CEN WORKSHOP AGREEMENT

CWA 17967

January 2023

ICS 13.180; 25.040.30

English version

Guidelines for design of advanced Human-Robot Collaborative cells in personalized HRC systems

This CEN Workshop Agreement has been drafted and approved by a Workshop of representatives of interested parties, the constitution of which is indicated in the foreword of this Workshop Agreement.

The formal process followed by the Workshop in the development of this Workshop Agreement has been endorsed by the National Members of CEN but neither the National Members of CEN nor the CEN-CENELEC Management Centre can be held accountable for the technical content of this CEN Workshop Agreement or possible conflicts with standards or legislation.

This CEN Workshop Agreement can in no way be held as being an official standard developed by CEN and its Members.

This CEN Workshop Agreement is publicly available as a reference document from the CEN Members National Standard Bodies.

CEN members are the national standards bodies of Austria, Belgium, Bulgaria, Croatia, Cyprus, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, Netherlands, Norway, Poland, Portugal, Republic of North Macedonia, Romania, Serbia, Slovakia, Slovenia, Spain, Sweden, Switzerland, Türkiye and United Kingdom.

EUROPEAN COMMITTEE FOR STANDARDIZATION
COMITÉ EUROPÉEN DE NORMALISATION
EUROPÄISCHES KOMITEE FÜR NORMUNG

CEN-CENELEC Management Centre: Rue de la Science 23, B-1040 Brussels

Ref. No.:CWA 17967:2023 E

Contents

Page

European foreword

This CEN Workshop Agreement (CWA 17967:2023) has been developed in accordance with the CEN-CENELEC Guide 29 “CEN/CENELEC Workshop Agreements – A rapid prototyping to standardization” and with the relevant provisions of CEN/CENELEC Internal Regulations - Part 2. It was approved by a Workshop of interested party representatives on 2022-10-07, the constitution of which was supported by CEN following the public call for participation made on 2022-07-25. This CEN Workshop Agreement does not necessarily include all relevant stakeholders.

The final text of the CEN Workshop Agreement is provided to CEN for publication on 2022-12-19.

Results incorporated in this CEN Workshop Agreement received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 820807.

The following organizations and individuals developed and approved this CEN Workshop Agreement:

- Néstor Garcia Hidalgo – Chairman, EURECAT (Spain)
- Simona Neri – Vice-Chairman, EURECAT (Spain)
- Nicola Pedrocchi; Andrea Orlandini, CNR (Italy)
- Luigi Calegari, MCM (Italy)
- Dionisis Andronas, LMS, UNIVERSITY OF PATRAS (Greece)
- Francesca Canale; Cristina Naso Rappis, STAM (Italy)
- Paul Eichler, FRAUNHOFER IWU (Germany)
- Julen Urain, TUDA (TECHNICAL UNIVERSITY DARMSTADT) (Germany)
- Nils Mandischer, RWTH (Germany)
- Alexandra Jaunet, STRANE (France)
- Mikel Anasagasti, GOIZPER (Spain)
- Jaume Altesa, ASLTOM (Spain)
- Andrea Scala, CEMBRE (Italy)
- Spyros Koukas, INTRASOFT (Luxembourg)
- Nicola Helfer, KAN (Germany)
- Aaron Prather, ASTM
- Richard Merkel, VDMA (Germany)
- Steven Carter, CEN/TC 310
- Amanda Suo, UNE, Spanish Association for Standardization (Spain)

Introduction

Human-Robot Collaboration (HRC) is expected to be a core element of future factories. Combining the repeatability and tirelessness of robots with humans' versatility and critical thinking skills often boosts flexibility and productivity of industrial processes. However, the design of effective methodologies to steer the deployment of this new paradigm in real production environments is an open challenge for both researchers and companies [1]. Work organization and technical solutions for Cyber-Physical Systems (CPS) are supposed to evolve between two extreme alternatives: (i) the *techno-centric scenario* and (ii) the *anthropo-centric scenario* [2]. In the techno-centric scenario, the technological aspects dominate the organization of the work. In contrast, in the anthropo-centric scenario, human workers control the work, and technology helps them make decisions.

Existing approaches to CPS (and, among them, HRC) oscillate between these two extremes. However, human factors have gained more attention in the design of novel methodologies for personalized production dynamics based on the operators' preferences, technical skills, and health-related issues.

Regarding HRC, human factors have been considered at different levels [3]. For example, optimization of human factors has been embedded into a task scheduler [4]; task allocation has been exploited to reduce the workload of human workers [5, 6]; task synergy between human-robot tasks were optimized to reduce the cycle time [7]; human-aware motion planners demonstrated to be preferable by human users [8]. The works mentioned above mainly focus on the planning aspects of HRC. However, they disregard the complex effect of human-robot communication on the user experience. A stuttering human-system communication is often a major bottleneck to a fruitful collaborative process. For this reason, the communication between the human operator and the system is an object of intense study. In this regard, Augmented Reality (AR) is a striking tool able to overlay instructions and knowledge from CPSs to the physical operator's view [9].

Driven by this consideration, this CWA aims to provide an all-around approach to HRC, where robots and operators collaborate at different cognitive and physical levels. A key objective is to make implicit and explicit communications between robots and humans smooth and fruitful. Explicit communications leverage multi-modal technologies and, in particular, Augmented Reality tools. Implicit communications require the robotic system to reason on the operator's intentions and act consequently. Therefore, task representation and planning are fundamental to provide the robot with the necessary autonomy and suitable initiative.

This document presents the design methodology and deployment actions needed to provide a *user-aware* approach to HRC that enhances the flexibility of HRC systems. In addition, human-aware paradigms usually consider a one-fits-all solution, considering the human an anonymous agent. Here, we go beyond this concept and propose a user-centric methodology to shape the robots' behaviour based on the specific characteristics of a single user (e.g., age, skills, experience) and preferences (e.g., left-handed vs. right-handed) [10], i.e., implementing personalized robot behaviour that can better serve the human operator and, potentially, increase the technology perception and acceptance. Therefore, we propose the integration of planning, perception, and communication into a unified technological framework.

An AI-based Knowledge Representation & Reasoning module encapsulates a user model representing features of human workers that are relevant with respect to production needs (e.g., match users' skills to the requirements of production tasks). Combined *AI-based task* & *motion planning* modules reason on this knowledge to coordinate human and robot agents taking into account known skills and features of the worker, while pursuing an optimization perspective. Furthermore, an AR-based *Human-System Interaction Module* realizes advanced interaction mechanisms to contextualize communication to and from the worker to facilitate explicit human-robot communications and collaboration.

Considering the common challenges due to human-robot collaboration (HRC) for various application domains, the objective of this CWA is to provide a unified transversal framework consisting of the integration of planning, perception, and communication in human-robot collaboration (HRC) systems, by using a transversal approach based on standards and on well-established best practices. It presents the design methodology and deployment actions to provide a user-aware approach to HRC that enhances the flexibility of HRC systems. This is a user-centric methodology to shape the robot behaviour based on a single user's specific characteristics (e.g., age, skills, experience) and preferences (e.g., left-handed vs. right-handed), implementing personalized robot behaviour that can better serve the human operator and potentially increase the perception and acceptance of the technology. The guideline is applicable to different robot categories and use scenarios.

This document does not apply to the following devices, systems, and applications: autonomous vehicles for the transportation of persons, drones, rescue robots (including ground, marine and aerial vehicles), surgical robots in relation to the body of the patient, passive wearable devices, external limb prostheses.

This CWA has been promoted by the SHAREWORK project (‘*Safe and effective human-robot cooperation towards a better competitiveness on current automation lack manufacturing processes*’). It is 4-year (2018 – 2022) project funded by the European Union’s Horizon 2020 Framework Programme for Research and Innovation under Grant Agreement No 820807. It brings together fifteen partners from 6 different European countries (Spain, Germany, France, Luxembourg, Italy, and Greece).

1 Scope

This CEN Workshop Agreement (CWA) defines a technical/methodological framework for human-robot collaboration (HRC) systems that integrates planning, perception, and communication. Specifically, it provides guidelines for the design methodology and deployment actions to provide a user-aware approach to HRC cell that increases the adaptability and flexibility of HRC systems. This is a user-centric methodology to shape robot behaviour based on a single user's specific characteristics (e.g., age, skills, experience) and preferences (e.g., left-handed versus right-handed), implementing personalized robot behaviour that can better serve the human operator and increase the perception and acceptance of the technology.

This CWA will not define requirements related to safety aspects.

Furthermore, any consumer or user of CWA framework, architecture, and component source code should do their own formal integrated risk assessment and EU Machine Safety Directive compliance. Users should also be responsible for integrating and testing any CWA solution architecture, network latency, security, and open-source software code to ensure that it meets the specific application requirements of the users, and that any modifications made are the responsibility of the system, integrator, etc.

This document is informative and is not aimed at substituting or simplifying production procedures required by standards. The objectives of this document are the following:

— Define the design methodology and deployment actions needed to provide a user-aware approach to HRC that enhances the flexibility of HRC systems.

— Present the user models and the knowledge-based formalism to represent users and production information.

— Explain how the framework embeds user-awareness, with a particular focus on the planning and communication modules.

— Present an example of the integration of the framework into a manufacturing scenario.

2 Normative references

There are no normative references in this document.

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/TS 15066:2016 apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at https://www.electropedia.org/

4 HRC system design methodology

This document describes an effective control system for anthropocentric HRC in fence-less environments. It follows a HRC architecture which is modular, distributed, service-oriented architecture (SoA) that defines a set of fifteen different software and hardware modules designed as stand-alone, interacting components communicating through well-defined interfaces. The architecture is fully interoperable and supports various module configurations that can be customized according to industrial needs. The HRC architecture includes modules that understand the environment and human actions through knowledge and sensors, predict future state conditions, implement smart data processing, provide augmented reality and gesture and speech recognition technology.

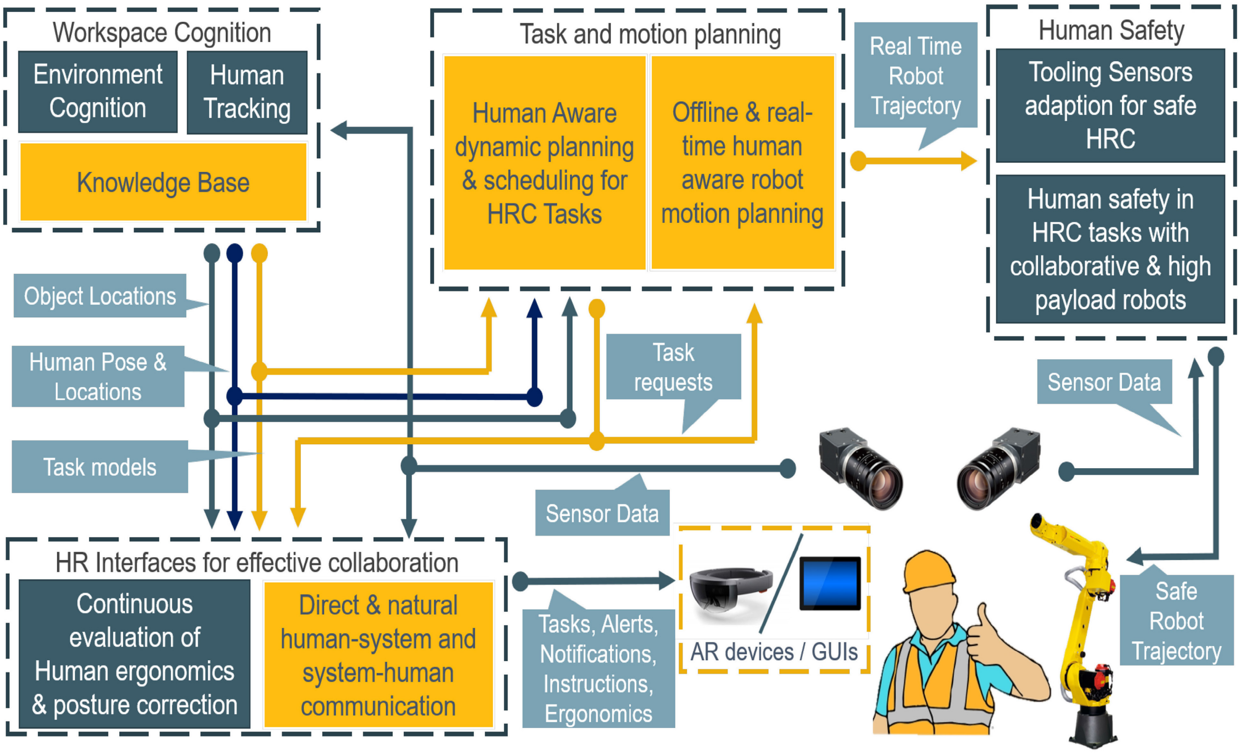


Figure 1 — Overview of the HRC reference architecture (Safety considerations are out of the scope of this document)

4.1 General Architecture and Module Overview

Figure 1 shows a high-level overview of the HRC reference architecture, depicting the set of different modules and the high-level flow of information among them. The picture highlights the interconnection of Workspace Cognition, Planning, and human-robot communication composing the backbone of the architecture. Notwithstanding the modularity of the proposed approach, the core modules are combined into a user-centric framework oriented to user preferences and human factors at all levels (e.g., process representation, robot motion, human communication). Not only the single modules are user-centric *per se*, but they are also connected in such a way that the output of each module contains helpful information to enhance the user-awareness of other modules. It is the case, for example, of the trajectories planned by the Action&Motion Planner, which are visualized by the Human-System Communication module, or the user profiles stored in the Knowledge Base, which instantiates different communication interfaces based on the users' preferences.

This section focuses on how each of the main modules support *user-awareness* within the proposed framework:

— **The Knowledge Base Module** stores a formal representation of the status of the production environment based on the *ontology for human-robot collaboration*. This module aggregates and elaborate information gathered from other modules to infer contextualized knowledge concerning for example situations/states of a worker, of the environment, of a production process being executed.

— **The Task Planning Module** coordinates the worker and the robot to cooperatively conduct production processes. This module synthesizes a flexible temporal schedule of the tasks the worker and the robot should perform.

— **The Action&Motion Planning Module** receives a task from the Task Planning Module and finds a sequence of feasible movements to execute it. It comprises an Action Decomposition layer that converts a high-level task (*e.g.*, pick an object, screw a bolt) into a sequence of motion planning problems. Then, it uses a motion planning algorithm to solve each problem and returns a sequence of trajectories that executes the high-level task. To consider the user, the Action&Motion Planning Module runs online; that is, all trajectories are calculated on the fly, just before their execution. To do so, it exploits human tracking data, usually acquired through a vision system. This is necessary for two reasons: first, avoiding collisions and interference with the user (who is moving in the cell); second, adapting to changes in the environment (*e.g.*, the user may move objects and tools during the work).

— **The Human-System Interaction Module** provides a bidirectional communication framework between operators and the HRC system. By incorporating various interface devices and sensors, a multi-modal interaction pipeline is structured to facilitate communication of (i) data and goals to the system (by the user) and; (ii) pending and current tasks, robot trajectories, event notifications, report results to the operator (by the system). Communication channels include AR devices and tablet interfaces. Supported by the knowledge base's ontology, the *Human-System Interaction Module* can be tailored to the operator's preferences and needs to establish an intuitive and user-aware working environment.

4.2 General Integration of Modules Supporting Personalized Collaboration

This section discusses how the modules introduced above work together for *user-awareness*. Figure 2 shows the integration of these modules and the information and control flow. Communication mechanisms and the exchange of messages/signals among the modules can rely on existing software libraries and tools (e.g., Robot Operating System – ROS). Each module indeed defines a set of topics and services used to offer information/functionalities to and gather the necessary information from other modules.

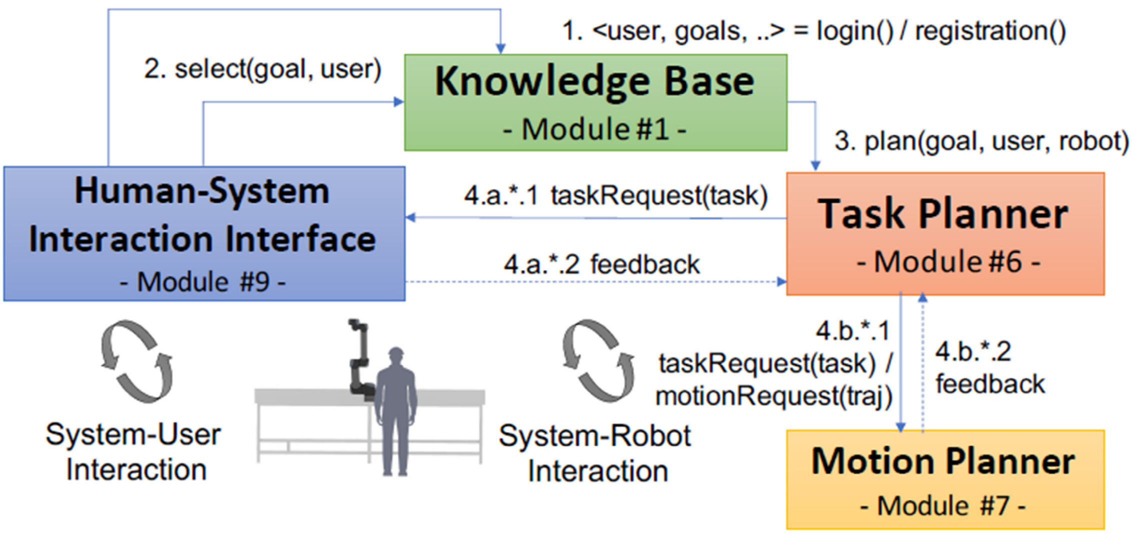


Figure 2 — Integration schema of the considered modules

First, the Human-System Interaction Interface authenticates a particular worker into the system and retrieves information about his/her *user profile* (e.g., data from previous sessions, preferences, known skills) and information about the production context (e.g., known production goals, related production procedures, skills of the collaborative robot). The worker decides the production goal to perform and sends a “starting signal” to the Knowledge Base module through the Human-System Interaction Interface. The generated message specifies the production goal and the ID of the user that takes part to the process.

The Knowledge Base receives this signal through a subscribed topic, contextualizes knowledge (e.g., infer the subset of operations the worker and the robot can perform) and configures the Task Planner by calling a dedicated service. This service specifically allows the Knowledge Base to automatically define the *control variables* of the planning model according to the requested production goal and the profile of the user (e.g., robot capabilities, operator skills, performance profile). The Task Planner then synthesizes and executes an optimized task plan. During the execution, the module dispatches task requests to the *Human-System Interaction Interface* and the Action&Motion Planner to interact with the robot.

The *Human-System Interaction Interface* displays information on the tasks requested to the human and waits for feedback from the operator. This ensures the correct dispatching of the task plan to the human actor. Similarly, the Action&Motion Planner receives tasks’ requests for the robot and puts them into action. After the execution of the task, it sends feedback to the Task Planner to inform it about the outcome. The Human-System Interaction Interface and the Action&Motion Planner offer a set of actions that enable visualization and monitoring of human and robot tasks. For example, the Action&Motion Planner informs the Human-System Interaction Interface on the future robot trajectories so that they can be visualized on an interface (e.g., through AR).

4.3 Modular deployment through containerization

The modularity of the HRC architecture should be also reflected in the software packaging. The system should use a toolset to increase productivity, reduce the setup time in complex environments, and easily configure a customized version of the HRC architecture. Each module is packaged in a separate docker image and uploaded to a docker repository. Exceptions to this rule can exist when there are software modules with specific run-time requirements (e.g., Android applications of modules running on a standard Android tablet). By using a docker compose tool that enables the definition and execution of multi-container applications, the HRC system can be configured to run in different configurations using an appropriate configuration file. Then, a HRC instance can be created and started with a single command.

5 Ontology-based Model of Workers

HRC scenarios pursues a tight “teamwork” between the human and the robot requiring shared view and “mutual understanding” of the objective, constraints, capabilities, and limitation of each other member as well as an implicit or explicit agreement about the procedure to follow [12, 13, 14]. The ontological model supports the effective coordination of human and robotic agents by providing a formal representation of:

(i) production objectives, tasks, and operational constraints;

(ii) worker and robot capabilities/skills;

(iii) known performances, preferences and physical/behavioural features of workers that may affect the interactions with the robot and the resulting collaborative processes.

5.1 Context-based Ontology for Collaborative Scenarios

The Ontology for Human-Robot Collaboration has been introduced in [11] as a general model characterizing collaborative dynamics between human and robot agents acting in a manufacturing scenario. It therefore defines the formal model (TBox) the *Knowledge Base Module* uses to build an abstraction of the production environment (ABox) and infer/contextualized useful information. Ontology is organized into a number of *contexts*, each defining concepts and properties that characterize an HRC scenario with respect to a particular perspective. A knowledge base is structured in shape of Knowledge Graphs (KGs) [15, 16] and thus can be manipulated through standard semantic technologies, e.g., based on the Web Ontology Language (OWL) from W3C [17].

As shown in [11] the environment, behaviour and production contexts describe respectively:

(i) physical entities and observable properties of a collaborative environment;

(ii) skills and capabilities of the human and robot;

(iii) production goals, tasks, and constraints of the HRC process.

The behaviour context uses the concept of Function [18] to correlate production tasks with the low-level operations the worker and the robot can perform (i.e., the functions).

5.2 Functions and Production Requirements

The goal of the Ontology is to characterize production objectives, human and robot capabilities and thus contextualize operations they can perform to conduct production tasks collaboratively. The concepts Cobot and HumanWorker are defined as a specialization of the DUL:Agent. The acting qualities of each agent are represented by means of Capability and Function. Capabilities characterize competencies that agents have according to their structures and skills. For example, a human worker can perform welding operations if she is skilled in that task. Similarly, a robot can perform “pick and place” of objects if it is endowed with a gripper. Figure 3 shows an excerpt of the Ontology pointing out the taxonomical structure of the concepts representing different types of production tasks.

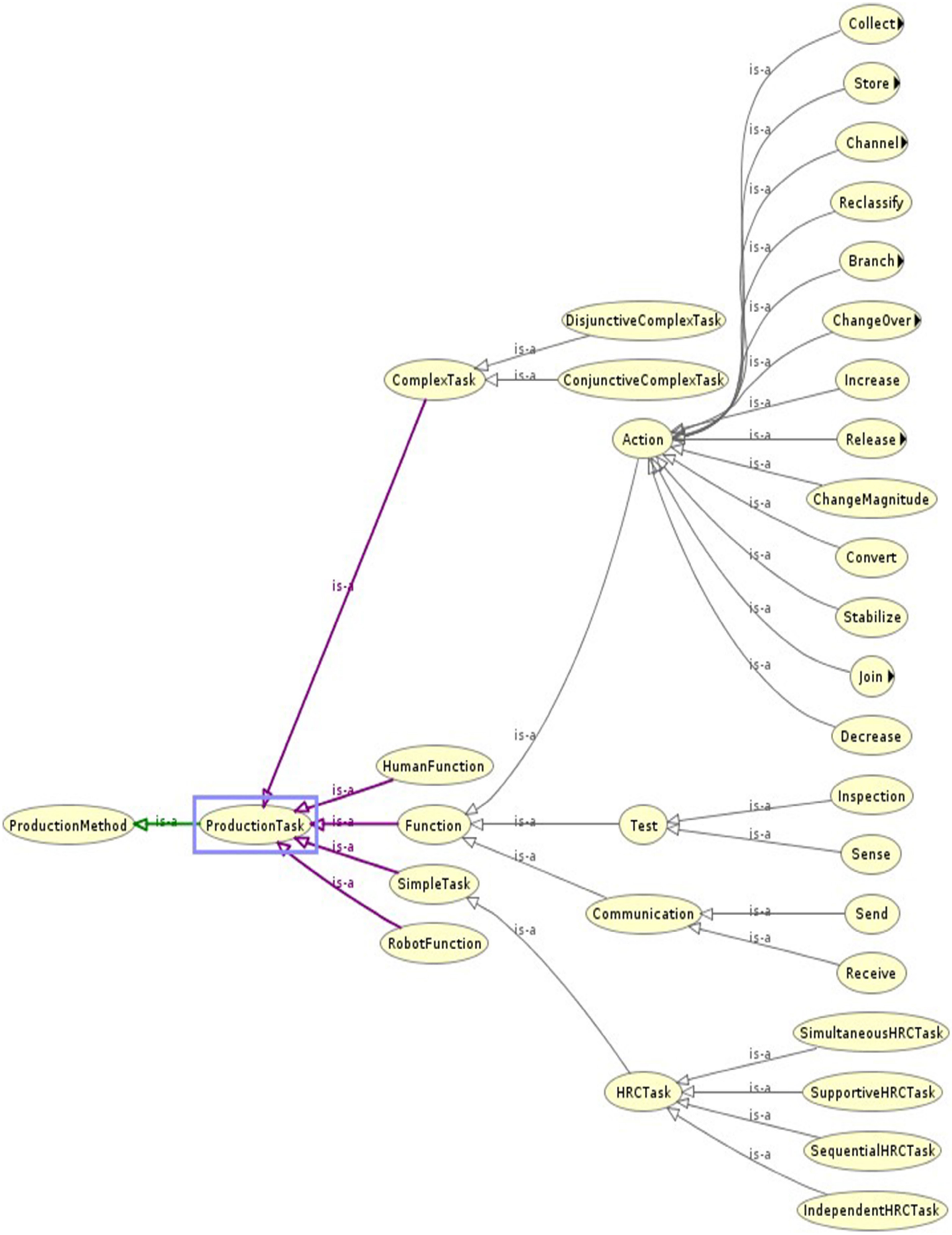


Figure 3 — Excerpt of the Ontology showing the taxonomical structure of production tasks. The picture shows the integrated taxonomy of functions introduced in [18]

While capabilities do not depend on the features of a production context, the concept of Function characterizes low-level production tasks humans and robots should perform in a manufacturing environment. The Ontology integrates the Taxonomy of Functions defined in [18] and defines different types of Function according to the effects they have on DUL:Quality of objects. The instances of Function a generic agent can perform can be dynamically inferred according to actual capabilities of that agent. Namely, the model of Function proposed in [18] is extended to correlate them to the set of Capability needed to correctly perform them. The separation between functions and capabilities supports contextual reasoning since functions contextualize general agents' capabilities with respect to the needs of a production scenario.

$$
\begin{aligned}
\text{Function} \sqsubseteq\ & \text{ProductionTask} \sqcap \\
& \exists\ \text{DUL:isDescribedBy.ProductionNorm} \sqcap \\
& \quad \exists\ \text{canBePerdformedBy.DUL:Agent} \sqcap \\
& \exists\ \text{hasEffectOn.DUL:Quality} \sqcap \qquad (1) \\
& \quad \exists\ \text{hasTarget.ProductionObject} \sqcap \\
& \quad \exists\ \text{requires.ProductionObject} \sqcap \\
& \quad \exists\ \text{requires.Capability}
\end{aligned}
$$

The description of a production process follows a task-oriented approach. The top-level element is the ProductionGoal which defines the general objectives of a production context. Each ProductionGoal is associated with a number of ProductionMethod (at least one method for each goal is necessary) specifying production and operational constraints.

Each ProductionMethod always refers to one ProductionGoal and is composed by a hierarchical organization of ProductionTask. The ontology defines three types of tasks: (i) ComplexTask (either *disjunctive* or *conjunctive*); (ii) SimpleTask and; (iii) Function. A ComplexTask is a ProductionTask (i.e., an instance of DUL:Method) representing a compound logical operation. The hierarchical structure is enforced by the property hasConstituent which associates ComplexTask with either SimpleTask or other ComplexTask.

$$
\begin{aligned}
\text{ComplexTask} \sqsubseteq\ & \text{ProductionTask} \sqcap \\
& \exists\ \text{DUL:hasConstituent.(ComplexTask} \sqcup \text{SimpleTask)} \sqcap \\
& \exists\ \text{DUL:isDescribedBy.OperativeConstraint}
\end{aligned}
\qquad (2)
$$

A SimpleTask represents a leaf of the hierarchical structure of a ProductionMethod. This concept describes primitive production operations that could be conducted leveraging the functional capabilities of the agents. A SimpleTask requires the execution of several Function instances by the agents.

$$
\begin{aligned}
\text{SimpleTask} \sqsubseteq\ & \text{ProductionTask} \sqcap \\
& \exists\ \text{DUL:hasConstituent.Function} \sqcap \\
& \exists\ \text{DUL:hasConstituent.SimpleWorkpiece} \sqcap \\
& \quad \exists\ \text{DUL:isDescribedBy.(InteractionModality} \sqcup \text{OperativeConstraint)}
\end{aligned}
\qquad (3)
$$

The execution of a task should comply with operational constraints that are represented as ExecutionNorm. Two main types of execution norms can be defined: the concept OperativeConstraint describes *norms* requiring the sequential or parallel execution of tasks; the concept of InteractionModality instead characterizes *norms* about how agents should cooperate to conduct a task.

5.3 Human Factor and User Model

The current work specifically focuses on the *Human Factor* context and elaborates on its correlations with the *behaviour* and the *production* contexts. Figure 4 shows part of the taxonomic structure defined to represent behavioural and physical features of workers. Such concepts define the variables composing the *user model* and therefore characterize the representational space of qualitative aspects of a worker (i.e., types of DOLCE:Quality).

Concepts characterizing the qualities of the physical body of a worker, Figure 4 (a), model physical, health, and cognitive parameters. Information about these variables enables the detection and monitoring of anomalous or dangerous working conditions, such as bad ergonomics, body position in hazardous areas or mental, and physical fatigue. Concepts concerning the qualities of the behaviour of a worker, Figure 4 (b), instead model his/her performance in a given production scenario (e.g., the expertise level or the average time taken to perform a task).

The concept WorkerExpertiseLevel estimates "how much knowledgeable" a worker is about a particular production scenario. On the one hand, the expertise level determines the (sub)set of production tasks a human worker can conduct. For example, some tasks may require a certain minimum level of experience to be performed by a worker. On the other hand, it characterizes the reliability of the performance of a worker and thus the expected *uncertainty* about the duration of executed tasks. Low experience determines higher uncertainty and thus higher *variance* of the performance. High experience instead denotes lower uncertainty and thus more consolidated performance (i.e., lower *variance*).

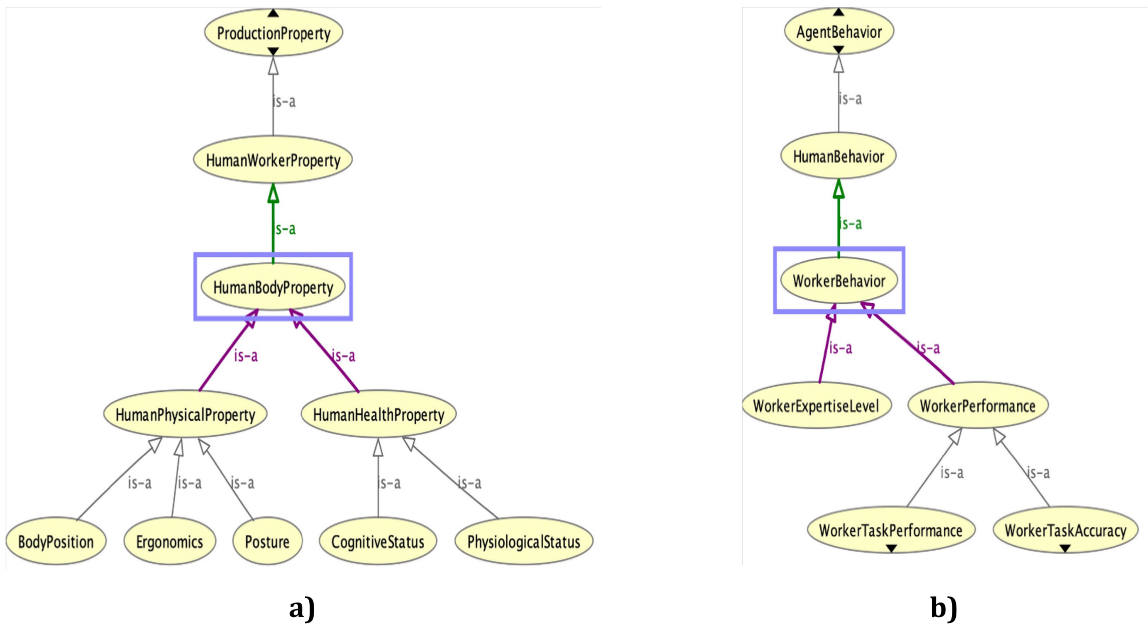


Figure 4 — Excerpt of the Ontology concerning the variables of the user model

The concepts WorkerPerformance supports a numerical representation of user performance. The Ontology distinguishes between *accuracy* (WorkerTaskAccuracy) and *efficiency* (WorkerTaskPerformance). These variables support the incremental definition of a *dataset* collecting historical data about performance. Such a dataset can be analysed to incrementally *learn* performance of users and adapt collaborative processes over time. It can be used for example to infer an efficiency matrix encoding the average completion time of production tasks for (known) users.

6 User-Aware Collaboration

The production and user-centred knowledge pushes forward novel collaboration paradigms where the system adapts interactions and collaborative processes to the *known* features of participating users. Knowledge inference and extraction procedures can be implemented to dynamically generate contextualized planning models to the specific features of a domain [19, 20] as well as specific skills and preferences of a human worker.

This section explains how the task planning, the Action&Motion, and the human-system interaction module take advantage of the user model to support personalization and adaptation.

Artificial Intelligence Planning & Scheduling [21, 22, 23] is well suited to endow robot controllers with the *flexibility* needed to autonomously decide actions and adapt behaviours to the state of the environment [24, 25]. Planning technologies pursue an *optimization perspective* aiming at finding plans that minimize or maximize a specific metric (e.g., minimization of the planning cost). Different metrics and features of a domain can be considered, depending on the specific planning formalism used. In application domains like HRC, reasoning about causality, time, concurrency, and simultaneous behaviours of domain features (e.g., the human and the robot) is crucial to synthesize and execute effective plans.

Task planning capabilities developed within HRC can rely, for example, on the timeline-based formalism [26] and the PLATINUm software framework [27, 28, 29]. This planning formalism integrates reasoning about causal and temporal aspects of a planning problem and has been successfully applied to several concrete scenarios [30, 31, 32]. PLATINUm and the formalism introduced in [26] integrates *temporal uncertainty* and *controllability issues* to generate *robust* plans when executed in the real world [25, 33]. Uncertainty is especially important in HRC, where robots should continuously interact with *uncontrollable* autonomous entities like human workers. Considering the manufacturing context and other works synthesizing optimal, multi-objective assembly processes [34, 35], PLATINUm is extended by integrating multiple objectives and uncertainty. This allows the synthesis of (timeline-based) plans that achieve a good trade-off between efficiency (i.e., minimizing the cycle time of collaborative processes), and considering temporal uncertainty for reliable execution [36].

It is worth underlining that the AI architecture we propose does not rely on statistical machine-learning techniques. In other words, the approach does not require any data set to work. As depicted in the following paragraphs, the information needed is a few attributes for modeling the tasks and users. We need only the information on who can do what, and how much time approximatively they need. The map of the skill between agents (humans and robots) and the measure of the duration of the tasks is the basic knowledge already available for all modern production lines according to the gold operation management rules.

Wrong estimation of such information could lead to a non-optimal schedule. However, the measure of the durations could be easily automated in the line, and therefore the system could easily improve the performance at each production step.

6.1 Personalized Task Planning

A two-layer model can achieve personalized task planning:

1) Using Timelines, exploit the possibility of modeling the sequence of a task in a simple, straightforward mode. The model inputs are the expected duration required, the sequence constraints with the other tasks, and a few other pieces of information.

2) The definition of the user skills, that is, a map of what the human can do or cannot. Each different human agent may have a different representation.

3) The definition of an expert level is used to characterize the expected variance of the average duration. The combination of this information

This approach easily adds a new user to the system and does not modify the solver implemented in the software tool. Indeed, the SW tool must be multi-agent; that is, it should allow a parametrization of the problem according to various resources, both robotic and human.

Specifically, A timeline-based specification consists of several state variables that describe behaviours of domain features. A state variable is formally defined as a tuple SV = ⟨V, T, D,γ⟩.

— A set of values vi ∈ V represent states and actions the domain feature can assume or perform over time.

— A transition function T : V → 2V specified valid sequences of values vi ∈ V.

— A duration function D: V → R × R specifies each value vi ∈ V the expected lower and upper bounds of its execution time.

— A controllability tagging function γ: V → {c,pc,u} specifies if the execution of a vi ∈ V is controllable (c), partially controllable (pc), or uncontrollable (u). Information about controllability allows a task planner to deal with uncontrollable dynamics of the environment when executing a (timeline-based) plan. This is known as the controllability problem [37] and is particularly important when an artificial agent like a collaborative robot should interact with "unpredictable" agents like a human worker. Synchronization rules constrain the "runtime" behaviour of the modelled domain features. They specify causal and temporal constraints necessary to coordinate the different features complex system (e.g., an HRC cell) and synthesize good temporal behaviours (i.e., the timelines).

The definition of state variables and synchronization rules modeling a HRC scenario follows a hierarchical decomposition methodology correlating high-level production goals to simpler production tasks and functions [38].

The system is model with a set of state variables:

— A state variable SV_G describes the high-level production goals supported by the HRC work-cell.

— A number of state variables $SV_L{}^i$ where $i = 0,...,K$ describe the production procedure at different levels of abstraction. The values of these state variables represent production tasks at a specific level of abstraction $i \leq K$ (where K is the number of hierarchy levels of the procedure).

— A set of variables SV_R describes the low-level operations (i.e., instances of Function) the robot can perform.

— A set of state variables SV_H describes the low-level operations (i.e., instances of Function) the human can perform. Each state variable sv_H describes the behavioural dynamics of the worker collaborating with the robot. Each state variable $sv_H = \langle V_H, T_H, D_H, \gamma_H \rangle$ is thus generated from the knowledge base according to the *user profile* of the participating worker. The values $v_j \in V_H$ are defined according to the tasks/functions the worker can perform in the given production scenario. No assumptions can be made on the actual duration of tasks/functions assigned to the worker. Consequently, all the values of SV_H are tagged as uncontrollable, $\gamma_H(v_j) = u, \forall\, v_j \in V_H$. The duration bounds of each value $v_j \in V_H$ and are defined by considering the mentioned performance matrix that can be extracted from the knowledge base.

— A set of synchronization rules S describes the procedural decomposition of high-level goals (i.e., values of state variable SV_G) into simpler production tasks (i.e., values of state variables $SV_L{}^i$), until they are associated with several functions of the human and the robot (i.e., values of state variables SV_R and SV_H).

Finally, a *performance vector* is extracted representing the known performance of user $u_i \in$ U. Such a vector specifies, for each value $v_j \in V_H$, the average time $\delta_{i,j}$ the user u_i takes to accomplish the *task*(v_j) = $t_j \in$ T ($\delta_{i,j} = \infty$ if no information is available).

At this point, the user's expertise level characterizes the expected variance of the average duration. The combination of this information is thus used to define the *personalized* lower and upper duration bounds for each value $v_j \in SV_H$. Specifically, a certain amount of uncertainty is associated to each of the three expertise levels defined into the ontological model: (i) *novice*; (ii) *intermediate*; (iii) *expert*. The higher the expertise level the lower the uncertainty about the performance. It is defined an *uncertainty index* associating each expertise level with constant value of uncertainty to consider: Ω = {0.8,0.5,0.2}. Given a user $u_i \in$ U, a function ϒ : U → Ω specifies the uncertainty index corresponding to the expertise level of the user. The resulting duration bounds of the values composing the state variable of the worker $v_j \in V_H$ are then defined as follows:

$$D\,(vj) = (\delta i,j - \omega i * \delta i,j, \delta i,j + \omega i * \delta i,j) \tag{4}$$

This mechanism dynamically adapts the temporal dynamics encapsulated into a task planning model according to the changing performance of the same worker as well as to the performance of different workers. The finer the temporal model of the worker, the better the optimization of plans and resulting collaborative processes [39].

6.2 Integrated Task and Motion Planning

To guarantee a high level of flexibility in the planning and execution of collaborative tasks, a hierarchical Task&Motion Planning framework that permits online planning of the robot trajectories according to the user and the environment's state should be implemented. This is key to ensuring a smooth collaboration between the human and the robot because the robot's tasks can be robust with respect to changes in objects and tools' positions, and the robot's movement can be optimized to avoid interference with the user's activities [40].

The key idea of this approach is that robot tasks coming from the task planner are symbolic and should be converted into a sequence of geometric movements by the Action&Motion Planning module. In this way, the task planner reasons about the best assignment and scheduling of tasks disregarding the actual geometric realization of each task. This is necessary because the robot's actual trajectories are not known a priori. Indeed, they need to be planned and adjusted on the fly according to the scene. Moreover, the motion planner can feature user-aware behaviour that makes the robot's motion more dependable according to his/her preferences.

The Action&Motion planning module consists of a hierarchical framework composed of (i) a task decomposition module, (ii) a proactive motion planner, (iii) a reactive speed modulation module. A scheme of the proposed framework is in Figure 5.

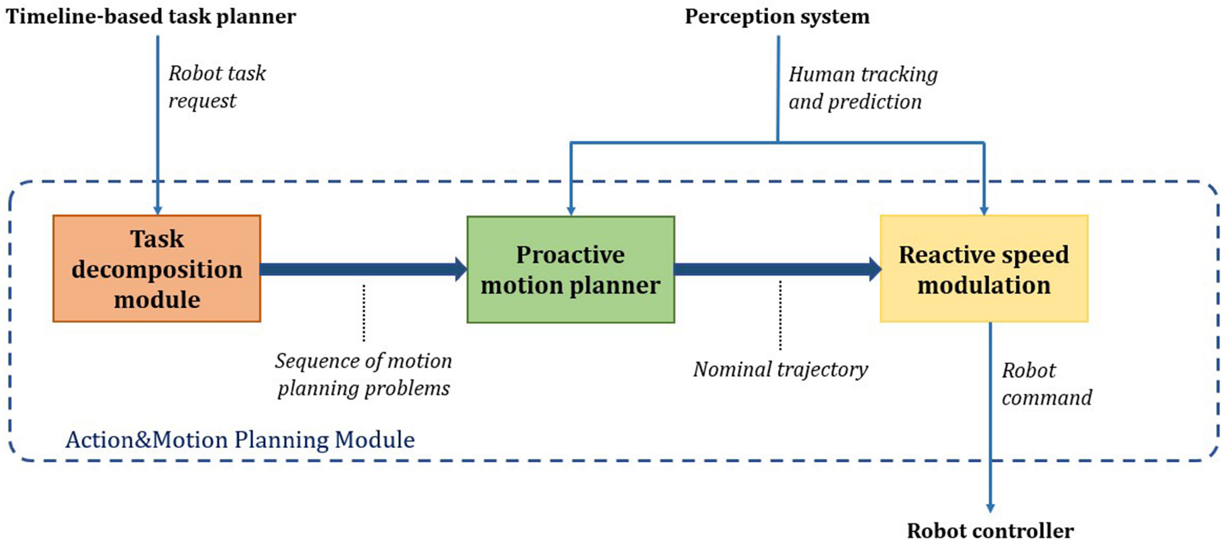


Figure 5 — Action&Motion Planning module

6.2.1 Task decomposition module

The task decomposition module owns a set of predefined skills; that is, high-level behaviors that the robot can execute autonomously (e.g., *pick an object*, *screw a bolt*). Skills are a model of an abstract task and allow the module to decompose a task into a sequence of robot movements. For example, the task *pick an object* is decomposed into a sequence of basic movements and conditions: (i) check if the gripper is empty; (ii) open gripper; (iii) move to approach pose; (iv) move to grasp pose; (v) close gripper.

Given an object to be picked, the task decomposition module retrieves the necessary geometric information from the scene (*e.g.*, by querying the Knowledge Base), checks whether the task conditions hold, and initializes the basic movements according to the scene state. Notice that, at this stage, a task might have multiple equivalent geometric realizations. For example, the symbolic task *pick a blue cube* may require choosing among multiple blue cubes, each with numerous grasping points. This level of complexity is addressed by the *proactive motion planner*.

6.2.2 Proactive motion planner

The proactive motion planner solves the motion planning problem related to each basic movement of a task, as decomposed by the Task decomposition module. The term proactive distinguishes this module from the reactive speed modulation module. The proactive planner is intended to find a collision-free trajectory according to a prediction of the user’s actions and movements. Once a trajectory has been found, its execution starts, and the reactive layer monitors and adjusts it according to real-time scene information. Moreover, the path is sent to the *Human-System Interaction Module* for visualization so that the user will foresee the robot’s movement in the short run.

The proactive trajectory planner has been implemented by using the standard path-velocity decomposition paradigm, in which a path planner finds a collision-free path from a start to a goal state, and a path parametrization algorithm (*e.g.*, TOPP [41]) optimizes the velocity profile along the path. Regarding path planners, sampling-based algorithms are preferred, for they can efficiently deal with high-dimensional search space [42].

User awareness is embedded in the path planner using a cost function that depends on the human state. The typical approach minimizes a weighted sum of an efficiency term (e.g., the path length) and user-aware terms, such as human-robot distance [43], trajectory repeatability [44], or human visibility [45].

NOTE Aspects of motion including safety considerations are out of the scope of this CWA and shall be designed in accordance with the relevant ISO/TC 299 or CEN/TC 310 standards, where necessary.

6.2.3 Reactive speed modulation module

The reactive speed modulation module modifies the nominal speed during the execution of each trajectory. In general, reactive motion planners are shifting from simple yet conservative strategies such as safety zones to optimized methods that adapt the robot motion continuously [46, 47].

The robot's speed is adjusted in this task in accordance with the requirements in the technical specification ISO/TS 15066 (*Robots and robotic devices – Collaborative robots*), which defines speed reduction rules for collaborative operations with and without admissible contact between robots and humans. For example, if *speed and separation monitoring* is applied, the human-robot distance S must not fall below a protective distance S_p. [48]

NOTE Aspects of speed modulation including safety considerations are out of the scope of this CWA and shall be designed in accordance with the relevant ISO/TC 299 or CEN/TC 310 standards, where necessary.

6.3 Augmented Human-Robot Interaction

The human-robot interaction framework aims to structure a usable and personalized interaction pipeline between the operator and the robot towards increased awareness and well-being. All those attributes are mostly accomplished by using multiple human senses (i.e., vision, hearing, touch) through interaction modalities available in customized interfaces [49]. Both modalities and customization options formulate a user-centric framework that can meet the requirements of novice and advanced operators.

In terms of architecture, the multi-modal interaction framework consists of three layers (see Figure 6). A broker forms the top layer of this module and is responsible for parsing information from the HRC's modules to the end devices, and vice versa [50]. The intermediate layer incorporates the available end devices, thus their respective applications. The bottom layer gathers all the supported interaction modalities based on the specifications of the intermediate hardware. The existence of the broker ensures stability against a varying number of deployed devices. During operations, there are redundant ways of interaction since information can flow simultaneously to all devices. This suppleness does not only serve the anthropocentric HRC system design principles, but also contributes to the overall system resilience against hardware limitations (e.g., battery, network range) or even issues (e.g., damages).

Focusing on the information streams, the module comprises mechanisms for human-system (HS) and system-human (SH) interaction. The former ones are needed either for operator monitoring or direct robot control. Despite the advances in machine learning for human activity recognition, the improvisation of operators can still highlight limitations in those systems. Thus, the developed interaction framework supports functionalities for monitoring purposes. In detail, all deployed applications should involve a "Task completed" feedback apparatus in the form of voice commands, touch, and augmented buttons. The same modalities can also be used as inputs for direct robot and system control (e.g., stop or proceed). Each application processes those inputs and communicates to the broker normalized commands or requests that are parsed to the rest of the HRC modules. On the contrary, when the system communicates to the human, the involved modules share information to the broker that is then streamed simultaneously to all end devices. Each application makes available the information based on the hardware's capabilities in the form of textual, graphical, or audio material.

The volume and type of communicated information should be closely related to the operator's experience level. For novice users, the module offers intuitive visual instructions that can support them during assembly operations via augmented 3D models, panels, arrows, or screen-based figures. For greater awareness, robot-centric information can also be provided through 3D augmented trajectories, notifications, and warnings. Textual instructions and info are standard in plain or extended format. Unlike novice users, who need support and a clear description of robot behavior, experienced operators could be distracted if they are communicated with all aspects of information. For this reason, the customization of the interaction framework can be performed during runtime through the related options panels. According to each operator's entity, tailoring of the interfaces is supported by the Knowledge Base. The customization options suggest the selection of available devices, available modalities, assembly information detailing, feature positioning, button positioning, and robot information detailing. The personalization of the system's front-end through customizable applications and the selection of multiple devices is achieved by implementing a distinct hierarchical architecture.

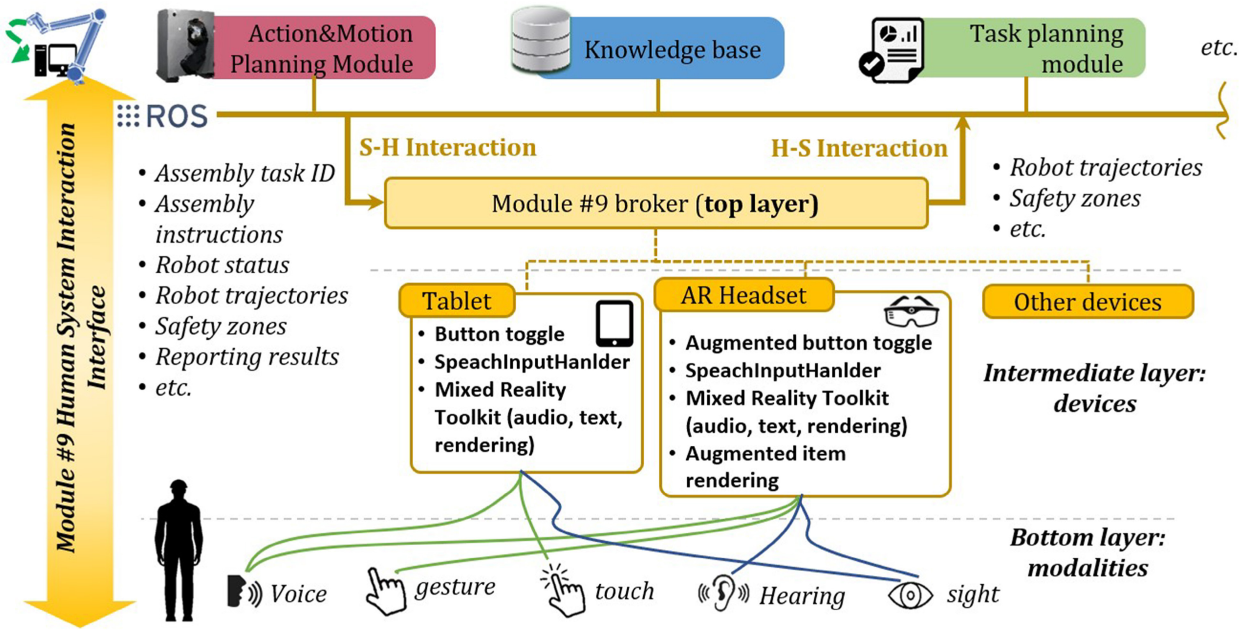


Figure 6 — SHAREWORK HRC's Human-System interaction module architecture

Annex A
(informative)

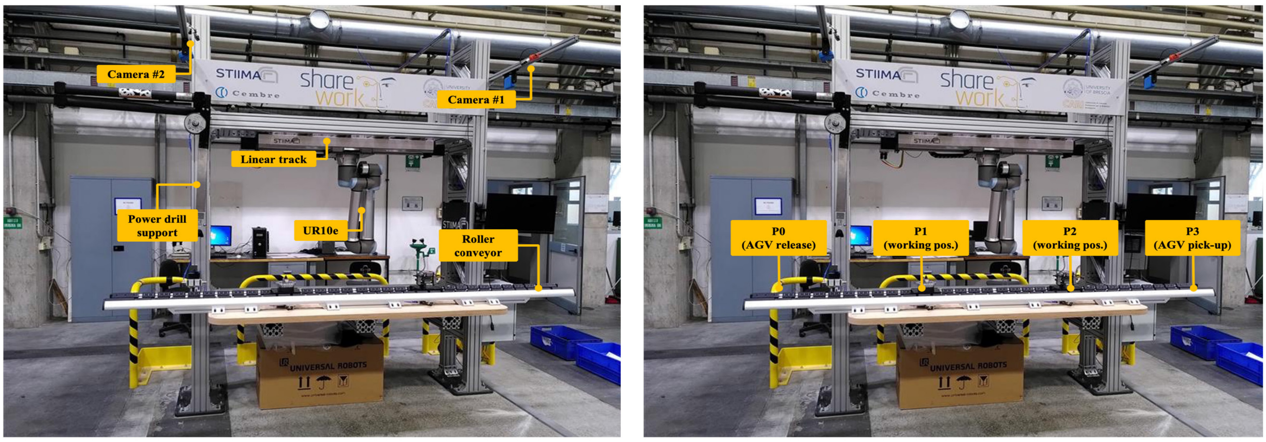
Integration and Deployment: use case

The proposed approach has been demonstrated in the SHAREWORK project (‘*Safe and effective human-robot cooperation towards a better competitiveness on current automation lack manufacturing processes*,’ funded by the European Union’s Horizon 2020 Framework Programme for Research and Innovation under Grant Agreement No 820807), in a case study derived from a mechanical machining scenario.

The case study is characterized by unpredictable market changes in terms of demand, which require massive use of Flexible Manufacturing Systems (FMSs) to remain highly competitive in the market [51]. In FMSs, parts to be machined are mounted on multi-fixturing devices called pallets. Pallets are manually assembled at a loading/unloading station (LUS) and moved from/to general-purpose machine centers to be machined. The number of pallet configurations, *i.e.*, pallet mounting clamping systems/jigs, and products present simultaneously in an FMS can be considerable. Due to the high number of different operations to be performed on the pallets, LUSs influence FMS performance in terms of final throughput. Specifically, three critical operations at LUS are performed: assembly, disassembly, and quality inspection. The application is stimulating for human-robot collaboration because the process throughput would benefit from juxtaposing humans’ manipulation skills and robots’ tirelessness. For example, robots could be exploited to perform batches of simple, repetitive operations, while a human could perform the most complex operations and perform quality checks. Note that no fixed scheduling usually applies [52]; a dynamic online reconfiguration of the workflow is required by operators who may change sequences and roles.

Consideration is given to the scenario depicted in Figure 7 in this context. A collaborative LUS is composed of a small-size robot, Universal Robots UR10e, mounted on a linear track to extend its range of motion. The LUS owns four pallet positions: P0 is the arrival position of a new pallet brought by a mobile robot; P1 and P2 are the working position, where the pallets are mounted, unmounted, and checked; P3 is the departure position, where an AGV will load the mounted pallets to move it to next stage of the process. The robot and the human operator can work simultaneously at the LUS, either on the same pallet or two different pallets. The process requires the following stages:

1. A mobile robot brings a new pallet to P0



a) b)

Figure 7 — Demonstrator setup and description of the work positions: a) Setup components; b) Setup working positions.

2. The pallet is moved to a free position (P1 or P2). Notice that the pallet can be moved to P2 only if P1 is not occupied.

3. The pallet is unmounted to extract the finished part.

4. A new raw part is inserted, and the pallet is mounted.

5. The pallet is moved to P3.

6. A mobile robot picks up the pallet from P3.

Steps 2 and 5 are always performed by the human operator because the robot is not able to lock/unlock and move the pallet. Steps 3, 4, and 5 can be performed by both the robot and the human. Notice that more than up to four pallets can be present at the LUS at the same time, meaning that Steps from 2 to 5 can be performed without a fixed scheduling and assignment wither by the human or the robot. Moreover, pallets can have different geometries and therefore requires different operations to be mounted and unmounted. In this case study, three different types of pallets requiring high flexibility in the planning and execution phases are examined.

A.1 Process Representation in the Knowledge Base

To successfully coordinate human and robot operations, it is necessary to configure the *Knowledge Base* module of Figure 2 first. This configuration step allows the system to build an abstraction of production procedures characterizing the specific needs/requirements of an HRC cell and the specific skills and features of participating acting agents. To this aim, an ontological model of the scenario is manually defined using Protege[1)]. It is defined individuals and assert properties necessary to characterize the (relevant) information about the production environment and the capabilities of the agents that take part in the process.

The main elements of the environment are the workpieces (i.e., *pallets*), the worker and the cobot agents, and the positions they occupy while performing operations. Workpieces can be of three types entailing different geometric constraints and low-level operations for their manipulation. These workpieces are thus modelled separately as three distinct instances of Workpiece: (i) 0218; (ii) 1121 and; (iii) 1122. This distinction supports the contextualization of production procedures according to the particular type of workpiece to be worked. During the execution of a production procedure, each workpiece occupies a specific environment location. In the considered scenario, such environmental locations are subject to *physical constrains* limiting the number of objects that can occupy them simultaneously. They are thus modelled as BinaryProductionLocation that are ProductionLocation associated with a ResourceCapacity which limits to 1 the number of ProductionObjects that can stay at the same location simultaneously (i.e., these locations are characterized by a *binary* state denoting the location as *free* or *busy*).

Each type of workpiece is associated with a ProductionGoal specifying a different ProductionMethod and different production operations. Such production operations are defined as individuals of ProductionTask. The knowledge base describes *operational constraints* and *alternative decomposition* of tasks as well as *alternative assignments* to the human and to the robot. In this regard, individuals of DisjunctiveTask describe alternative way of implementing/decomposing a ProductionTask. For example, the general task process 1121 is modelled as DisjunctiveTask and is associated with two alternative sub-tasks through the property DUL:hasConstituent: (i) process 1121 p1 and; (ii) process 1121 p2. Both sub-tasks are instances of ConjunctiveTask and represent two alternative ways of performing the production

1) A well-known editor for ontologies and knowledge bases - https://protege.stanford.edu

task process 1121 p2: (i) perform production operations for workpiece 1121 on position1 and; (ii) perform production operations for workpiece 1121 on position2. The actual choices would be made dynamically by a task planner depending on previously scheduled operations and the *known* state of physical locations/positions of the HRC cell.

A similar decomposition is defined for low-level tasks that can be assigned to the human or to the robot. An example is the operation requiring to mount the pallet 1121 in a specific position of the cell. The DisjunctiveTask mount 1121 p2 is decomposed into two (alternative) simpler ProductionTask that are: (i) mount 1121 p2 worker and; (ii) mount 1121 p2 cobot. This disjunction characterizes the alternative of assigned the mounting task to the worker or to the robot. The two sub-tasks are both instances of IndependentTask meaning two individuals of SimpleTask associated with a CollaborationModality of type Independent. Following the ontological definition of *independent collaborative tasks*, they are respectively decomposed into a HumanFunction and a RobotFunction of type Assembly representing the actual operations performed on the workpiece.

The defined knowledge base completely characterizes the production process and can be used to configure the planning and interaction modules deployed into the scenario. A designed *knowledge extraction procedure* automatically generates *contextualized* timeline-based specifications for: (i) hierarchical decomposition and planning constraints concerning known goals and; (ii) temporal dynamics and controllability properties associated with robot and worker capabilities. Such specification provides the Task Planner with the rules to compute collaborative plans for the considered manufacturing scenario at hand. A graph-based description of production procedures is automatically extracted from the knowledge base and used to generate a suitable timeline-based task planning model [19, 20]. The resulting production procedure is organized into several *hierarchical levels* correlating high-level *production goals* with low-level tasks and individuals of Function the human and the robot should perform to conduct related production processes correctly. The following section describes with further detail the timeline-based model and provides an example of a plan.

A.2 Task Planning and Scheduling

A timeline-based task planning model is synthesized by the knowledge base to "operationalize" production procedures and coordinate human and robot behaviors. A number of *state variables* are defined to characterize states and/or actions that relevant domain features assume and/or perform over time. Four state variables $SV_{p0}, SV_{p1}, SV_{p2}, SV_{p3}$ describe the state of the working positions of the pallets. Since these physical locations are modelled as BinaryProductionLocation in the knowledge base, these variables are associated with two values, $V_{p0}, V_{p1}, V_{p2}, V_{p3}$ = {*Free,Busy*}. Then, transitions T_{p0}(*Free*), T_{p1}(*Free*), T_{p2}(*Free*) are defined, T_{p3}(*Free*) = *Busy*, T_{p0}(*Busy*), T_{p1}(*Busy*), T_{p2}(*Busy*), T_{p3}(*Busy*) = *Free* and duration (1,+∞) for all of them. These state variables are used to encode *binary resource constraints* and, thus, enforce a mutually exclusive use of the associated physical locations.

Other two state variables describe behaviors of the human and the robot in terms of the set of Function that they can perform over time. These functions are the *inferred* instances of low-level operations the human and the robot can perform in the considered scenario. The state variable of the robot SV_R is thus an associated with the set of values denoting the function it is supposed to perform V_R = {*Idle, Release* $p1$, *Release* $p2$, *Pick* $p1$, *Pick* $p2$, *Assembly* $p1$, *Assembly* $p2$, *Disassembly* $p1$, *Disassembly* $p2$}. The values *Release* $p1$, *Release* $p2$ and *Pick* $p1$, *Pick* $p2$ are instances of the function PickPlace and denote respectively the operations of removing a worked piece from the pallet (i.e., *release piece*) and placing a new raw piece into the pallet (i.e., *pick piece*). The transition function requires all value changes to pass through the idle state as follows: T_R (*Idle*) ∈ {*Release* $p1$, ..., *Disassembly* $p2$}; T_R (*Release* $p1$) = {*Idle*}; ...; T_R (*Disassembly* $p2$) = {*Idle*}. All the values of the robot $v_{R,i} \in V_R$ are tagged as *partially controllable* (γ_R ($v_{R,i}$) = *pc*) because the actual duration of their execution can interfere with the worker. The duration bounds of these values instead is set according to the average observed execution time. The state variable of the human SV_H is structured similarly to SV_R. In this case, it is necessary to consider the additional operations the worker can perform, and robot cannot. These are modelled with additional values V_H = {*PickPlace* $p0p1$, *PickPlace* $p1p2$, *PickPlace* $p2p3$,...}. The value transition function follows the same "pattern" of SV_R. However, in this case, all the values $v_{H,i} \in V_H$ of the state variable SV_H are tagged as *uncontrollable* (γ_H ($v_{H,i}$) = u) since the system cannot control the behaviour of the worker. Furthermore, the duration bounds of the values are defined according to Equation 4 and thus they depend on both the average duration of their execution and on the *uncertainty index* δ set according to the *expertise level* of the worker.

To synthesize production operations, it is necessary to define "functional" state variables encapsulating abstract production tasks. Such state variables are directly associated with the production procedure extracted from the knowledge base. The actual number of these variables (and their values) depend on the complexity of the modelled procedure. In general, each "production" state variable is associated with a specific *abstraction level* of the extracted hierarchical procedure. A *goal state variable* SV_G encapsulates high-level production requests and is associated with the individuals of ProductionGoal. Individuals of this concept are root elements of the production procedure and are mapped to the values of SV_G. In this case there are three diverse types of goals, each associated to a particular type of pallet V_G = {*process* 1121, *process* 1122, *process* 0218}. Three different hierarchical procedures correspond to these three goals. These values are all *controllable* (γ ($v_{G,i}$) = c) and do not have specific duration bounds since their actual duration depends on the planning and scheduling of underlying human and robot operations. Intermediate N −1 levels of the procedure are modelled through "production state variables" SV_{L1}, ..., SV_{LN-1}. The last hierarchical level (N) of the decomposition entails individuals of Function that are already represented through SV_R and SV_H. The values of production state variables represent individuals of ProductionTask such as *unmount* 1121 $p2$, *mount* 1121 $p2$ and thus complex/abstract production operations that need to be further decomposed in simpler ones. Starting with high-level production requests (i.e., values of the goal state variable $v_j \in V_G$) task decomposition and the needed causal and temporal constraints are modelled through a set of *synchronization rules*. Each rule has individuals of ProductionTask (i.e., values $v_{Li} \in V_{Li}$) as trigger (i.e., the *head* of the rule). Individuals of DisjunctiveTask are triggers of different rules in order to model alternative decomposition.

The Task Planning Module implements *goal-oriented acting* capabilities using the open-source ROSJava Package ROXANNE[2]). Once configured, the module is ready to receive production requests (i.e., *planning goal*) through a dedicated input topic. The synthesis of a task plan consists in deciding the assignment of production tasks to the human and the robot that best takes advantage of the collaboration (i.e., optimize the production process) in the given scenario [7, 28, 39]. The resulting assignment is then online dispatched to the human and to the robot by sending task execution requests respectively through the Human-System Interaction Module and the Motion Planning Module (see Figure 2).

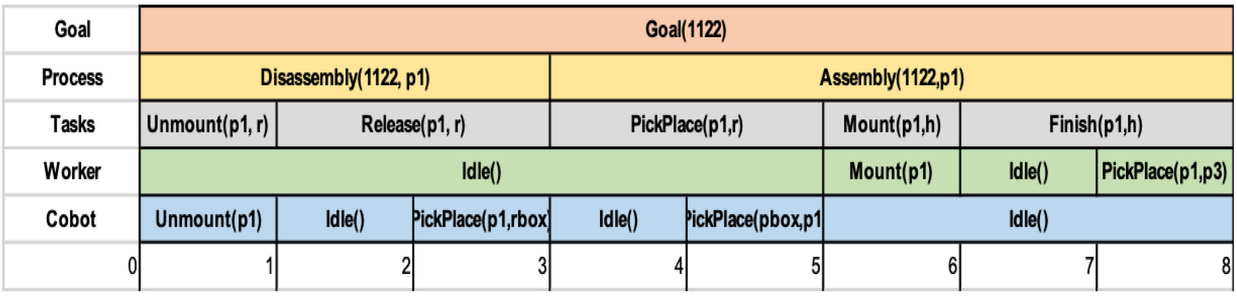


Figure 8 — Simplified view of a plan synthesized for the execution of a collaborative process concerning Workpiece 1122 – It shows the timelines synthesized for each state variable of the

2) https://github.com/pstlab/roxanne_rosjava.git

task planning model (i.e., *Goal*, *Process*, *Tasks*, *Worker* and *Cobot* state variables) with the scheduling of related tokens

Figure 8 shows an example of a timeline-based plan. It specifically shows the timelines of the plan through a Gantt representation depicting *tokens* planned for the state variables of the domain and their allocation over time. Note that this Gantt representation shows a specific instance of the plan called the *earliest start time*. Timelines indeed encapsulate an envelope of possible temporal behaviors (i.e., plan instances) through *temporal flexibility* [26]. This flexibility is crucial to deal with *temporal uncertainty* and support *reliable execution of timelines* in real environments [39, 53].

A.3 Action Planning and Execution

High-level tasks dispatched by the task planner module are put in place by the action planning module. This module converts symbolic tasks into a sequence of robot movements and tool operations (*e.g.*, open/close gripper). The *Task Decomposition* module receives the task from the task planner and queries a database to decode the type of the task and its geometrical properties. The type of task and its properties determine the set of operations that a task requires.

When a task request comes from the task planner, the Task Decomposition module converts it into a set of basic operations. For example, task *mount* 1121 $p2$ boils down to: (a) move to P_2 approach position; (b) approach nut; (c) unscrew nut (activate power drill); (d) push locking bracket; (e) move to piece grasping pose; (f) close gripper; (g) move to unloading box; (h) open gripper. Each operation corresponds to a point-to-point robot movement or a change in the state of the robot's auxiliaries (e.g., the gripper and the power drill). For all robot's movements, the *Task Decomposition* sends the decomposed actions to a motion planning algorithm. When the task is executed, it returns the outcome to the task planner. If the task is successful, the task planner will dispatch the next task in the plan. Otherwise, it would replan according to the reported error. Notice that, since the proposed framework is designed for dynamic environments, task decomposition and motion planning are performed online, based on the current state of the cell.

The *Task Decomposition* module was developed in C++ and Python 3 within the ROS framework. The communication with the task planner is managed by a ROS-action server that receives the tasks from the task planner and queries a MongoDB database to retrieve the task properties. The planning and execution phases are managed by *manipulation* framework, an open-source library that implements basic skills [54]. The *manipulation* framework uses *MoveIt!* planning pipeline and planning scene. Thanks to *MoveIt!*'s plugin-based architecture, it is possible to load motion planners dynamically from state-of-the-art libraries available in *MoveIt!* (e.g., OMPL, CHOMP, and STOMP). In this work, a human-aware path planner [55] is utilized, which accounts for the position of the operator in the cell, according to Section 6.2.2. The *manipulation* framework is also modular with respect to the controller. In this work, the human-aware reactive speed modulation module is implemented (see Section 6.2.3) as a ROS controller that changes the robot speed according to the human-robot relative distance. This allows for a real-time implementation with a sampling rate equal to that of the robot controller (500 Hz), ensuring prompt reaction of the robot motion.

A.4 Human-System and System-Human Interaction

For this industrial case, the HS-SH interaction module was deployed by spawning two applications, hosted on Augmented Reality Headset Microsoft HoloLens 2 and Android Tablet Samsung Galaxy S4 (Figure 9). Voice, gesture, touch, hearing, and sight-related modalities are available during operation, either for direct system control or for worker support.

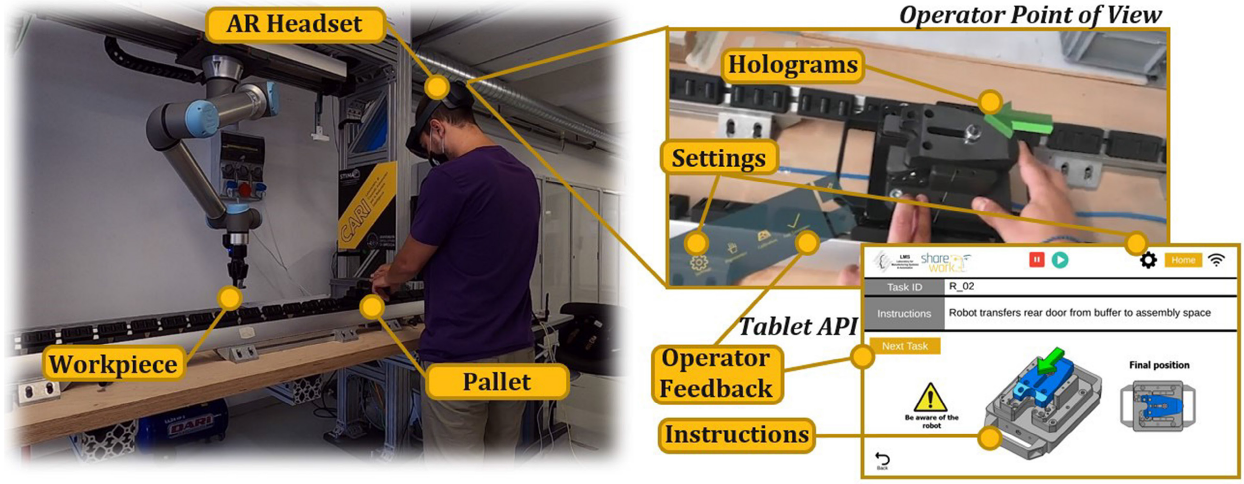


Figure 9 — Demonstrator setup and HS-SH interaction module

The Knowledge Base configures the type of modals and features within the application environments according to the operator's level of expertise. Online customization options are offered to users to maximize personalization thanks to a several options for each feature. Authentication via operator profiles ensures that user models are updated with the customization settings and are linked to each operator.

The human worker and the robotic arm are aware of each other through bilateral communication of information about each agent's actions. More specifically, the user can press easy-to-use buttons to send feedback to the Task planner about the successful execution of a human task or action. On the contrary, the robot's status is broadcasted via textual panels in addition to visualized robot trajectories in 3D augmented-reality (i.e., AR headset) or 2D screen-based (i.e., tablet) formats, as planned by the Motion Planner. Awareness about robot actions is also promoted via audio notifications that are enabled upon robot movements.

The implemented interaction module also supports users during manufacturing operations through intuitive instructions in extensive or plain form, depending on their preferences and expertise. The AR application augments the physical system by visualizing digital assistive content within the workstation (Figure 10). In detail, 3D augmented models and arrows (static or moving) instruct the operator on how to manipulate related components toward successful assembly. On the same basis, the tablet application provides assistive figures. In parallel, task information panels are filled by the "Task planner", providing Task id, name, remaining tasks, and instructions about current operation in both applications.

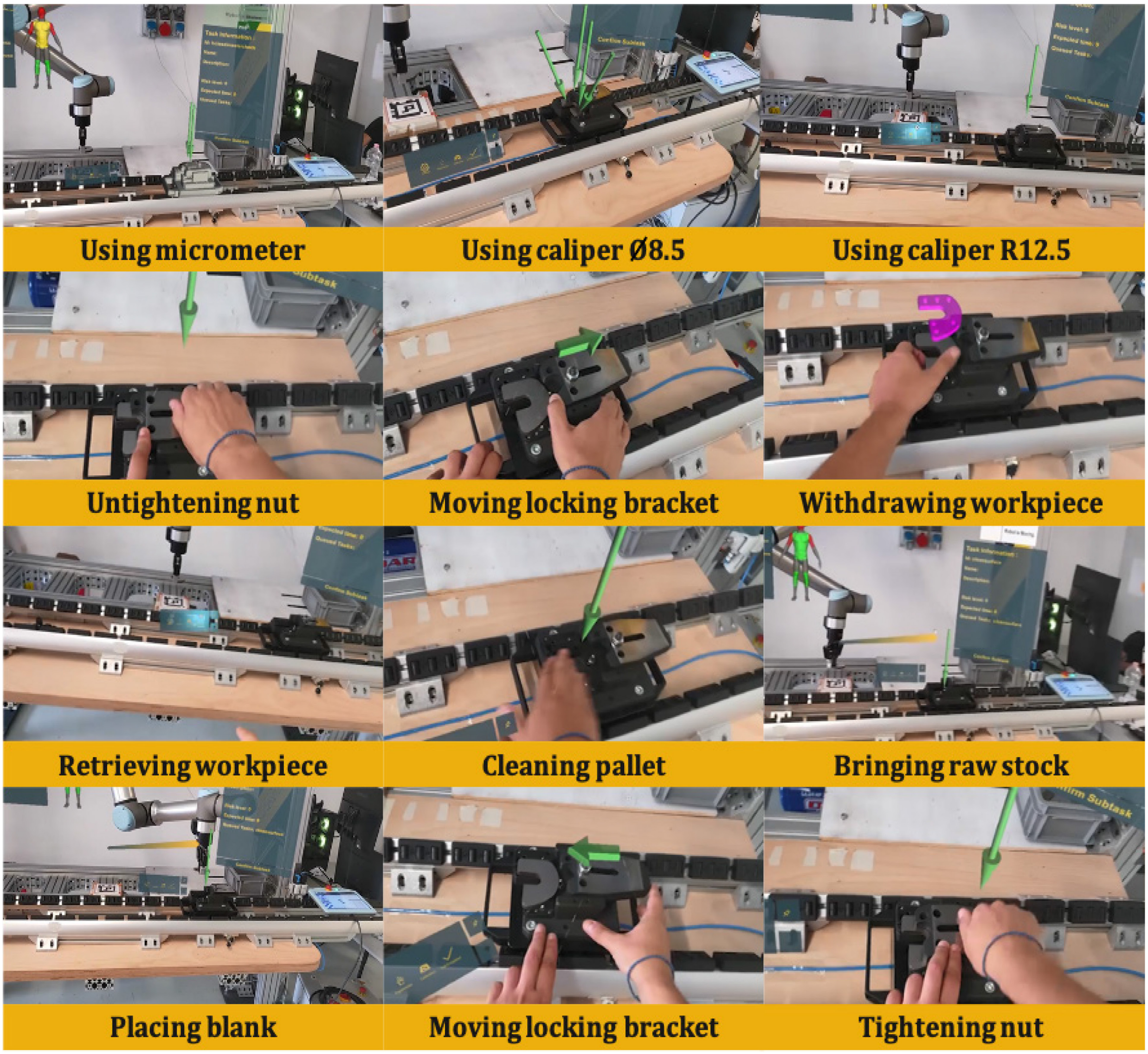


Figure 10 — Operator point of view for indicative tasks (AR application)

Bibliography

[1] Anil Kumar Inkulu, MVA Raju Bahubalendruni, Ashok Dara, and K SankaranarayanaSamy. Challenges and opportunities in human robot collaboration context of industry 4.0-a state of the art review – Industrial Robot: the international journal of robotics research and application, 2021

[2] Bernd Dworschak and Helmut Zaiser – Competences for cyberphysical systems in manufacturing – first findings and scenarios. Procedia CIRP. 2014, 25 pp. 345–350

[3] Thomas B. Sheridan – Human–robot interaction: status and challenges. Hum. Factors. 2016, 58 (4) pp. 525–532

[4] Evangelou G., Dimitropoulos N., Michalos G., Makris S. – An approach for task and action planning in human–robot collaborative cells using ai –. Procedia CIRP. 2021, 97 pp. 476–481

[5] Fusaro F., Lamon E., De Momi E., Ajoudani A. An integrated dynamic method for allocating roles and planning tasks for mixed human-robot teams. In IEEE International Conference on Robot & Human Interactive Communication, pages 534–539, 2021

[6] Ali Ahmad Malik and Arne Bilberg. Complexity-based task allocation in human-robot collaborative assembly. Industrial Robot: the international journal of robotics research and application, 2019

[7] Faroni M., Beschi M., Ghidini S., Pedrocchi N., Umbrico A., Orlandini A. et al. A layered control approach to human-aware task and motion planning for human-robot collaboration – In IEEE International Conference on Robot and Human Interactive Communication, 2020

[8] Przemyslaw A Lasota and Julie A Shah. Analyzing the effects of human-aware motion planning on close-proximity human–robot collaboration. Hum. Factors. 2015, 57 (1) pp. 21–33

[9] Makris S. Cooperating robots for flexible manufacturing. Number 1. Springer International Publishing, 2021

[10] Umbrico A., Orlandini A., Cesta A., Koukas S., Zalonis A., Fourtakas N. et al. Towards user-awareness in human-robot collaboration for future cyber-physical systems – In IEEE International Conference on Emerging Technologies and Factory Automation, pages 1–8, 2021

[11] Umbrico A., Orlandini A., Cesta A. – An ontology for human-robot collaboration –. Procedia CIRP. 2020, 93 pp. 1097–1102

[12] Aurélie C., Pacherie E., Alami R., Chatila R. – Key elements for human-robot joint action, pages 159–177 – Springer International Publishing, Cham, 2017

[13] Cristiano Castelfranchi. Modelling social action for ai agents – Artificial Intelligence, 103(1):157-182, 1998 – Artificial Intelligence 40 years later

[14] Devin S., Alami R. An implemented theory of mind to improve human-robot shared plans execution – In ACM/IEEE International Conference on Human-Robot Interaction (HRI), pages 319– 326, 2016

[15] Auer S¨., Kovtun V., Prinz M., Kasprzik A., Stocker M., Vidal M.E. Towards a knowledge graph for science – In Proceedings of the International Conference on Web Intelligence, Mining and Semantics. Association for Computing Machinery, 2018

[16] Xiaojun Chen, Shengbin Jia, and Yang Xiang – A review: Knowledge reasoning over knowledge graph –. Expert Syst. Appl. 2020, 141 p. 112948

[17] Antoniou G., van Harmelen F. Web Ontology Language: OWL, pages 67–92 – Springer Berlin Heidelberg, Berlin, Heidelberg, 2004

[18] S. Borgo, M. Carrara, P. Garbacz, and P.E. Vermaas – A formal ontological perspective on the behaviors and functions of technical artifacts –. Artif. Intell. Eng. Des. Anal. Manuf. 2009, 23 (1) pp. 3–21

[19] Stefano Borgo, Amedeo Cesta, Andrea Orlandini, and Alessandro Umbrico – Knowledge-based adaptive agents for manufacturing domains –. Eng. Comput. 2019, 35 (3) pp. 755–779

[20] S. Borgo, A. Cesta, A. Orlandini, and A. Umbrico – A planningbased architecture for a reconfigurable manufacturing system – In ICAPS, the 26th International Conference on Automated Planning and Scheduling, 2016

[21] Richard E Fikes and Nils J Nilsson – Strips. A new approach to the application of theorem proving to problem solving –. Artif. Intell. 1971, 2 (3-4) pp. 189–208

[22] Drew Mcdermott, Malik Ghallab, Adele Howe, Craig Knoblock, Ashwin Ram, Manuela Veloso, Daniel Weld, and David Wilkins – PDDL – The Planning Domain Definition Language – Technical report, CVC TR-98-003/DCS TR-1165, Yale Center for Computational Vision and Control, 1998

[23] Fox M., Long D. – PDDL2.1: an extension to PDDL for expressing temporal planning domains –. J. Artif. Intell. Res. 2003, 20 pp. 61–124 [JAIR]

[24] Ghallab M., Nau D., Traverso P. – The actor’s view of automated planning and acting: A position paper –. Artif. Intell. 2014, 208 pp. 1–17

[25] Félix Ingrand and Malik Ghallab – Deliberation for autonomous robots: A survey –. Artif. Intell. 2017, 247 pp. 10–44 [– Special Issue on AI and Robotics]

[26] Marta Cialdea Mayer, Andrea Orlandini, and Alessandro Umbrico – Planning and execution with flexible timelines: a formal account –. Acta Inform. 2016, 53 (6-8) pp. 649–680

[27] Alessandro Umbrico, Amedeo Cesta, Marta Cialdea Mayer, and Andrea Orlandini – PLATINUm: A New Framework for Planning and Acting – Lecture Notes in Computer Science, pages 498–512, 2017

[28] Amedeo Cesta, Andrea Orlandini, and Alessandro Umbrico – Fostering robust human-robot collaboration through ai task planning – Procedia CIRP, 72:1045–1050, 2018 – 51st CIRP Conference on Manufacturing Systems

[29] A. Umbrico, A. Cesta, M. Cialdea Mayer, and A. Orlandini – Integrating resource management and timeline-based planning – In The 28th International Conference on Automated Planning and Scheduling (ICAPS), 2018

[30] Johnson M., Bradshaw J.M., Feltovich P.J., Jonker C.M. M. Birna van Riemsdijk, and Maarten Sierhuis – Coactive design: Designing support for interdependence in joint activity –. J. Hum. Robot Interact. 2014, 3 (1) pp. 43–69

[31] Javier Barreiro, Matthew Boyce, Minh Do, Jeremy Frank, Michael Iatauro, Tatiana Kichkaylo, Paul Morris, James Ong, Emilio Remolina, Tristan Smith, and David Smith – EUROPA: A Platform for AI Planning, Scheduling, Constraint Programming, and Optimization – In ICKEPS 2012: the 4th Int. Competition on Knowledge Engineering for Planning and Scheduling, 2012

[32] Cesta A., Cortellessa G., Denis M., Donati A., Fratini S., Oddi A. et al. – MEXAR2: AI Solves Mission Planner Problems –. IEEE Intell. Syst. 2007, 22 (4) pp. 12–19

[33] Kanna Rajan and Alessandro Saffiotti – Towards a science of integrated AI and Robotics –. Artif. Intell. 2017, 247 pp. 1–9

[34] MVA Raju Bahubalendruni and Bibhuti Bhusan Biswal – A novel concatenation method for generating optimal robotic assembly sequences –. Proc. Inst. Mech. Eng., C J. Mech. Eng. Sci. 2017, 231 (10) pp. 1966–1977

[35] G Bala Murali, BBVL Deepak, MVA Raju, and BB Biswal – Optimal robotic assembly sequence planning using stability graph through stable assembly subset identification –. Proc. Inst. Mech. Eng., C J. Mech. Eng. Sci. 2019, 233 (15) pp. 5410–5430

[36] Alessandro Umbrico, Amedeo Cesta, Marta Cialdea Mayer, and Andrea Orlandini – Evaluating robustness of an acting framework over temporally uncertain domains – In Mario Alviano, Gianluigi Greco, and Francesco Scarcello, editors, AI*IA 2019 – Advances in Artificial Intelligence, pages 250–263, Cham, 2019 – Springer International Publishing

[37] Vidal T. A unified dynamic approach for dealing with temporal uncertainty and conditional planning – In AIPS-00 – Proc. of the Fifth Int. Conf. on Artificial Intelligence Planning and Scheduling, pages 395–402, 2000

[38] A. Cesta, A. Orlandini, G. Bernardi, and A. Umbrico – Towards a planning-based framework for symbiotic human-robot collaboration – In 21th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA) – IEEE, 2016

[39] Stefania Pellegrinelli, Andrea Orlandini, Nicola Pedrocchi, Alessandro Umbrico, and Tullio Tolio – Motion planning and scheduling for human and industrial-robot collaboration – CIRP Annals –. Manufacturing Technology. 2017, 66 pp. 1–4

[40] Ren W., Xiaonan Yang Y., Hu Y. – The decision-making framework for assembly tasks planning in human– robot collaborated manufacturing system –. Int. J. Comput. Integrated Manuf. 2022, ••• pp. 1–19

[41] Hung Pham and Quang-Cuong Pham – A new approach to timeoptimal path parameterization based on reachability analysis –. IEEE Trans. Robot. 2018, 34 (3) pp. 645–659

[42] Mohamed Elbanhawi and Milan Simic – Sampling-based robot motion planning: A review –. IEEE Access. 2014, 2 pp. 56–77

[43] Tarbouriech S., Suleiman W. – Bi-objective motion planning approach for safe motions: Application to a collaborative robot –. J. Intell. Robot. Syst. 2020, 99 pp. 45–63

[44] Hayne R., Luo R., Berenson D. Considering avoidance and consistency in motion planning for human-robot manipulation in a shared workspace – In 2016 IEEE International Conference on Robotics and Automation (ICRA), pages 3948–3954, 2016

[45] E. A. Sisbot and R. Alami – A human-aware manipulation planner –. IEEE Trans. Robot. 2012, 28 (5) pp. 1045–1057

[46] Christoph Byner, Björn Matthias, and Hao Ding – Dynamic speed and separation monitoring for collaborative robot applications-concepts and performance –. Robot. Comput.-Integr. Manuf. 2019, 58 pp. 239–252

[47] Faroni M., Beschi M., Pedrocchi N. – An MPC framework for online motion planning in human-robot collaborative tasks – In Proceedings of the IEEE Int. Conf. on Emerging Tech. and Factory Automation, Zaragoza (Spain), 2019

[48] ISO/TS 15066:2016, *Robots and robotic devices — Collaborative robots — Technical report, International Organization for Standardization, Geneva, CH, 2016*

[49] Tsarouchi P., Makris S., Chryssolouris G. Human–robot interaction review and challenges on task planning and programming. Int. J. Comput. Integrated Manuf. 2016, 29 (8) pp. 916–931

[50] Dionisis Andronas, George Apostolopoulos, Nikos Fourtakas, and Sotiris Makris – Multi-modal interfaces for natural human-robot interaction –. Procedia Manuf. 2021, 54 pp. 197–202

[51] Hakan Gultekin, Ozden O. Dalgı¸c, and M. Selim Akturk – Pure cycles¨ in two-machine dual-gripper robotic cells –. Robot. Comput.-Integr. Manuf. 2017, 48 pp. 121–131

[52] Foumani M., Gunawan I., Smith-Miles K. – Resolution of deadlocks in a robotic cell scheduling problem with post-process inspection system: Avoidance and recovery scenarios – In 2015 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), pages 1107–1111, 2015

[53] A. Orlandini, M. Suriano, A. Cesta, and A. Finzi – Controller synthesis for safety critical planning – In IEEE International Conference on Tools with Artificial Intelligence, pages 306–313, 2013

[54] Villagrossi E., Pedrocchi N., Beschi M. Simplify the robot programming through an action-and-skill manipulation framework – In IEEE International Conference on Emerging Technologies and Factory Automation, pages 1-6, 2021

[55] Cesare Tonola, Marco Faroni, Nicola Pedrocchi, and Manuel Beschi – Anytime informed path re-planning and optimization for humanrobot collaboration – In IEEE International Conference on Robot Human Interactive Communication (RO-MAN), pages 997-1002, 2021

[56] Gianfranco E. Modoni, Enrico G. Caldarola, Marco Sacco, and Walter Terkaj – Synchronizing physical and digital factory: benefits and technical challenges – Procedia CIRP, 79:472-477, 2019 – CIRP Conference on Intelligent Computation in Manufacturing Engineering, 18-20 July 2018, Gulf of Naples, Italy

[57] Schroeder G.N., Steinmetz C., Pereira C.E., Danubia B. Espindola – Digital twin data modeling with automationml and a communication methodology for data exchange –. IFAC-PapersOnLine. 2016, 49 (30) pp. 12–17

[58] Michael Schluse and Juergen Rossmann – From simulation to experimentable digital twins: Simulation-based development and operation of complex technical systems – In 2016 IEEE International Symposium on Systems Engineering (ISSE), pages 1-6, 2016

[59]Zhihao Liu, Xinran Wang, Yijie Cai, Wenjun Xu, Quan Liu, Zude Zhou, and Duc Truong Pham – Dynamic risk assessment and active response strategy for industrial human-robot collaboration –. Comput. Ind. Eng. 2020, 141 p. 106302