The International Journal of Advanced Manufacturing Technology (2023) 128:5249–5260
https://doi.org/10.1007/s00170-023-12280-6

ORIGINAL ARTICLE

From CAD to Plug & Produce

A generic structure for the integration of standard industrial robots into agents

Anders Nilsson[1] · **Fredrik Danielsson**[1] · **Bo Svensson**[1]

Received: 18 April 2023 / Accepted: 30 August 2023 / Published online: 7 September 2023

Abstract
Industries of low-batches or one-off manufacturing aim for automation that is competitive enough to adapt to new or modified products daily through in-house knowledge that focuses on manufacturing processes and not on machine function programming. To solve this, a complete set of actions that utilize seamless data transfer from product design in CAD to a Plug & Produce automation concept is proposed together with a generic structure for the integration of standard industrial robots into agents. This structure enables agents to handle their local reference coordinate systems and locations relative to a global perspective. Seamless utilization of data from product designs to Plug & Produce will simplify and shorten the time of digital development through concurrently usable text-based and graphical configuration tools of a configurable multi-agent system. Needed data extracts directly from the product design as requirements of operational goals. Extraction of data from the product design, sequence of goals, and process plans, which are recipes of how to solve goals, can by this concept be configured by in-house knowledge that has the process knowledge but not necessarily programming competence.

Keywords Robotics · Automation · Manufacturing · Multi-agent systems · Plug & Produce · Process planning

1 Introduction

One-off or low-batches manufacturing is forced into manual manufacturing due to inflexible automation alternatives that have too long and costly change-over-time. Conventional industrial control systems are flexible by reprogramming and therefore require skilled programmers. Skilled programmers are sought after and therefore entail a high cost and a risk of shortage. Hence, reconfigurable and by design flexible control systems are questioned [1]. However, the situation in the industry is that a high degree of flexibility in automation systems requires a high degree of competence to handle [2]. This article aims to change this relationship by expanding an existing Plug & Produce concept, controlled by the Configurable Multi-Agent System (C-MAS) [3]. C-MAS has been proven to be efficient in adapting to changed manufacturing. Standard industrial robots are, because of the existing programming structure of the control systems, hard to handle in a Plug & Produce concept. A generic structure for the integration of standard industrial robots into agents and a generic extraction of data directly from the product design for seamless use in a Plug & Produce manufacturing system is the main contribution of this article. The proposed method is generic and applicable to any kind of product and automated manufacturing process on the cell/workstation level. A common approach in the literature is to use multi-agent for the orchestration of processes and neglect the low-level integration. This article contributes to a low-level integration of standard industrial robots into agents. The whole chain of data and set of actions from Computer-Aided Design (CAD) to a Plug & Produce automated manufacturing system is covered and described. The aim of the proposed control system, C-MAS, is that a Plug & Produce automation concept should be adaptable by utilizing only existing in-house knowledge within the manufacturing company that focuses

✉ Anders Nilsson
anders.nilsson@hv.se

Fredrik Danielsson
fredrik.danielsson@hv.se

Bo Svensson
bo.svensson@hv.se

[1] Engineering Science, Department of Production Systems, University West, Gustava Melins gata 2, 461 32 Trollhättan, Sweden

on manufacturing processes rather than the programming of machine functionalities.

Multi-Agent Systems (MAS) are proposed by several researchers as a strategy for solving flexible robust automated manufacturing. MAS consists of a network of computational distributed intelligence implemented as agents that have strategies to achieve their goals. A common base for strategies is to utilize reasoning, Belief Desire Intention (BDI) mixed with planning and learning functionality [4]. Agents within automated manufacturing strive to change their surroundings to fulfill their goals based on real-time data directly from sensors and information communicated from other agents. An individual agent can solve holistic and common problems through cooperation and negotiations among surrounding agents. C-MAS is a configurable multi-agent structured framework for automated manufacturing where software-based part agents contain the manufacturing goals for the physical parts. A part is in this article considered as a part of the manufactured product or the product itself if the product only includes one part. Resource agents reform parts according to the goals by utilizing their skills. Goals are organized in sequence or parallel, configured by a graphical tool named sequence of goals chart. Parallel goals will execute concurrently on resources if available resources exist otherwise sequentially by one resource [3]. The manufacturing capacity can thereby be balanced by plugging in and out concurrent resources. For each goal will the part agent select and execute the most suitable and executable process plan by matching names and selecting the process plan with the lowest cost. A process plan specifies how to achieve a goal by utilizing skills on resources without specifying which resource to use. This is handled by abstract interfaces which enable loosely coupled skills of resource agents. Further, abstract interfaces allow for digital configuration of the automation without having a full understanding and knowledge of all resources that will be engaged when the manufacturing starts. The abstract interfaces will be connected to specific resource agents during runtime through negotiation. This arrangement will ease the digital configuration by enabling the possibility of having an operator or process engineer who has the process focus to do the digital configuration, not machine specialists.

Most manufacturing companies, independent of branches, use 3D-CAD models of the parts they manufacture. A 3D-CAD model contains detailed data such as locations, dimensions, and tolerances which are requirements that the manufacturing process must consider. Data extracted directly from CAD without any translation and reentering will simplify the planning process, avoid typing errors, and reduce the time from design to manufacturing [5]. The method of extracting data from CAD that is presented in this article is generic and can be applied as a requirement of the goals of any multi-agent system.

The concept of Plug & Produce was initiated in the late 1990s by Arai et al. [6]. Plug & Produce supports a quick and seamless connection of production equipment with minimal or no digital reconfiguration [7]. Plug & Produce controlled by C-MAS consists of pluggable process modules often located around a standard industrial robot with automatically exchangeable robot tools [8]. Process modules are standardized pluggable modules that carry a process, exchangeable between different Plug & Produce cells/workstations which enables sustainable and cost-efficient solutions. Process modules can easily be reused in new situations. Scrapping and purchasing new equipment are normally the solution when the hardware setup is changed. In C-MAS, each process module is represented by a resource agent that has process-unique skills. Skills are supposed to be programmed by the machine builder of the module but there is seldom a need for changes in skills during the lifetime of process modules, even if the manufacturing changes. C-MAS focuses on how to handle the "produce" part of Plug & Produce as efficiently as possible from a manufacturing point of view. When a process module or robot tool is plugged in, it should be ready to produce immediately without any reprogramming or digital reconfiguration. Further, it must be possible to add new parts on the fly to the systems by only defining goals and process plans, hence, without any deeper knowledge of robot and logic programming. Indeed, this also requires a "plug-in" concept. C-MAS relies on hardware standards and automatic discovery/identification, aiming to enable digital (re-)configuration by utilizing the existing in-house knowledge.

A standard industrial robot is in this article defined as a six-axis industrial robot for generic purposes. Common brands are ABB, Kuka, Fanuc, Comau, and Yaskawa/Motoman, and will in the following be named a robot. Robots have functionalities that support the creation of local reference coordinate systems (work/user and tool frames) useful when resources change or are tilted and rotated according to the base coordinates of the robot as in the case of Plug & Produce. C-MAS is extended in this article with a generic structure for the integration of robots into agents, which utilizes the flexibility of C-MAS without losing robot functionality. This is done by storing and handling all robot-related frames on different resource agents.

2 Related work

A concept that incorporates both product design and manufacturing is Computer-Aided Process Planning (CAPP). CAPP is used in a variety of manufacturing systems but most often to plan and optimize CAD and Computer-Aided Manufacturing (CAM) for automated machining [9]. This technique is also applicable to using robots as 3D printers

for additive manufacturing [10]. The automatic generation of process plans has been realized by a contact interference analysis and feature-based analysis for automized assembly tasks [11, 12]. Optimized by the algorithm "ant colony optimization," where even the physical characteristics of the manipulator were considered, and the robot trajectory was generated [13]. Further, tolerances from CAD models and force sensors have been used to feel and verify the assembly [14]. Another approach is to link product design directly into assemblies by machine-readable answers to queries by object-oriented and feature-based data modeling, divided into five different ontologies: part, relationships, mating, joint, and handling [15, 16]. A semantic description language RoboEarth describes the kinematics and actions of mobile service robots utilizing maps of the environment defined by standardized semantics [17]. The semantics facilitates robots to understand the environment and matching requirements to find missing components and, from that, select a proper recipe for robot actions. Well-described semantics allow flexible sharing of information. This project within service robotics has successfully been adopted for industrial robots [18]. Web Ontology Language (OWL) was used to make CAD models of parts understandable for computers. Constraints between points, curves, and surfaces of an object were used to specify the assembly operations. A model of constraints for the robot and grippers was created for the generation of low-level robot operations [19]. The And/Or graph was originally introduced as a heuristic search method and is commonly used for designing sequential and parallel assembly sequences [20, 21]. An alternative to And/Or graphs is the Sequence Of Operation (SOP) charts which describe both assembly and processing operations [22]. The Sequence of goals chart that is presented in this article is inspired by the SOP chart.

Machine vision supporting industrial robots must be tightly integrated into robot controllers for efficiency. Machine vision provides robots with eyes and must consider the kinematics of the robot [23]. To give robots human-like behavior, machine vision has been extended with a system for voice recognition [24]. Another example is random bin picking where a Programmable Logic Controller (PLC) and a seamless integrated Neural Processing Unit (NPU). NPU utilizes deep learning calculations on 3D images of an RGB-D camera for generic grasping [25]. A Digital Twin (DT) created from CAD models was used as a virtual representation of the shopfloor status in a manufacturing case. Cameras and sensors were added to update the DT with runtime information [26, 27]. A unified semantic data model makes the information from DT understandable for the real-time adaption of a movable robot in pick-and-place and assembly tasks [28]. An execution coordinator, a robot arm motion planner, and a path planner together with the DT were the platform for creating real-time robot behavior.

Industrial field buses are commonly used to integrate supervision control systems into robot controllers [29]. Communication protocols such as Open Platform Communications Unified Architecture (OPC-UA) and REpresentational State Transfer (REST) give possibilities to transfer values of complex data structures, and objects [30, 31]. The data structure is organized by one of the text-based formats Extensible Markup Language (XML) or JavaScript Object Notation (JSON) [32]. JSON and REST were selected for C-MAS due to slightly lither formats and thereby faster response. Device primitives on control systems of resources such as robots gain flexibility. Device primitives are free to be composted by a skill independently of each other [1]. A similar approach implemented on top of OPC UA is described as an ontology for skills divided into three layers: task layer, composite skills layer, and atomic skills layer [33]. The atomic skills layer consists of atomics in similarity to device primitives. Manufacturer of robots has provided their controllers with an object-oriented Application Protocol Interface (API) [34]. APIs enable Machine-To-Machine (M2M) communication for the integration of controllers into other systems. The open-source Robot Operating System (ROS) on robots gives real-time access to the control loops [35, 36]. ROS can, for example, be used for teleoperation for direct control of robots [37]. Tight integration to the robot control loops makes it possible to integrate sensors to give the robot perception of its surroundings to create humanoid-like industrial robots [38]. Another approach is to give a Programmable Logic Controller (PLC) robot controller functionality. PLCopen has specified function blocks for the PLC programming language IEC61131-3 enabling control of servo drives for electrical motors to achieve robot functionalities [39–41]. This arrangement makes the robot fully integrated into the PLC, powerful together with sensors that feel the surroundings.

Distributed intelligence as a multi-agent system is beneficial for handling Plug & Produce [42]. Physical issues such as calibration and collision avoidance must be considered when robots are plugged in. A semi-automatic method for calibration and coordination of robots that share a workspace is usable for collision avoidance [43]. The system uses stereovision and markers on the end-effectors of the robots. Collisions were prevented by mutual robots and later improved with workspace allocation [44]. The cameras must be located manually in a way that markers can be detected. Cameras can be avoided by using an ontological representation of the manufacturing environment [45]. Automation Markup Language (AML) represents models of manufacturing equipment that originate from CAD [46, 47]. AML operates as a provider of work frames, locations, and functional roles. Despite all attempts to simplify the transformation of manufacturing to adopt new situations, expert competence is required for reconfiguring Plug & Produce although a

framework for decision support was developed to simplify and shorten the ramp-up time [48].

Research and development around the STandard for the Exchange of Product model data – Numeric Control (STEP-NC) was initiated at the end of the last century to increase flexibility and empower a seamless integration of activities and data from design to resource-independent CNC-based manufacturing. STEP-NC is specified in the newly updated standards ISO 14649 and ISO 10303-238 (AP238) [49]. In essence, STEP-NC focuses on what to make instead of how to make it, and resource-independent process plans describe the order of manufacturing tasks, features, and operations. A complete chain from CAD to manufacturing can thereby be fulfilled. In addition, STEP-NC supports shop floor modification feedback to the process planners which will gain cooperation among the employees and exchange of experience and insight. CNC machines together with STEP-NC have many similarities to Plug & Produce; both utilize resource-independent process plans, are reconfigurable by a set of changeable tools, and can alter between different processes. Plug & Produce is more generic and modular utilizing process modules that have individual intelligence with a strong ability to collaborate and negotiate to find solutions. Hence, Plug & Produce enables agents to automatically divide and distribute operations among available and suitable resources.

To summarize, several authors have created efficient methods for extracting data from 3D-CAD for use in automation. Ontologies have been created to facilitate automatic generations of process plans. Missing parts can be recognized by cameras in the real world by comparing images with the CAD model. All this functionality can beneficially be implemented in C-MAS but only the data extraction to a subset of earlier implemented ontology is utilized in this work to demonstrate the whole chain of data from CAD to Plug & Produce. Manual process planning is chosen to allow manufacturing companies to enhance automation processes with their knowledge and experience. Several solutions have been proposed for the integration of standard industrial robots into flexible automation systems, for example, utilizing ROS drivers to get direct access to the robot control loops where a high integration is reached, but functionalities of the standard industrial robot will be lost when ROS takes over. Methods to ease up calibration and avoid collisions in shared workspaces and methods to shorten the ramp-up time when a robot is plugged in are proposed. However, these solutions are created for a set of dedicated resources and must be redone every time a resource is plugged in; however, this is not in line with the Plug & Produce concept where resources can be moved around or altered. Much flexibility exists in a modern standard industrial robot. The challenge is to map the flexibility in a standard industrial robot and the flexibility of a multi-agent system. Nothing was found in the literature on generic integrations of agents into robot controllers where agents handle coordinates, frames, and robot paths. Hence, a generic structure for the integration of standard industrial robots into agents is proposed in this article. Further, the entire chain of data and actions to take, from CAD to Plug & Produce, by C-MAS is evaluated.

3 C-MAS structure and digital configuration

Operations in a manufacturing process are treated as goals of part agents in C-MAS. Examples of goals are placing material, joining, and making a hole. The agents in C-MAS are divided into part agents that are active and have goals, passive material agents containing related data without goals, and resource agents that have skills to refine the parts to fulfill part goals. The order of goals is important, e.g., a hole cannot be made before the material that should have the hole is placed. The sequence of goals chart is a tool to visualize and handle goals in C-MAS and has former been presented by the authors [3]. The sequential goals are visualized as a flow from top to bottom (see Fig. 1). Parallel goals are indicated by two horizontal parallel lines and contain two or more sequential behaviors that are executed in parallel. Note that parallel goals will only be executed in parallel if enough resources are available for parallel execution and concurrent operations. The behavior will become sequential if there are not enough available resources. With this desired behavior, resources can effortlessly be plugged in or out to balance the throughput of manufactured parts.

The time, from a new part design to the start of manufacturing, can be reduced if it is possible to extract useful manufacturing-related data directly from CAD, for seamless use without reentering and recalculations. The C-MAS digital configuration tool (see Fig. 2) was developed in Python scripts as an add-on to the CAD software for the extraction of data needed for Plug & Produce automation.

To extract data, a part agent for the actual part must first be selected, then a goal and the goal variable to where

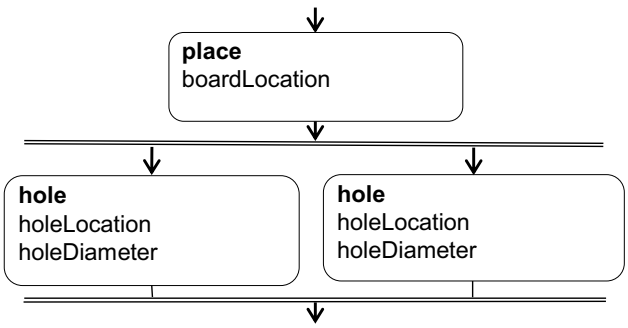

Fig. 1 The sequence of goals chart. Visualizing three goals, one place and two parallel hole goals, and goal variables

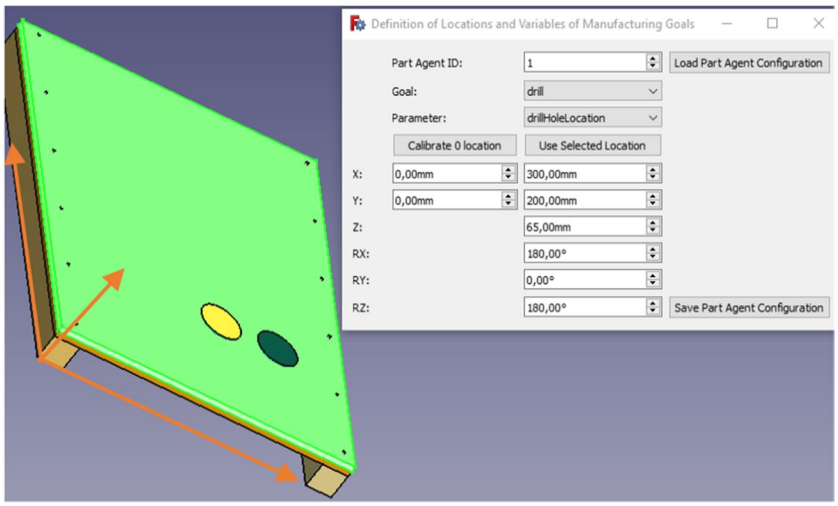

Fig. 2 3D model of a part with a calibrated reference coordinate system and a dialog box for extracting data into goal variables of part agents

the selected data should be assigned, and then the wanted location on the 3D model of the part. All variables are stored in a J-SON file that is structured according to the C-MAS ontology in Fig. 3. An example of the J-SON structure for the variable holeLocation is given in Fig. 4. Locations are related to the reference coordinate system of the part model which is possible to calibrate in the dialog box (Fig. 2). The add-on Python script reads goals, variables, and assigned values of part agents from the C-MAS J-SON structure and writes assignments of values to goal

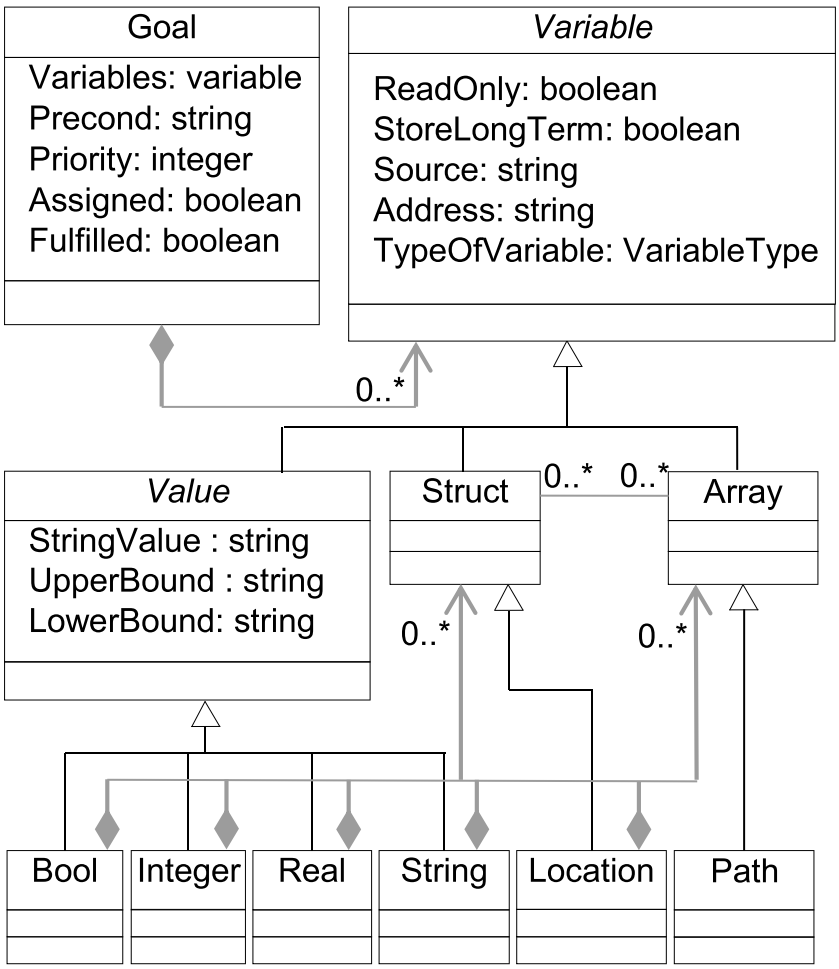

Fig. 3 C-MAS ontology of a part agent goal and goal variables

```
"Parameters": [ {
        "Address": "",
        "Description": "Location of hole 1",
        "Error": "FALSE",
        "ID": "126",
        "InstanceID": "212.25.132.102-163….",
        "LowerBound": "",
        "Name": "holeLocation",
        "ReadOnly": "FALSE",
        "Relative": null,
        "Source": "",
        "StoreLongTerm": "FALSE",
        "StringValue": "",
        "Type": "VARIABLE",
        "TypeOfLocation": "NONE",
        "TypeOfVariable": "LOCATION",
        "UpperBound": "",
        "rx": 180.0,
        "ry": 0.0,
        "rz": 180.0,
        "x": 300.6,
        "y": 200.0,
        "z": 65.0
    } ],
```


Fig. 4 An example of the extracted hole location in J-SON format according to the C-MAS ontology

variables that are pointed out on the CAD model of the part.

The C-MAS ontology in Fig. 3 complements the formerly presented ontology [3], the name of class variable types begins with an uppercase letter, and primitive variable types are named with lowercase letters. Variables, structs, and arrays are free to use for general purposes. A robot Location variable consists of a structure of Real containing coordinates (x, y, z) and rotations (Rx, Ry, Rz). A Path consists of an array of Locations to create a complete robot trajectory.

A typical automation resource that is hard to handle in a Plug & Produce setup is a robot. Utilizing device primitives implemented on control systems of resources gains flexibility by moving decisions from the local robot controller to a flexible control system such as C-MAS [1]. Each device primitive represents a discrete function of the resource, enabling the possibility of having a fixed standardized implementation containing all device primitives of the resource that must not be reprogrammed even if the conditions change. Device primitives must be implemented in the language of the actual brand of controller, but are brand-independent, useful for any kind of controller, and reachable from any kind of flexible control system not only multi-agent systems.

The four main abstraction layers of the C-MAS structure are shown on the left side of Fig. 5. The top layer represents a set of part agents. Part agents maintain the goals, sequence of goals, and goal variables, and have a weak relation to process plans. Process plans are found by matching names, the process plans must have the same name as the goal. More than one process plan is possible but only one process plan with the lowest cost will be selected to solve the goal. A process plan contains sequences of skills to be performed by any available and suitable resource agents to fulfill a goal. The relation between process plans and resource agents is based on abstract interfaces to enable a loosely coupled relation. Resource agents can run skills on their own or with the help of skills from other agents after searching and negotiating

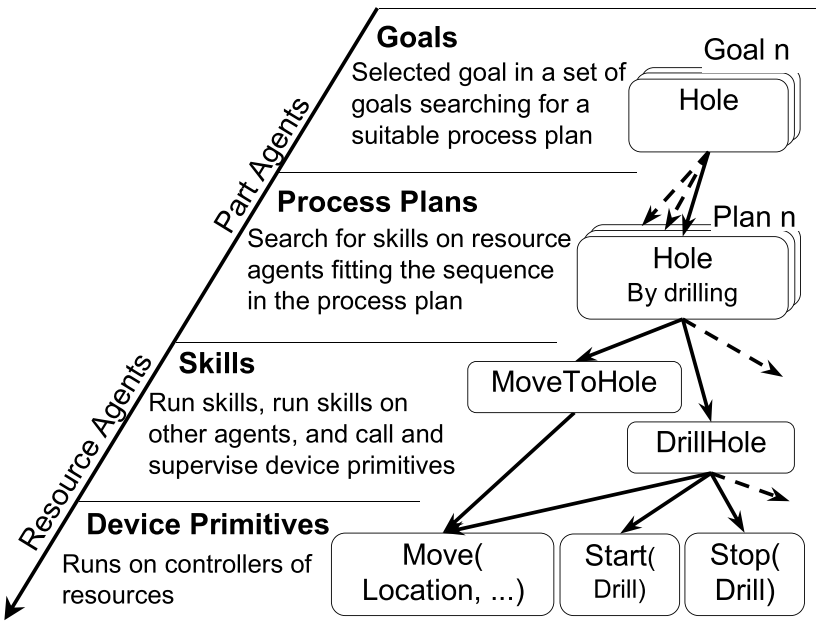

Fig. 5 Execution abstraction layers in C-MAS. To the right is an example of the goal of making a hole

among the resource agents. A resource agent utilizes the same mechanism, abstract interfaces, to maintain flexibility and loosely coupled relations within the system. A machine or robot that embeds a control system should have defined device primitives that represent atomic actions implemented as functions. A resource agent is then free to utilize and mix device primitives to describe and present more complex functionality as skills. This structure enables loosely coupled layers with a high degree of flexibility. Further, even if the abstraction model is hierarchical, the independent heterarchical multi-agent control structure is preserved.

The right side of Fig. 5 depicts a simplified example of a digital configuration for one of the goals on one of the part agents, the goal to make a hole. Several process plans may exist, a hole can for example be made by drilling or machining, and the process plan that generates the lowest cost will be selected, a drilling plan is used in this example. The skills MoveToHole and DrillHole utilized the device primitives Move(Location), Start(Drill), and Stop(Drill) to move the robot and drill the hole. Extracting data from CAD and defining process planes and sequence of goals have all a process focus and can daily be reconfigured by the manufacturing company in any order or concurrently (see Fig. 6). Note that programming is still required when it comes to defining device primitives and skills, but they are relatively static even if the parts or Plug & Produce setup change. If generic skills and device primitives can be created, they are more likely to last longer. Skills can be threatened as reusable drive routines that are implemented on resource agents, ready to serve the system when they are plugged in. The calibration is performed by functionalities that exist on robots and are stored as data objects in the robot controller. In the case of C-MAS, the data objects are for flexibility reasons uploaded and stored on the related resource agent and downloaded again when needed, handled in this work by the implemented generic structure of agent integration to robots.

4 A generic structure for the integration of robots into agents

A standard industrial robot has six motorized joints for motions in six degrees of freedom divided into three cartesian directions x, y, z, and three rotations r_x, r_y, r_z. A generic description of the kinematic chains of joints from the base frame to the tool frame is given in Fig. 7. The six joints are noted Θ_1 to Θ_6. The base frame coordinates noted x_0, y_0, z_0 have their origin in the robot foot. The origin of the coordinates x_6, y_6, z_6 is located at the point where the robot tool is fitted, normally named flange center point or tool0.

The Tool Center Point (TCP) is from the start located in the flange center point but is possible to move by defining a tool object according to the actual tool in use. The TCP will then reflect the offset the tool makes from the flange center. A work frame or user frame is a reference used for locations in the actual work area. Work frames are possible to define by defining work objects related to the base frame. A full description of the industrial robot coordinates and translations can be found in Shah et al. [50]. The location and rotation of TCP are continuously

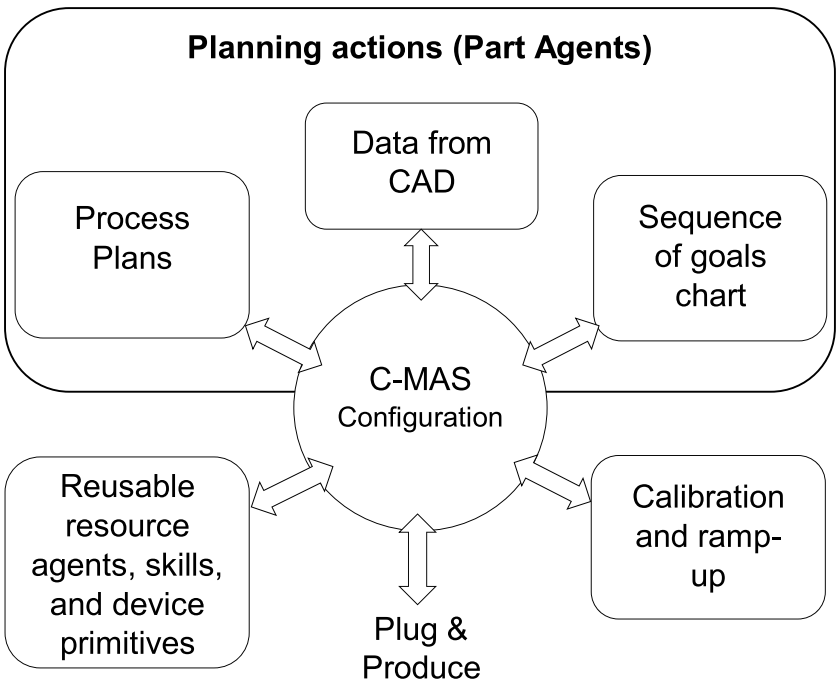

Fig. 6 Parallel and concurrently digital configuration actions of C-MAS

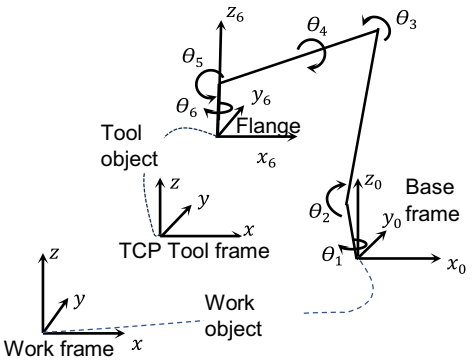

Fig. 7 Kinematic chain of six-axis standard industrial robot and reference coordinate systems

calculated by the robot controller using the kinematic chain from the base, related to the base or a work frame. This structure for defining robot positions makes the robot more flexible. It becomes easier to change applications or minor adjustments to target locations. Hence, frames are possible to update if the tool or work area changes without affecting the robot program. Figure 8 shows a robot in a Plug & Produce implementation and related frames. The functionality of standard industrial robots is customized by a program, implemented according to the structure in Fig. 9. Tool objects, work objects, and target locations are all stored as data objects in the robot controller and are utilized in the robot program containing a fixed set of instructions. However, the conventional solution where the robot owns the tool/work objects and locations of other resources will create a tight coupling between the robot and these resources by a composition structure. Hence, a standard industrial robot is hard to integrate into the flexible concept of Plug & Produce without losing the aim of the concept.

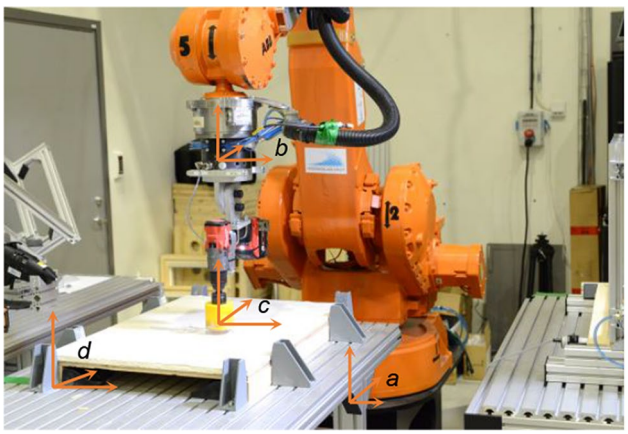

Fig. 8 Six-axis robot (ABB 4400), Plug & Produce, and related frames. **a** Base frame. **b** Flange center point. **c** Tool frame. **d** Work frame

Declarations
Set of tool objects
Set of work objects
Set of locations
Set of variables

Sequence of instructions
Move(WorkObject, ToolObject, Location, Speed, Zone)
Move(WorkObject, ToolObject, Location, Speed, Zone)
SetOutPut(Variable)
Move(WorkObject, ToolObject, Location, Speed, Zone)
ResetOutPut(Variable)
….

Fig. 9 Structure of a standard industrial robot program where all objects are owned by the robot (composition)

A better approach, which is in line with Plug & Produce, is to make each resource the owner of its data objects (see Fig. 10). Hence, the behavior of individual robots can be based on C-MAS digital configuration to avoid costly and time-demanding robot programming. Such a generic structure is proposed in this article and the link to CAD design. The structure implements a loosely coupled relationship between the robot and surrounding resources and parts. The work/tool objects and locations are created in the robot controller and uploaded to resource agents by in this work developed HMI of each agent, configurable in skills. The data is uploaded as a data object in the JSON format from the robot via a communication protocol through an API of the robot controller. The data is stored untouched in string variables of the agents (see Fig. 3). The agent HMI is configured in skills and visualized on an operator panel.

Figure 11 shows agents, variables, and interfaces that are engaged in the runtime execution of the specific example of a goal Hole. The PartAgent starts searching for process plans by matching names and selects the one with the lowest cost.

Part Agents
Set of goals
Set of goal variables (Locations, ..)
Sequence of goals
Set of process plans

Resource Agents
Work obect
Set of Locations
Set of skills

Tool Agents
Tool obect
Set of skills

Robot Agents
Set of skills utilizing
device primitives

Robot program of Device Primitives
Move(WorkObject, ToolObject, Location, Speed, Acc, Zone)
Start(Output)
Stop(Output)

Fig. 10 Generic structure of C-MAS objects distributed on agents and a standard robot program of device primitives

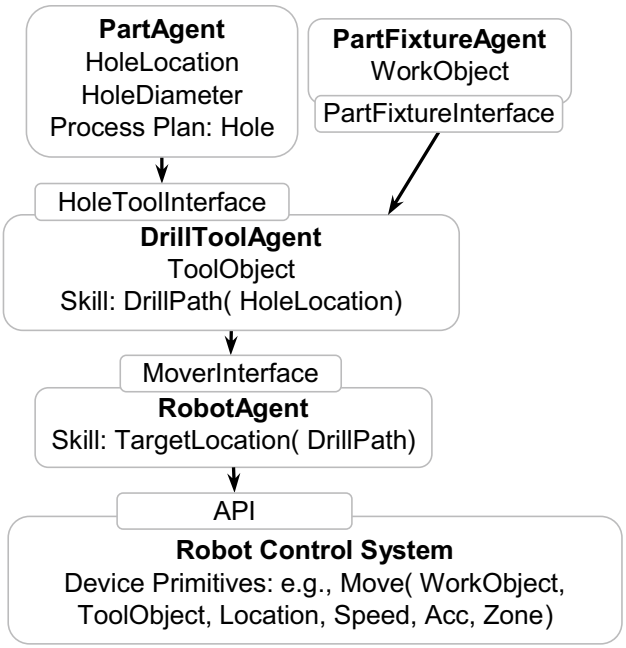

Fig. 11 Example of C-MAS agent structure executing the goal to make a hole

The execution of the process plan Hole starts by searching for a suitable tool agent by using the HoleToleInterface that can make the hole in the specified location and diameter; the DrillToolAgent was found. The DrillToolAgent knows its tool object, and it was uploaded from the robot via the HMI of the DrillTooAgent when the tool was set up for the first time. The DrillToolAgent finds the PartFixtureAgent through PartFixtureInterface, which has the actual work object for the part fixture. The DrillPath skill of the DrillToolAgent creates a path to the location stated in the goal variable HoleLocation which originates from CAD. The DrillToolAgent starts searching for an agent that can move the drill tool, and the RobotAgent was found through the MoverInterface. The robot skill TargetLoacation assigns values to the variables of the device primitive Move. Move receives the actual work object, tool object, target location, wanted speed, acceleration, and the zone around the actual target location where the robot can alter to a new target. When the robot has started to move, it is ready to receive the next target location. The robot agent will then send the next target location and repeat until the full path is finalized.

5 Test cases, verification, and results

A Plug & Produce demonstrator for automatically manufactured wall sections of wooden houses was built to verify the functionality of the proposed method for the integration of CAD and robots into C-MAS [3]. A process module with a miniaturized wall section is visible in the front of Fig. 8. A question to answer through this work is: which actions are needed to be able to extract data from 3D-CAD models of the part for seamless use in the Plug & Produce without reentering and transformation of data? Three actions were identified: (Action A) Efforts must be made to calibrate the robot's work frame to match the coordinates of the part's 3D-CAD model. If the work frame is related to a fixed location marked on the process modules, the work frame can be reused for many different types of parts, which makes the calibration a one-time work when a process module is being built. The accuracy between the location extracted from CAD and the achieved robot TCP depends on how accurately the calibration is performed and the accuracy of the robot and fittings. (Action B) The 3D-CAD model of a part must be calibrated according to the work object created in action A (see the calibration button in Fig. 2). This work must be done the first time a new part is introduced to the system. (Action C) Configuration of process plans, sequence of goals, and extracting process-related data from the 3D-CAD model of the part. All planning actions for a part agent are visible in Fig. 6. From the 3D-CAD model, data such as required processes and process parameters can be identified and extracted. When translated into C-MAS, each process/operation is considered a goal. Related process parameters are extracted to goal variables and the order of the goals is configured in the Sequence of goals chart (Fig. 1). The time it takes to modify a part, from CAD to Plug & Produce, was verified through the industrial reference group connected to this work, described in [3]. Changes such as adding a new goal or moving a location can be done in less than 3.5 min by utilizing the in-house knowledge of a manufacturing company that has CAD experience and a process focus. The digital configuration for a new part can be completed during ongoing manufacturing. The configuration will automatically be used when the new part is introduced to the Plug & Produce system.

The generic structure for the integration of robots into agents performed well in the described Plug & Produce demonstrator for wall sections of wooden houses. Work/Tool objects and locations were calibrated and uploaded to the different agents where they belong, included in action A. During the execution, agents utilized the data and automatically downloaded it to the robot when needed in different situations. No deviations or delays could be observed due to incorrect or late responses from C-MAS and the robot controller in this quite common industrial setup. Performance and behavior were comparable to a standalone and conventional programmed robot.

A second test case was set up to test the limits and verify how fast the robot can move and still take on-the-fly decisions without any delays or affections of the trajectory by letting the robot write a triangle on a piece

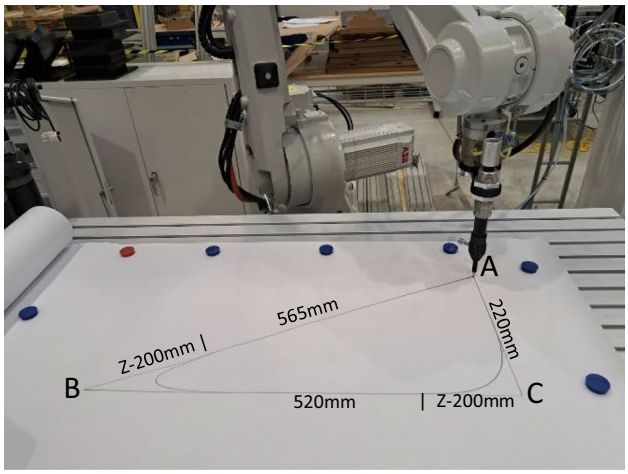

Fig. 12 ABB IRB2600 robot holding a pencil that draws the trajectories on a piece of paper. Note that the zone for B and C of 200mm makes a smooth transition

of paper at a variety of speeds. The triangle was drawn from the start location A via B and C, and then back to A again (see Fig. 12). The target zone was set to 200mm, marked as Z-200mm. A zone is a robot-related parameter to specify how close to the programmed location the TCP must be before it starts to move to the next location and is visible in the figure as a smooth transition between the two lines in A and B. The standalone robot program was simple and contained just three Move instructions implemented according to the structure in Fig. 9. C-MAS was configured with three goals to move to the three locations and the device primitives were implemented on the robot controller according to the structure in Fig. 10. The test was carried out on an ABB IRB2600 robot with an IRC5 controller running robotware 6.14. C-MAS was executed on a laptop computer with a 2.60GHz 8-core i9 processor. The agents and robot communicate through J-SON formatted data over a REST web service protocol on a 100Mbps Ethernet network. No differences in the robot behavior running standalone or utilizing the C-MAS structure of robot agent integration could be observed for TCP speeds up to 1m/s. The motion was delayed by 70ms for TCP speeds above 1m/s in the case of C-MAS (see Table 1). The reason for these latencies could be traced to the agent robot communication, e.g., the response time of the robot was up to 10ms during fast motion due to the internal behavior of the robot. No deviations of the robot tracery were observed at any speed; the robot was able to do smooth transitions to new target locations within the zones even at the maximum speed, in this test case 4m/s.

Table 1 Robot TCP speed and time to perform the triangle by standalone robot and C-MAS structure of robot agent integration

Robot TCP Speed (*m/s*)	Time (s) Standalone	Time (s) C-MAS
0.6	2.00	2.01
0.8	1.58	1.58
1	1.33	1.33
1.5	1.07	1.13
2	0.99	1.06
4 (max)	0.97	1.04

6 Discussion and conclusion

Efforts and time needed to adapt C-MAS to new Plug & Produce scenarios have earlier been tested and evaluated compared to other flexible systems with good results [3]. C-MAS is extended in this work with a graphical method for extracting data directly from 3D-CAD for seamless use in Plug & Produce without any manual reentering and transforming enabled by this article's proposed generic structure for the integration of standard industrial robots into agents. This approach, which has the manufacturing processes in focus, should be compared to conventional industrial approaches where the process data are manually translated to program code and scaled to entities that fit the functionalities of the resources. All digital configuring actions and automatic data transfers from CAD to Plug & Produce are now considered by C-MAS. Actions such as extracting data from CAD and configuring process plans, goals, and sequence of goals can be performed independently in any order or concurrently. New or changed parts can be configured during runtime for use the next time the part will be manufactured, and a new part agent is deployed. Skills of resource agents and device primitives of controllers are programmed one time when the resources are built. Independently of the actual Plug & Produce systems they will be plugged into, an approach that gains resource flexibility and reusability of manufacturing equipment. C-MAS is based on manually created process plans to fulfill the requirement of the industry to have full control of the manufacturing process and will allow process engineers to implement their knowledge into the Plug & Produce automated processes. Several authors propose an automatic generation of process plans based on 3D-CAD models enabled by a well-structured and evolved ontology. The ontology of C-MAS is feasible to extend equally to support auto-generated process plans.

All standard industrial robots utilize data objects for work frames, tool frames, and locations, but the format differs among the brands. The proposed method of integration of robots into agents is generic since the agents do not translate or interpret specific features of the data. The agents are instead aware of the structure of the data and the relation in-between data objects.

The agents are then able to provide the correct data when needed by a robot. A lack of a standardized API of controllers implies that the communication is brand-dependent. A REST web service protocol of an ABB IRC6 controller was utilized in this work and a rest client was implemented in C-MAS. Support of several communication protocols and APIs is needed to maintain generalized communication. The integration of standard industrial robots to agents is generic for any flexible systems that have resource agents or equivalent. C-MAS is aimed at machine-close operations where fast response is important. The test of on-the-fly altering of robot trajectory showed no delay in reasonable speeds for industrial manufacturing robots, at higher speeds was a minor delay observed due to delays in the communication. The efficiency of a standalone programmed robot is then slightly better in some cases but the proposed C-MAS structure admits flexibility that is built in from scratch, possible to handle daily by the in-house knowledge of manufacturing companies.

Author contributions All authors contributed to the conception study and design. Material preparation, data collection, and analysis were performed by Anders Nilsson. The first draft of the manuscript was written by Anders Nilsson and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding Open access funding provided by University West. This work was supported by Region Västra Götaland (VGR) Dnr. RUN 2018-00476 and Swedish Agency for Economic and Regional Growth ID: 20201948 (Tillverka i trä).

Declarations

Competing interests The authors declare no competing interests.

References

1. Pedersen MR, Nalpantidis L, Andersen RS, Schou C, Bøgh S, Krüger V, Madsen O (2016) Robot skills for manufacturing: From concept to industrial deployment. Robot Comput-Integrated Manuf 37:282–291. https://doi.org/10.1016/j.rcim.2015.04.002
2. Nilsson A, Danielsson F, Mattias B, Bo S (2021) A classification of different levels of flexibility in an automated manufacturing system and needed competence. In: Towards Sustainable Customization: Bridging Smart Products and Manufacturing Systems. Springer, Cham Aalborg, pp 27–34. https://doi.org/10.1007/978-3-030-90700-6_2
3. Nilsson A, Danielsson F, Svensson B (2023) Customization and flexible manufacturing capacity using a graphical method applied on a configurable multi-agent system. Robot Comput-Integr Manuf:79. https://doi.org/10.1016/j.rcim.2022.102450
4. Monostori L, Váncza J, Kumara SRT (2006) Agent-based systems for manufacturing. CIRP Ann Manuf Technol 55:697–720. https://doi.org/10.1016/j.cirp.2006.10.004
5. Perzylo A, Rickert M, Kahl B, Somani N, Lehmann C, Kuss A, Profanter S, Beck AB, Haage M, Hansen MR, Nibe MT, Roa MA, Sornmo O, Robertz SG, Thomas U, Veiga G, Topp EA, Kesslar I, Danzer M (2019) SMErobotics: Smart robots for flexible manufacturing. IEEE Robot Autom Mag 26:78–90. https://doi.org/10.1109/MRA.2018.2879747
6. Arai T, Aiyama Y, Maeda Y, Sugi M, Ota J (2000) Agile Assembly System by "Plug and Produce.". CIRP Ann Manuf Technol 49:1–4. https://doi.org/10.1016/S0007-8506(07)62883-2
7. Ribeiro da Silva E, Schou C, Hjorth S, Tryggvason F, Sørensen MS (2022) Plug & Produce robot assistants as shared resources: A simulation approach. J Manuf Syst 63:107–117. https://doi.org/10.1016/j.jmsy.2022.03.004
8. Bennulf M, Danielsson F, Svensson B (2019) Identification of resources and parts in a Plug and Produce system using OPC UA. In: Procedia Manufacturing. Elsevier B.V., pp 858–865. https://doi.org/10.1016/j.promfg.2020.01.167
9. Zhou J, Camba JD (2021) Computer-aided process planning in immersive environments: A critical review. Comput Ind 133:103547. https://doi.org/10.1016/j.compind.2021.103547
10. Zheng H, Cong M, Dong H, Liu Y, Liu D (2017) CAD-based automatic path generation and optimization for laser cladding robot in additive manufacturing. Int J Adv Manuf Technol 92:3605–3614. https://doi.org/10.1007/s00170-017-0384-0
11. Soori M, Asmael M (2021) Classification of research and applications of the computer-aided process planning in manufacturing systems. Ind J Manag Prod 12:1250–1281. https://doi.org/10.14807/ijmp.v12i5.1397
12. Hasan B, Wikander J (2017) Features extraction from CAD as a basis for assembly process planning. In: IFIP Advances in Information and Communication Technology. Springer, New York LLC, pp 144–153. https://doi.org/10.1007/978-3-319-56077-9_13
13. Ying KC, Pourhejazy P, Cheng CY, Wang CH (2021) Cyber-physical assembly system-based optimization for robotic assembly sequence planning. J Manuf Syst 58:452–466. https://doi.org/10.1016/j.jmsy.2021.01.004
14. Pane Y, Arbo MH, Aertbelien E, Decre W (2020) A System Architecture for CAD-Based Robotic Assembly with Sensor-Based Skills. IEEE Trans Autom Sci Eng 17:1237–1249. https://doi.org/10.1109/TASE.2020.2980628
15. Khabbazi MR, Wikander J, Bergsteth E, Maffei A, Onori M (2017) Assembly Feature Data Instance Modeling: Prototype Implementation and Outputs. In: 2017 International Conference on Mechanical. System and Control Engineering (ICMSC), IEEE, pp 343–347. https://doi.org/10.1109/ICMSC.2017.7959498
16. Khabbazi MR, Wikander J, Onori M, Maffei A (2018) Object-oriented design of product assembly feature data requirements in advanced assembly planning. Assembly Autom 38:97–112. https://doi.org/10.1108/AA-07-2016-084
17. Tenorth M, Perzylo AC, Lafrenz R, Beetz M (2013) Representation and exchange of knowledge about actions, objects, and environments in the ROBOEARTH framework. IEEE Trans Autom Sci Eng 10:643–651. https://doi.org/10.1109/TASE.2013.2244883
18. Perzylo A, Somani N, Rickert M, Knoll A (2015) An ontology for CAD data and geometric constraints as a link between product models and semantic robot task descriptions. In: IEEE International Conference on

Intelligent Robots and Systems. Institute of Electrical and Electronics Engineers Inc., pp 4197–4203. https://doi.org/10.1109/IROS.2015.7353971
19. Somani N, Gaschler A, Rickert M, Perzylo A, Knoll A (2015) Constraint-based task programming with CAD semantics: From intuitive specification to real-time control. In: IEEE International Conference on Intelligent Robots and Systems. Institute of Electrical and Electronics Engineers Inc., pp 2854–2859. https://doi.org/10.1109/IROS.2015.7353770
20. Mahanti A, Bagchi A (1985) AND/OR Graph Heuristic Search Methods. J Assoc Comput Machinery (JACM) 32:28–51. https://doi.org/10.1145/2455.2459
21. Perzylo A, Somani N, Profanter S, Kessler I, Rickert M, Knoll A (2016) Intuitive instruction of industrial robots: Semantic process descriptions for small lot production. In: IEEE International Conference on Intelligent Robots and Systems. Institute of Electrical and Electronics Engineers Inc., pp 2293–2300. https://doi.org/10.1109/IROS.2016.7759358
22. Lennartson B, Bengtsson K, Yuan C, Andersson K, Fabian M, Falkman P, Akesson K (2010) Sequence planning for integrated product, process and automation design. IEEE Trans Autom Sci Eng 7:791–802. https://doi.org/10.1109/TASE.2010.2051664
23. Cheng F, Chen X (2008) Integration of 3D Stereo Vision Measurements in Industrial Robot Applications. In: Proceedings of The 2008 International Association of Journals & Conferences – International Journal of Modern Engineering. IAJC, International Conference, pp 17–19
24. Tasevski J, Nikolic M, Miskovic D (2013) Integration of an industrial robot with the systems for image and voice recognition. Serb J Electr Eng 10:219–230. https://doi.org/10.2298/sjee1301219t
25. Solowjow E, Ugalde I, Shahapurkar Y, Aparicio J, Mahler J, Satish V, Goldberg K, Claussen H (2020) Industrial Robot Grasping with Deep Learning using a Programmable Logic Controller (PLC). In: 2020 16th IEEE International Conference on Automation Science and Engineering (CASE) August 20-21, 2020. Online Zoom Meeting, pp 97–103. https://doi.org/10.1109/CASE48305.2020.9216902
26. Fang X, Wang H, Liu G, Tian X, Ding G, Zhang H (2022) Industry application of digital twin: from concept to implementation. Int J Adv Manuf Technol 121:4289–4312. https://doi.org/10.1007/s00170-022-09632-z
27. Gkournelos C, Kousi N, Christos Bavelos A, Aivaliotis S, Giannoulis C, Michalos G, Makris S (2019) Model based reconfiguration of flexible production systems. Procedia CIRP 86:80–85. https://doi.org/10.1016/j.procir.2020.01.042
28. Kousi N, Gkournelos C, Aivaliotis S, Giannoulis C, Michalos G, Makris S (2019) Digital twin for adaptation of robots' behavior in flexible robotic assembly lines. Proc Manuf 28:121–126. https://doi.org/10.1016/j.promfg.2018.12.020
29. Cavalieri S, di Stefano A, Mirabella O (1997) Impact of Fieldbus on Communication in Robotic Systems. IEEE Trans Robot Autom 13:30–48. https://doi.org/10.1109/70.554345
30. Ferreira LA, Souto MÁ, Chappuis C, el Khaldi F (2020) Off-line programming of a flexible and adaptive production line for composite-metal multi-material manufacturing based on OPC-UA communication. Proc Manuf 51:520–526. https://doi.org/10.1016/j.promfg.2020.10.073
31. Liu XF, Shahriar MR, Al Sunny SN, Leu MC, Hu L (2017) Cyber-physical manufacturing cloud: Architecture, virtualization, communication, and testbed. J Manuf Syst 43:352–364. https://doi.org/10.1016/j.jmsy.2017.04.004
32. Nurseitov N, Paulson M, Reynolds R, Izurieta C (2009) Comparison of JSON and XML data interchange formats: a case study. Caine 9:157–162
33. Brovkina D, Riedel O (2021) Assembly Process Model for Automated Assembly Line Design. In: 2021 IEEE 3rd Eurasia Conference on IOT. Communication and Engineering (ECICE), pp 588–594. https://doi.org/10.1109/ECICE52819.2021.9645604
34. Angerer A, Hoffmann A, Schierl A, Vistein M, Reif W (2013) Robotics API: Object-oriented software development for industrial robots. J Software Eng Robot 4:1–22
35. Koubâa A (2017) Robot Operating System (ROS). Springer (Switzerland) 1:112–156
36. Andersen T (2015) Optimizing the Universal Robots ROS driver. Technical University of Denmark, Department of Electrical Engineering
37. Baklouti S, Gallot G, Viaud J, Subrin K (2021) On the improvement of ros-based control for teleoperated Yaskawa robots. Appl Sci (Switzerland) 11. https://doi.org/10.3390/app11167190
38. Martinez C, Barrero N, Hernandez W, Montaño C, Mondragón I (2017) Setup of the Yaskawa SDA10F robot for industrial applications, using ROS-Industrial. In: Advances in Automation and Robotics Research in Latin America. Springer International Publishing, Cham, pp 186–203. https://doi.org/10.1007/978-3-319-54377-2_16
39. Dai F, Becker O (2013) A PLCopen-Based Approach for Utilizing Powerful Industrial Robot Functions in PLC-Controlled Applications. In: Advances in Sustainable and Competitive Manufacturing Systems. Springer, pp 547–557. https://doi.org/10.1007/978-3-319-00557-7_45
40. International Electrotechnical CommissionI (2013) IEC 61131–3:2013 Programmable Controllers, Part 3: Programming languages. International Standard TC 65/SC 65B. International Electrotechnical Commission, Geneva, CH. https://webstore.iec.ch/publication/4552
41. Jeong HS, Ji SH, Jung HS, Koo JC (2017) Design of SW Architecture for PLC Integrated Robot. In: 2017 14th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI). IEEE Future Networks World Forum, pp 874–876. https://doi.org/10.1109/URAI.2017.7992851
42. Leitão P, Barbosa J, Pereira A, Barata J, Colombo AW (2016) Specification of the PERFoRM architecture for the seamless production system reconfiguration. In: IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society, pp 5729–5734. https://doi.org/10.1109/IECON.2016.7793007
43. Arai T, Maeda Y, Kikuchi H, Sugi M (2002) Automated Calibration of Robot Coordinates for Reconfigurable Assembly Systems. CIRP Annals 51:5–8. https://doi.org/10.1016/S0007-8506(07)61454-1
44. Maeda Y, Kikuchi H, Izawa H, Ogawa H, Sugi M, Arai T (2007) "Plug & Produce" functions for an easily reconfigurable robotic assembly cell. Assembly Autom 27:253–260. https://doi.org/10.1108/01445150710763286
45. Makris S, Alexopoulos K, Michalos G, Sardelis A (2020) An Agent-Based System for Automated Configuration and Coordination of Robotic Operations in Real Time—A Case Study on a Car Floor Welding Process. J Manuf Mater Process 4. https://doi.org/10.3390/jmmp4030095
46. Drath R (2021) AutomationML: a practical guide. Walter de Gruyter GmbH & Co KG. https://doi.org/10.1515/9783110746235
47. Wojtynek M, Steil JJ, Wrede S (2019) Plug, Plan and Produce as Enabler for Easy Workcell Setup and Collaborative Robot Programming in Smart Factories. KI - Kunstliche Intelligenz 33:151–161. https://doi.org/10.1007/s13218-019-00595-0
48. Zimmer M, Ferreira P, Danny P, Al-Yacoub A, Lohse N, Gentile V (2019) Towards a Decision-support Framework for Reducing Ramp-up Effort in Plug-and-Produce Systems. In: 2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS). 2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS), pp 478–483. https://doi.org/10.1109/ICPHYS.2019.8780369
49. Srivastava D, Komma VR (2023) STEP-NC AP238-an excellent paradigm for smart manufacturing. Int J Interact Design Manuf (IJIDeM):1–13. https://doi.org/10.1007/s12008-023-01289-6
50. Shah BC, Nagal DD, Sharma DS (2016) Coordinate Systems for Industrial Robots. Int J Technol Res Eng 2347(4718):191–193