L4MS

Smart logistics for manufacturing
Table of Contents

1  INTRODUCTION .................................................................................................................. 3
2  GENERIC COMPONENTS ...................................................................................................... 4
3  INTERFACES .......................................................................................................................... 7
   3.1  Interface with non-ROS-based AGVs .............................................................................. 7
   3.2  Interface with ROS-based AGVs .................................................................................... 7
1 Introduction

The OPIL framework consists of several software components that can be grouped into three main groups: the generic components, the interfaces, and the case-specific ones.

- **Generic components**: this group contains the components responsible for management of the whole logistic process. They can be successfully used regardless of the actual scope of the use case.

- **Interfaces**: this group contains both the OPIL middleware, which is essential for the communication between the components of the framework, and the interfaces connecting the framework to other systems e.g. the Legacy HW nodes or ROS components.

- **Case-specific components**: these provide functionalities required by concrete applications. The case-specific components may be adapted and used in different scenarios if they fit the requirements. Although reusing the existing software is strongly recommended, new components can be developed and integrated by third-parties, as long as they are connected to the messaging middleware of OPIL.

The core, obligatory element of each OPIL deployment is the middleware. This is necessary to provide communication capabilities for the heterogeneous components of the framework.
2 Generic components

The generic components are distributed in a three-layer architecture as depicted in Figure 1:

**Layer 1 (IoT Nodes Layer):** Agent nodes of layer 1 are the components of the architecture that interact with the physical world. For instance, they can interact by sensing, e.g., Sensor Agent node, by acting, e.g., Robotic Agent nodes, or by interfacing with humans, e.g., Human Agent nodes. The OPIL modules of which these nodes are made of, can interact with layer 3 by exchanging messages with Layer 2 and they either directly operate on these messages or translate them to an appropriate format for internal use through their Communication/Messaging sub-modules.

**Layer 2 (Cyber Physical Middleware layer):** IoT nodes at layer 1 can talk to each other and with the other components of the OPIL, as well as with Enterprise Applications, by means of the Cyber Physical Middleware layer provided by layer 2. This communication is enabled by specific modules of this layer such as the Pub/Sub Context Broker or the Enterprise Service Bus (ESB), depending on the cases, through the network infrastructure. For example, in the case of Layer 1, Communication/Messaging sub-modules are tasked with translating toward the
Layer 2 Pub/Sub Context Broker which has the goal of dispatching information to the other OPIL actors involved by the specific scenario. On the other hand, Layer 3 can also interact with Layer 1 throw Pub/Sub Context Broker or be integrated with Enterprise applications through the ESB component of Layer 2.

Layer 3 (Software Systems layer): pure software applications interact with the layer 1 IoT nodes, as well as with Enterprise Applications, by means of message exchange via the Cyber Physical Middleware (Layer 2).

The intermediate layer, i.e., layer 2, in the OPIL architecture has a twofold role, it decouples world-interacting components from pure software components from enterprise applications and it allows for components interoperability. Indeed, with reference to the OPIL architecture, orange blocks are the OPIL “lingua franca” which enable the different components to talk each other (also referred as “the glue”). Indeed by means of the OPIL “lingua franca” the blue blocks in the OPIL architecture and the case-specific components, can talk to each other and can be composed into full-fledged logistic system in a manufacturing environment.

The software components of the different layers are explained hereinafter.

Layer 3 OPIL Components and Functional blocks are:

- **Task Planner**: the functional block of the task planner component is in charge of: (i) deciding and optimizing the tasks to be dispatched to the different agents in the OPIL architecture by means of a Business Process Optimization functional block; (ii) plan the motion tasks for the robot agents in the OPIL architecture by means of the Motion Task Planning; (iii) monitor the execution of the task dispatched to the agents by means of the Task Supervisor.

- **Advanced HMI**: This OPIL node consists of three different sub-modules. The Task Monitoring and Control enables continuous monitoring and control of tasks. Subscription and visualisation of information available in OPIL, and the controllability of the operations, tasks and other actions planned by OPIL and human actors are the key functionalities. The Task Parameterization collects and parameterizes data collected from Enterprise Applications. The Task Specification receives task related information from Task Monitoring and Control, and task specific parameters from the Task Parameterization. The Task Specification formulates a task based on the task parameters

- **Sensing and Perception**: the functional blocks in the sensing and perception component allow OPIL to provide information suitable for safe and accurate motion planning to the actors of an OPIL system, i.e., the Robot Agent Nodes or the Motion Task Planning. The Localization functional block provides 2D pose and/or 3D pose estimates for the agents and for all the items, e.g., goods and products, in the manufacturing system. The Mapping functional block provides information about the structure of the manufacturing shop-floor for the components involved in navigation

Layer 2 OPIL Components are:

- **Context Management**: this component enables the exchange of context information among each OPIL components (as well as with the external ones such as legacy enterprise applications and 3D simulators) through a Publish & Subscribe pattern and brokerage of contexts. On the other hand, this module provides capabilities to analyse event data in real-time thus enabling instant response to changing conditions. It is based on two main sub-modules, the first one is the Context Broker which enables each OPIL-based system to gather, publish, exchange, process and analyse context data in a fast and efficient way. The second one is the CEP which paired with the first one, allows detecting patterns above contexts namely triggering some actions or raising some alarms in reaction to situations rather than to single events.

- **Advanced Widget Mash-up HMI**: this component is a tool to create web application mash-ups\(^1\), integrating heterogeneous data, application logic, and UI components (widgets) to create new coherent and value-adding composite applications to be exploited at OPIL layer 3.

- **Backend Device Management & Protocol Adapter**: this component is the central enabler for the IoT objects integration of the Layer 1 with the OPIL platform. In fact, it connects IoT devices/gateways to the Context Management component, by translating IoT-specific protocols into the context information protocol used by the Context Broker sub-component. This means that users can query or subscribe -and being notified about- changes of device parameters status by querying or subscribing to the corresponding entity attributes in the Context Broker.

- **Service integration via Enterprise Service Bus**: This component enables end-users of OPIL to make their enterprise data accessible in the OPIL platform, such as work orders and related task definitions, through the integration of their legacy software and data stores, as well as transforming data seamlessly across different formats.

Layer 1 OPIL Components are:

\(^1\) A web page or application created by combining data or functionality from different sources
- **Robotic Agent Node**: this OPIL component provides the capability to deal with the physical actors of a logistics domain. Robot agents, according to the OPIL framework, can fall into two main distinct categories: (i) manipulation agents, i.e., those which are in charge of loading and unloading the good and product on the plant and on the AGVs; (ii) moving agents, i.e., those which are in charge of moving good or products from one place to another of the production plant. A robot agent has a functional block for Communication and Messaging which is in charge of handling the translation from the internal agent communication infrastructure and the OPIL middleware, the Local Abstraction Layer enables system configurability and provides an abstract description of the Robot Agent node suitable for system configurability, for instance it, provides information about whether the agent is a manipulation or a moving one, describe in the former case number of axes or the kinematics in the latter case. The Local Execution Layer provides the robot agent to execute the assigned tasks by interpreting it and mapping it to the internal functioning of the Robot Agent node, e.g., it performs autonomous navigation to accomplish a “move from A to B” task. Finally the Legacy Hardware Abstraction Layer, abstracts from the actual kinematics or dynamics of the robot providing a control abstraction to the Local Execution Layer, e.g., it provides an joint/cartesian set-points interface for a manipulation robot or a linear/angular velocity set-points interface for a mobile robot or an AGV.

- **Human Agent Node**: This OPIL node provides the capability for Human Agent to communicate with the system. The Human Agent can give inputs for the system and also receive information via this node. The Human Agent has a sub-module for Communication and Messaging, which is in charge of handling the translation from the internal agent communication infrastructure and the OPIL middleware. The second sub-module, the Local HMI Layer, provides multimodal, multichannel interaction capabilities to allow warehouse operators to communicate with OPIL in an easy and intuitive way.

- **Sensor Agent Node**: The Sensor Agent Node allows to collect relevant information from production (e.g. how many goods have been produced) and warehouse (e.g. is the place free to put the pallet down) for OPIL components like Task Planner, Sensing and Perception, Robot Agent etc. For information collection industrial HW Sensors are used. The Sensor Agent consist of following functional blocks: (i) HW Sensors are connected directly to the Sensor Agent Legacy Hardware Abstraction Layer, allowing to read the values provided by the HW Sensor and send this to the local execution layer; (ii) The Local Execution Layer provides the function to work with the data collected by the HW Sensors and to translate it to usable form, which is send to other OPIL layers over the communication and messaging; (iii) The Local Abstraction Layer enables system configurability (change Sensor Agent communication and other parameters) and to monitor to the Sensor Agent connected HW Sensors real-time values; (iv) The Communication and Messaging is handling the connection from Sensor agent to the OPIL middleware and translation.
3 Interfaces

3.1 Interface with non-ROS-based AGVs

An OPIL PC is needed for integration with Robotic Agents. The OPIL PC contains the Robotic Agent Node to take the control of the AGV. The communication between the AGV and the OPIL PC is performed by an ethernet cable. The AGV implements an UDP socket to send the information about the odometry and the current state, and to receive the speed commands. The diagram of the integration is shown in the Figure 2. RAN means Robotic Agent Node. RPS means ROS Publisher Subscriber.

The non-ROS-based AGV Easybot of the company ASTI Mobile Robotics has been successfully integrated in OPIL. The integration of another vendor-specific non-ROS AGVs is possible.

![Diagram of Non-ROS-based AGVs integration](image)

3.2 Interface with ROS-based AGVs

The AGV will integrate the RAN components (local execution layer, local abstraction layer and node communication/messaging) in its own ROS system. Speed commands and odometry will be interchanged by ROS messaging. In this case the information will be interchanged directly with the Local execution layer. The diagram of the integration is shown in the Figure 3.
Figure 3 – ROS-based AGVs integration
H2020 Innovation Action – This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 767642