows of the CTA. Ensuring seamless interoperability required careful synchronization with updated component specifications, as well as iterative redesign of interaction mechanisms to maintain consistency in information exchange and robustness in decision-making.

These two combined drivers (i) empirical feedback from experimental validation and (ii) the co-evolution of system components, led to a series of targeted enhancements in the CTA. In the following paragraphs, we present the main new features introduced during the last reporting period, highlighting how they address identified challenges and contribute to improved coordination, flexibility, and overall system performance.

3.2.1 Action Server Interfacing

Following the adoption of the ROS2 framework, SOPRANO transitioned to an action server–based communication paradigm with two primary objectives: (i) improving the control and flexibility of robot actions, and (ii) facilitating the reuse and integration of system components by third parties, by separating general purpose task scheduling from use-case specific customizations that can be included in action servers. Additionally, compared to previous communication mechanisms, the action server model enables non-blocking execution, supports runtime cancellation of actions, and provides continuous feedback during task execution. These features are important for long-running or interactive processes, where responsiveness and adaptability to changing conditions are critical.

During the previous reporting period, the implementation of the CTA relied on dedicated bridge software for each individual robot behaviour. While functional, this design introduced significant overhead in terms of development, maintenance, and scalability, as each new behaviour required the

creation and management of a corresponding communication interface that had to be use case specific. To address these limitations, the CTA in collaboration with RCI introduced a unified action definition that standardizes communication across all robotic platforms and behaviours. As a result, instead of maintaining multiple use-case specific communication bridges, all robots now utilize instances of a common action interface. In this updated design, action parameters are transmitted via a flexible, stringified JSON attribute, enabling dynamic specification of behaviour-specific inputs.

This approach standardizes communication across all action servers, resulting in a consistent protocol that can be readily adapted to different use cases and robots. Additionally, it facilitates separating mechanisms focused on task scheduling from those focused on enhancing fluent Human-Robot Interaction. The latter can be now implemented as supporting action services that can be flexibly integrated in workflows to enhance responsiveness in robot operation.

The interacting ROS2 action servers are specified in the CTA initialization file that is used for the configuration of the planner in accordance with the application requirements. This abstraction significantly reduces system complexity and improves extensibility, as new behaviours can be integrated without modifying the underlying communication structure. Clearly, this evolution enhances modularity, reduces integration effort, and improves system efficiency and scalability, making the CTA more robust and easier to adopt in both internal and external development contexts.

3.2.2 3.2.2 Multi-level task sequencing

Although the CTA was originally designed to maintain full control over robot action sequencing and explicitly orchestrate primitive actions to guide human–robot collaboration, in practice, use cases within SOPRANO rely on robots that already provide pre-implemented behaviour modules. Adapting or re-engineering these existing functionalities to conform to CTA's control model would require significant effort and lead to duplication of already available skills.

To address this, the CTA has been extended to support two complementary modes of operation. In the first mode, CTA retains full control by managing detailed behaviour descriptions, where sequences of parameterized primitive actions are triggered step-by-step, with execution results continuously reported to the Kafka context broker. This mode is suitable for scenarios requiring fine-grained coordination and tight human–robot interaction. In the second mode, CTA supports higher-level, abstracted behaviours encapsulating multiple primitive actions executed internally by the robot. In this case, CTA does not directly trigger each low-level action but instead interfaces with the higher-level behaviour, ensuring that intermediate feedback and execution results are still communicated to the Kafka context broker. This guarantees system-wide visibility and consistency, even when control is partially delegated. In both modes, CTA acts as a middleware layer that provides monitoring and supervision capabilities. It enables users to track task execution in real time and, where necessary, intervene by parameterizing the ongoing process.

This dual-mode operation broadens the applicability of the CTA, allowing seamless integration with existing robotic systems while preserving the ability to support detailed, tightly coordinated behaviours when required. As a result, CTA can be deployed more efficiently across diverse environments without imposing constraints on underlying robot implementations.

3.2.3 Interaction with the end-user interface

Efficient human–robot collaboration requires that the user remains continuously informed about robot actions and is able to intervene when suboptimal decisions are made or unexpected events occur. To address this, the CTA establishes explicit two-way communication with the SOPRANO user interface (UI), supporting both system-to-user feedback and user-to-system control.

On the system-to-user side, CTA continuously publishes execution updates to the UI, including the status of the current action, intermediate progress feedback, and the next planned actions. This ensures full transparency of the workflow and allows the user to maintain situational awareness throughout task execution.

On the user-to-system side, CTA accepts commands from the UI via a button-based interaction interface. These inputs are processed in real time and can influence the ongoing human-robot collaboration scenario, e.g. by directly triggering specific robot actions. This capability is particularly important for handling errors, resolving unexpected situations, or adapting execution to operator preferences.

To support this bidirectional interaction in a flexible and reusable way, CTA provides a library of use-case–agnostic “special actions” implemented as action servers, which can be embedded into workflows during configuration. These actions act as standardized interaction points between the user and the system, without requiring modifications to the core execution logic. Examples include pausing robot execution, introducing delays, triggering the direct execution of a primitive action, setting parameters for action execution and enabling conditional branching based on environment states defined by the production manager.

By explicitly supporting two-way communication with the UI, CTA improves system transparency and responsiveness, enhances operator control, while maintaining flexibility across different use cases. This design significantly improves usability and facilitates the integration and reuse of SOPRANO technologies in diverse application environments.

3.2.4 Grouping, prioritizing, and resuming workflows

The operational context of the SOPRANO use cases involves tasks that are executed repeatedly throughout a work shift. To support this cyclic operation, CTA introduces mechanisms for reactivating previously executed work sequences, represented as petals in the execution model. In practice, reactivation enables smooth transitions between consecutive task cycles while preserving the logical organization of the work plan, allowing efficient continuation over extended periods.

In addition, complex behaviours are often decomposed into sequences of petals executed in a predefined order. This creates the need for explicit grouping and prioritization mechanisms, allowing related petals to be managed as cohesive units and ensuring that their reactivation follows a meaningful and efficient execution order. Prioritization enables the system to determine which tasks should be resumed first based on operational criteria (e.g., workflow dependencies), while grouping ensures that logically connected actions are consistently executed together during each cycle. The combination of these mechanisms allows CTA to support structured repetition of collaborative workflows, maintaining continuity in human–robot interaction and improving overall system efficiency.

For the given complexity of the scenarios currently addressed in SOPRANO, the present implementation focuses on providing sufficient functionality to ensure reliable execution of the targeted use cases. However, the underlying design remains extensible. Deployment in more complex environments or application domains may introduce additional requirements, such as dynamic reprioritization, looping constraints between multiple agents or more sophisticated dependency management. These aspects remain out of the scope of the current SOPRANO implementation, but can be incorporated in future extensions of the CTA as needed.

3.2.5 Dynamic Multi-Agent Coordination and Role Adaptation

CTA and in particular the Daisy Planner, has been designed from the outset to support collaboration among a variable number of agents. This flexibility is achieved through a dynamic task allocation mechanism, where tasks are assigned to agents based on their current availability and capability set. Specifically, task–agent pairing is performed by matching the skill requirements of each task with the action primitives supported by each agent. This enables CTA to distribute work in a way that maximizes overall team performance while adapting to changes in team composition over time.

Beyond repeating collaboration scenarios, SOPRANO also addresses cases in which robots may change roles during a work shift. Role switching can be manually triggered through external commands (e.g., via a user interface), allowing a robot to transition between different task domains or collaboration contexts. To ensure consistency and safety, such transitions are not immediate: the robot must first complete its current task and reach a stable state before switching roles. This includes verifying that no residual constraints exist (e.g., attached tools or task-specific configurations) that could interfere with the old and new role. Once these conditions are satisfied, the robot can safely assume new responsibilities, subject to the constraints of the new workflow.

In the first reporting period of SOPRANO, each robot–human pair was managed by a separate CTA instance. However, the introduction of dynamic role switching and the requirement for robots to collaborate with multiple operators led to a shift toward a unified CTA architecture. In the current approach, a single CTA instance manages the entire shop-floor, coordinating multiple robots whose roles, task assignments, and proximity to each other and human collaborators may vary over time.

To support this level of flexibility, CTA incorporates mechanisms that allow production managers to dynamically define robot responsibilities by activating or deactivating corresponding workflows. These workflows are designed with role transitions in mind, ensuring compatibility and continuity across different operational modes. Furthermore, role switching is coupled with explicit role-initialization procedures, including adjustments to the robot's physical position and internal state, to guarantee safe and predictable behaviour in demanding industrial environments. This centralized shop-floor coordination improves resource utilization and ensures consistent decision-making.

3.2.6 Information rich back-end interface

To facilitate experimentation and enable systematic testing and validation of both individual robot actions and complete workflows, the CTA is coupled with a dedicated back-end user interface. Upon initialization, the interface verifies communication between the CTA and the Kafka context broker, which serves as the primary integration layer for components operating outside the ROS2 ecosystem. This allows users to confirm connectivity, monitor data exchange, and diagnose potential communication issues, supporting debugging and system validation activities.

The back-end interface provides the experimenter with a comprehensive overview of robot tasks and responsibilities, visualized through the Daisy Plan (see Figure 4). In this representation, tasks are organized as petals, allowing users to clearly identify the current execution state, including actions that have already been completed and those that remain pending for each robot.

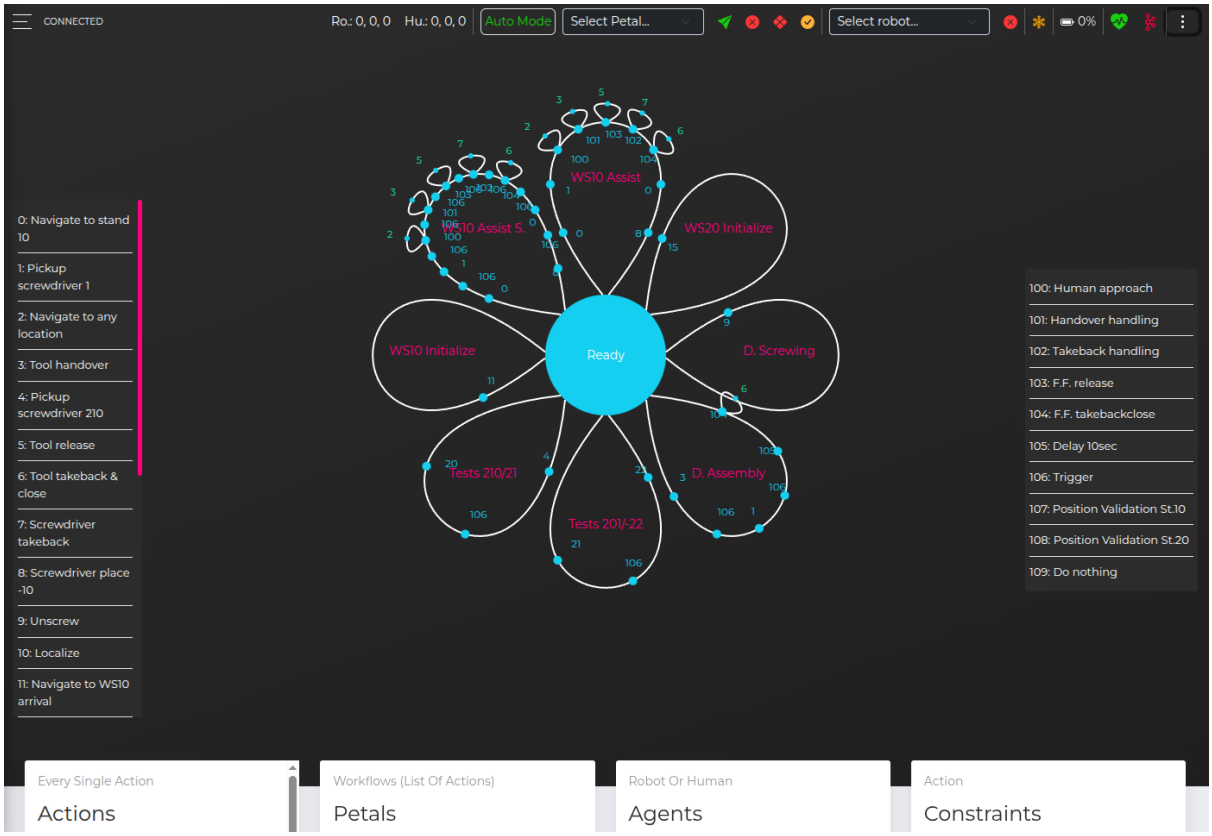


Figure 4. A Daisy Plan of robot tasks, where each petal represents a sequence of primitive actions. Smaller (nested) petals connected to specific primitive actions denote special actions triggered within the workflow.

In addition to workflow visualization, the back-end interface supports the monitoring of critical sensory data and inferred system states (see Figure 5). These include inputs from SOPRANO perception modules, providing a coherent and up-to-date representation of the operational environment. In particular, the interface enables the visualization of 2D maps in which the positions of human operators and collaborating robots are dynamically displayed. This functionality enhances situational awareness and allows experimenters to better understand and evaluate system behaviour during execution.

The interface is designed to accelerate development and testing cycles by allowing experimenters (the developers of SOPRANO components) to load preconfigured behaviours associated with known action servers. Through this mechanism, experimenters can manually trigger complete workflows, sub-workflows, or individual primitive actions, all aligned with the use-case scenarios defined in the project. This capability supports rapid iteration and repeatability in testing, without requiring modifications at the system level.

In addition, the back-end interface is compatible with any action server that adheres to the SOPRANO message schema, ensuring extensibility and interoperability with newly integrated components. It

provides the experimenter with detailed feedback on execution, including real-time progress updates and final action outcomes (e.g., success or failure), presented through lightweight notification mechanisms such as toast messages.

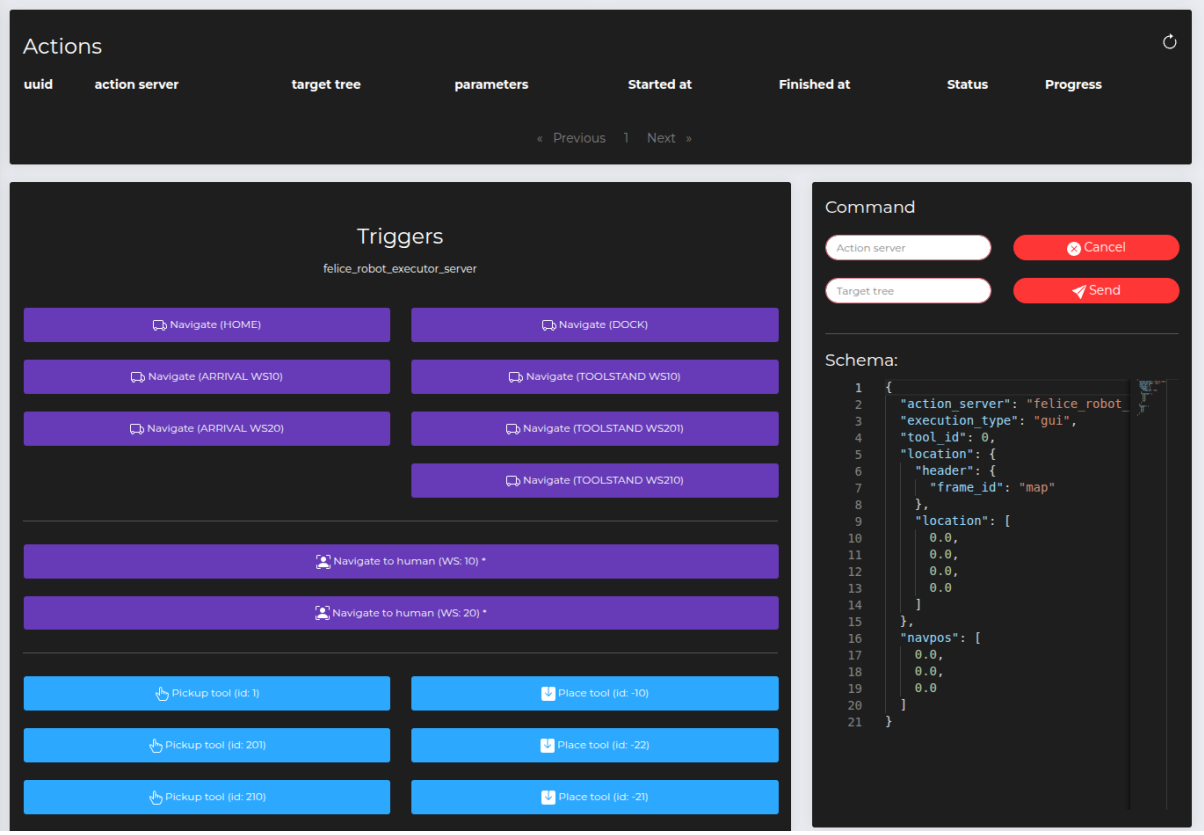


Figure 5. The backend-interface enables the direct trigger of petals (workflows) and primitive robot actions to facilitate experimentation.

4 MRC & CTA AVAILABLE FOR DEPLOYMENT BY THIRD PARTIES

The CTA and MRC components have been packaged and made available for download and deployment by the ten SOPRANO subprojects funded through the project's open call. This initiative aims to support SMEs and start-ups in leveraging SOPRANO technologies to develop new demonstrators or enhance existing ones, with a particular focus on Multi-Human Multi-Robot (MH–MR) collaboration scenarios across a range of industrial environments and sectors beyond those directly addressed by the core consortium. This external adoption serves as a strong validation of the components' robustness, scalability, and reusability, reinforcing their potential to support MH–MR applications across diverse industrial domains.

To facilitate deployment, the components are provided with standardized interfaces, documentation, and configuration mechanisms, enabling third-party developers to integrate them into their own systems with minimal effort. This includes support for modular deployment, interoperability with external components, and compatibility with diverse robotic platforms and application requirements.

The successful uptake of CTA and MRC by multiple independent subprojects demonstrates their flexibility and generality, confirming that the underlying design can accommodate a wide variety of use cases. It also highlights their capability to operate in heterogeneous environments, supporting different collaboration patterns, task structures, and operational constraints.

5 MRC & CTA DEPLOYMENT IN THE CRF USE CASE

5.1 REQUIREMENTS

The application of MRC and CTA to the use cases examined in SOPRANO required adaptations, which could be implemented efficiently due to the improved modular architecture and the extended use of ROS2 action services. The enhanced BT-based execution framework of the MRC provides a combination of adaptability and robustness enabling better handling of dynamic environmental changes. At the same time, the decoupling of the DP from the fHRI module, combined with the adoption of standardized communication interfaces, allowed application-specific functionalities to be easily integrated with the core CTA logic. As a result, adapting CTA to different operational environments primarily involved configuring workflows and linking existing or newly developed action services, rather than modifying the underlying system architecture.

Within this context, MRC and CTA were successfully applied to the CRF use case, supporting human–robot collaboration in automotive manufacturing environments, particularly for door disassembly processes. In this scenario, CTA coordinates the actions of two robots controlled by MRC to complement human operators performing disassembly tasks, ensuring efficient task allocation and synchronization. Similarly, MRC and CTA were deployed in the KUKA use case, addressing human–robot collaboration in residential building renovation applications.

In contrast, the application of the MRC and CTA components to the symbiotic farming use case was not pursued. In particular, during approximately the first half of the project, the initial plan was to deploy the CTA component within the E-TERRY platform, and an early version of the CTA tailored to the symbiotic farming use case was developed.

However, the E-TERRY platform already incorporates a mature and fully operational task scheduling system that is well aligned with its domain-specific operational requirements. Replacing this existing solution with the MRC and CTA components would have required substantial integration and adaptation effort on the end-user side, including modifications to established workflows, interfaces, and operational procedures, while offering only limited added value in practice. Consequently, maintaining the existing scheduling framework was considered the most efficient, pragmatic, and technically appropriate approach for this application domain.

5.2 IMPLEMENTATION

In the CRF environment, scenarios involving two workstations, in particular WS10 and WS20, were examined to evaluate different modes of human–robot collaboration. In WS10, the focus is on the use of the mobile robotic platform FELICE, which retrieves tools from a tool-stand and delivers them to the worker to support the disassembly process. This setup allows the robot to act as an active assistant, reducing the need for the worker to move away from the task area. The same platform can also be deployed in WS20, where two different tools must be provided to the worker. In WS20, a UR10 robot is additionally installed to autonomously perform part of the disassembly process, creating a more complex collaborative environment involving both mobile and stationary robotic systems.

Deployment in WS10. Within WS10, multiple variations of two core scenarios were explored to better understand alternative tool provisioning strategies. In the first scenario, the robot retrieves a tool from the tool stand, navigates to the worker, and hands it over. It then waits until the worker

completes the task, takes the tool back, and returns it to the tool stand. This sequence is repeated for each disassembly cycle. In the second scenario, the initial steps remain the same; however, after the worker returns the tool, the robot does not place it back on the tool stand. Instead, it retains the tool and remains positioned near the worker, allowing for quicker re-delivery when the same tool is required again in subsequent cycles (e.g., during the disassembly of the next door). This approach reduces unnecessary robot motion at the expense of keeping the robot engaged in close proximity to the human.

These two scenarios correspond to distinct workflows, each defined by a different sequence of looping BT-powered primitive actions made available by MRC. Using the CTA back-end, both workflows were systematically evaluated to assess how effectively they align with the human operator's actions and the overall task flow. This evaluation was not limited to high-level workflow validation, but also enabled real-world debugging and fine-tuning of critical robot configuration parameters that directly affect performance, safety, and user experience.

In particular, several key aspects were examined. First, the definition of the robot's navigation arrival polygon near the tool stand was tested to ensure reliable and efficient tool picking, minimizing positioning errors while maintaining motion accuracy and picking success. Second, temporal thresholds for the validity ("freshness") of human-reported location data were evaluated, allowing the system to ignore outdated estimates and rely only on recent, accurate information for decision-making. This proved important for maintaining robust human–robot coordination in dynamic conditions where the worker moves around the door cart.

Additionally, the proximity of the robot's navigation goal relative to the human operator was tuned. The goal was to identify a distance that enables efficient tool handover without compromising the worker's sense of safety or comfort, which is a critical factor for acceptance in collaborative environments. Finally, the orientation and height of the robot's arm during the handover action were adjusted to improve human ergonomics, ensuring that tool exchange occurs in a natural and convenient posture for the worker.

Particular attention was given to the identification and design of appropriate interaction points with the end-user interface, with the goal of ensuring that workers receive timely, relevant, and actionable information during task execution. This includes not only clear visibility of the robot's current and planned actions, but also mechanisms that allow the user to effectively intervene when necessary, without disrupting the overall workflow.

A key aspect of this effort concerns the implementation of user control features, such as the ability to pause and resume robot workflows. The definition of these mechanisms has been the subject of focused discussion across the AVT, CTA, and MRC components, as it directly impacts the balance between system autonomy and human control. Different strategies have been examined to determine how best to support intuitive and safe human intervention, while preserving efficiency and coordination in collaborative tasks.

In parallel, experimentation activities have been used to evaluate how these interaction mechanisms influence task performance and user experience, particularly in scenarios involving robot-to-human tool provisioning. The insights gained from these experiments are being used to assess alternative interaction strategies and refine system behaviour. The final selection of the most appropriate approaches will be consolidated during the final phase of the SOPRANO project.

Deployment in WS20. The tests conducted in WS20 can be seen as an extension of those performed in WS10, incorporating additional complexity in terms of both task scope and system configuration. In addition to the previously evaluated scenario of fetching and handing over tools to a human operator, WS20 includes the handling of two distinct tools and integrates the functionality of a UR10 robotic arm operating within the same workstation.

Building on the WS10 experiments, a series of tests were carried out to evaluate the ability of the FELICE robot to reliably detect, grasp, and deliver two additional tools required for the task. In parallel, the UR10 robot was deployed to perform unscrewing operations on the door components targeted for disassembly. This division of responsibilities allowed the robots to offload repetitive tasks from the human worker, thereby reducing workload and improving overall efficiency.

The simultaneous operation of three agents (human, FELICE robot, and UR10 robot) within a shared workspace introduced significant coordination challenges. These included the need to synchronize task execution, manage spatial interactions, and prevent conflicts or unsafe situations arising from overlapping work areas or unintended robot motions. Ensuring safe operation required careful orchestration of actions, as well as continuous monitoring of the state of all agents.

Given the demanding nature of the work environment, where human operators may experience physical and cognitive fatigue over extended shifts, particular emphasis was placed on safety-aware workflow design. The evaluated scenarios prioritized conservative and predictable robot behaviours, clear task separation when necessary, and mechanisms to allow human intervention. This approach aimed to maintain a high level of safety while still achieving the desired gains in efficiency and workload reduction.

So far, communication with the user interface in WS20 has primarily focused on providing the human operator with clear and timely feedback on robot activities. In particular, the results of the UR10 operations are systematically reported to the AVT component, where each successfully unscrewed screw is visually represented in the interface. This allows the user to track task progress and maintain awareness of the disassembly status. Potential errors are transparently communicated to the operator, so problems can be resolved in a timely manner in case the system is not able to recover autonomously. In parallel, the FELICE robot's ongoing and upcoming tasks are also communicated to the user, offering visibility into both current execution and short-term planning. This system-to-user communication ensures transparency and supports informed supervision of the collaborative process. However, user-to-system interaction capabilities are currently under development. Specifically, the integration of SOPRANO components has yet to be fully configured to accept direct user commands for triggering operations, such as requesting the unscrewing of a specific screw; the action primitive is variable and remains to be linked with the user interface. Enabling this functionality will allow the human operator to intervene more actively in the workflow, providing greater flexibility in handling exceptions, prioritizing specific actions, or adapting execution to real-time needs.

Dynamic Multi-Workstation Robot Allocation. A particularly noteworthy feature of the SOPRANO approach is the use of a single mobile robot, namely the FELICE platform, to support multiple workstations by dynamically switching between them. This capability allows the robot to be allocated where it is most needed, either upon direct worker request or based on decisions made by the production manager, thereby improving overall resource utilization on the shop-floor.

To ensure safe and seamless transitions between workstations, robot movement is tightly coordinated with the operational cycles of each station. Specifically, human–robot collaboration at a given workstation is aligned with the start and completion of a disassembly cycle. The robot remains assigned to a workstation for the duration of one or more consecutive cycles and only initiates relocation after completing its current tasks and reaching a stable state. This approach avoids interruptions, prevents partial task execution, and minimizes interference with ongoing human activities.

The switching mechanism is designed to maintain continuity of operations across both workstations while respecting safety constraints, such as shared workspace occupancy and operator presence. As a result, robot navigation and task allocation are performed in a controlled and predictable manner, reducing the risk of disruption in a busy industrial environment.

Coordination across multiple workstations is managed by a single CTA instance, which maintains a global view of task execution and resource availability. This centralized coordination enables efficient scheduling and dispatching of the FELICE and UR10 robots, ensuring that their actions consistently complement human activities across different locations. By dynamically adapting to changing demands, the system supports balanced workload distribution and enhances the overall efficiency of the production process.

Overall, for the FELICE Robot, the integration of MRC and CTA has been successfully tested for WS10 and WS20 and also evaluated for WS10 in deliverable D5.3. All the capabilities developed for the FELICE robot and exposed by MRC-RCI are tested in the real world and integrated with CTA to support collaborative human-robot car door disassembly.

Additionally, the integration of MRC and CTA for the UR10 robot in its latest protocol revision was finished in Q1 2026. Current tests show that CTA reliably receives detailed results and feedback after each processing step. Further tests including the whole SOPRANO module stack are in progress.

6 MRC & CTA DEPLOYMENT IN THE KUKA USE CASE

6.1 REQUIREMENTS

The deployment of the CTA and MRC components in the KUKA use case requires adapting their functionality to the specific characteristics of a robot-supported renovation scenario. In contrast to the CRF use case where the disassembly cycle is predefined and repeated throughout the shift, the KUKA scenario is dynamic, with tasks varying depending on the specific requirements of each room.

These requirements are defined through a Building Information Model (BIM) file, which encodes the renovation needs of the environment. As a result, the system must support:

- Dynamic definition of tasks at runtime, rather than relying on preconfigured workflows.
- Interpretation of externally provided job specifications.
- Flexible grouping and execution of tasks based on spatial and operational constraints (e.g., per wall) .

This introduces the need for CTA to generate workflows dynamically and adapt robot behaviour based on incoming data, rather than static configuration. For the MRC there is a need to break down the high-level skill of drilling holes into navigation to the wall and then going step by step over the holes to be drilled. The plugins defined by MRC then communicate with the low-level applications like Nav2 and KUKA robot driller.

6.2 IMPLEMENTATION

To address the above requirements, the BIM data is processed by the MH-MR Architecting Tools (MAT) SOPRANO component, which extracts the jobs assigned to the KUKA robot and formats them into a structured JSON message. This message is then published to the Kafka context broker.

CTA consumes this message and dynamically generates the corresponding workflow. In this setup, the Daisy Planner is initialized not through a static configuration file, but through the received JSON input, representing an extension of its original functionality toward runtime workflow generation.

The extracted jobs are grouped based on their spatial context (i.e., per wall) and translated into pairs of navigation and drilling targets. These are then passed to the MRC component, which executes the corresponding actions using the KUKA robotic drilling platform. The results of each robot action are reported back to the CTA and subsequently forwarded to the AVT, where they are used to update the user interface and inform the operator about the progress of the drilling tasks.

At the current stage, CTA and MRC have been deployed and tested independently using simulated versions of the interacting SOPRANO components. Full integration with all system components and real robot execution is planned for May 2026.

7 CONCLUSIONS AND FUTURE WORK

During the current reporting period, significant progress has been achieved in advancing the maturity, robustness, and integration of the CTA and MRC components within the SOPRANO framework. Both components have reached Technology Readiness Level (TRL) 6, having been successfully validated in relevant industrial environments through their deployment in real-world use cases, including the CRF scenarios and the pending KUKA full scale scenario assessment. These validations demonstrate that the developed solutions are not only technically sound but also capable of supporting practical human–robot collaboration workflows under realistic operational conditions.

A key achievement concerns the tight integration and interaction between CTA and MRC, which has been substantially improved compared to previous iterations. The communication based on ROS2 Action Services has been refined to support reliable, non-blocking execution of robot actions, continuous feedback exchange, and robust handling of execution outcomes, including failure and recovery cases. This interaction has been extensively tested across collaborative scenarios and multiple robotic platforms, confirming its stability, flexibility, and suitability for coordinating complex task execution pipelines involving both mobile and stationary robots.

In addition to the internal integration, interoperability with other SOPRANO components has been successfully established and validated. CTA and MRC effectively interact with the broader ecosystem through both ROS2-based communication and the Kafka context broker, which serves as the main integration backbone for cross-component data exchange. This enables seamless information flow between perception, planning, execution, and user interface modules, ensuring that decisions are informed by up-to-date contextual, sensory, and user-related information. The ability to operate within this distributed architecture highlights the modularity and scalability of the developed components.

Looking ahead, future work will focus on the systematic evaluation of Multi-Human Multi-Robot (MH–MR) collaboration scenarios under increasingly complex and realistic conditions. These activities will aim to identify application-specific corner cases, such as rare failure modes, edge-case interactions, or unexpected environmental dynamics, which may not have been fully captured in the current validation phase. Insights gained from these experiments will guide targeted refinements of both CTA and MRC, particularly in areas related to coordination strategies, human–robot interaction fluency, and adaptive behaviour under uncertainty.

Overall, the progress achieved so far establishes a solid foundation for the final stages of the SOPRANO project, where emphasis will be placed on consolidating system performance, enhancing usability, and ensuring the reliability of the developed technologies across diverse industrial applications.

8 UPDATES SINCE D2.2

This section presents the main advancements of the components covered in the current deliverable relative to their earlier versions reported in D2.2, highlighting the most significant development efforts made to enhance functionality, robustness, and integration readiness.

8.1 MAPPING OF HIGH-LEVEL PLANS TO ROBOTIC CAPABILITIES [MRC]

- Mapping of High-Level Plans to Robotic Capabilities (MRC) component was advanced from an initial design-time concept towards an integrated and use-case validated task execution component.
- The work focused on translating high-level task descriptions into executable robot behaviours using Behavior Trees, ROS2 action/service interfaces, and component plugins that bridge planning and execution. This enabled MRC to act as the design layer for robot behaviours while RCI executed these behaviours at runtime through the Dynamic Behavior Tree Engine
- A layered architecture has been created for MRC which allows efficient use of the component.
- For all the robot types namely Felice Robot, Kuka robot and UR10 the plugins and behavior trees were defined.
- The validation of MRC and interaction with CTA has been also validated.

8.2 MH-MR TASK ALLOCATION [CTA]

- Redesigned the CTA architecture to support a modular and use-case agnostic DP task scheduling framework. Introduced ROS2 action-service interfaces enabling seamless integration of fHRI customizations for use case specific workflows and heterogeneous deployment scenarios. Completed ROS2 integration and containerization of CTA using Docker to facilitate portability and reproducible deployments.
- Optimized the DP backbone to improve multi-agent coordination in collaborative operational environments. Optimizations focused on task allocation, constraint-based agent synchronization, and looping tasks.
- Implemented centralized coordination capabilities that allow a single CTA instance to manage and supervise multiple robots. The solution supports synchronized task execution, conflict avoidance, and coordinated mission planning.
- Upgraded the CTA back-end GUI to support dedicated interfaces tailored to the individual use cases. Improvements included enhanced information visualization, operator usability, decision-support mechanisms, and streamlined execution workflows.
- Implemented bidirectional Kafka-based communication mechanisms to ensure reliable data exchange between the CTA and other SOPRANO components. Defined and integrated use-case messaging structures aligned with AVT operational requirements.
- Extended the CTA framework to support AVT-driven workflow specification and initialization, without requiring manual reconfiguration.
- Developed ROS2-based interfaces that allow CTA to monitor subtasks initiated and managed externally from the DP. Integrated robot-state updates into the Kafka context broker to maintain consistent state awareness across the SOPRANO system.

- Customized the CTA framework for the CRF and Kuka use cases, including the adaptation of operational scenarios, workflows, and integration testing activities (deployment activities for the e-Terry platform were initiated; however, the final deployment phase was cancelled to optimize resource allocation).