Available online at www.sciencedirect.com

ScienceDirect

Procedia Manufacturing 51 (2020) 416–423

Procedia MANUFACTURING

www.elsevier.com/locate/procedia

30th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM2021)
15-18 June 2021, Athens, Greece.

A CAD feature-based manufacturing approach with OPC UA skills

Magnus Volkmann[a,*], Tatjana Legler[a,b], Andreas Wagner[a], Martin Ruskowski[a,b]

[a] *Chair of Machine Tools and Control Systems (WSKL), TU Kaiserslautern, Gottlieb-Daimler-str. 42, 67663 Kaiserslautern, Germany*
[b] *Innovative Factory Systems (IFS), German Research Center for Artificial Intelligence (DFKI), Trippstadter Str. 122, 67663 Kaiserslautern, Germany*

Abstract

One of the biggest obstacles on the path to digitalization in an industrial environment is the lack of communication standards. This is particularly evident where several domains meet. In this case, conventional machine tools, robotics and computer vision have to interact over a common interface. We propose a structure based on OPC UA, which unifies the procedure of programming for the human operator, regardless of whether the task is performed by a robot or a machine tool in the end. The idea is to use OPC UA skills for the manufacturing of components, that consist of common standard geometric features such as holes or pockets. Our approach enables the translation of required manufacturing parameters directly from a CAD model to OPC UA skills instead of using CAM. Therefore, it is necessary to investigate suitable CAD interfaces as well as enabling an autonomous implementation of the process on a manufacturing system. From this approach, the definition of an OPC UA Companion Specification for manufacturing processes could be derived. The implementation of the concept is carried out on a flexible robot cell.

Keywords: OPC UA skills; skill-based engineering; feature-based manufacturing; production level 4; Industrie 4.0

1. Introduction

Through the development of Industrie 4.0, new standards such as Open Platform Communications Unified Architecture (OPC UA) [24] enable the standardization of machine communication. More and more production technologies of the same type are being described in a uniform semantic way by OPC UA Companion Specifications. For example the universal machine tool interface project (UMATI) [35] currently brings together several machine tool manufacturers who are developing a common specification to share data. However, the current specifications are mainly limited to variables that can be read from the respective controller. Due to the demand for more flexible production, it will not only be important to read data from the control systems, but also to start and coordinate processes. The traditional hierarchical architecture to control systems is not able to meet these requirements. For this purpose, a flexible manufacturing approach based on OPC UA is demonstrated. In our approach tasks can be implemented in dependence of a few variables in a reliable way. The aim is to show the possibility of implementing certain manufacturing methods with only a few necessary parameters. In previous literature these methods were referred to as skills [1, 4, 7, 17, 25, 34]. The existing papers are mainly focused on motion sequences or assembly processes. This paper will show the possibility to carry out manufacturing processes autonomously on a machine control with the help of the information saved in computer-aided design (CAD) models. Up to now, manufacturing requires modelling the geometry in CAD and a simulation in a computer-aided manufacturing (CAM) environment. A post-processor then generates the corresponding machine code. As shown in Fig. 1, this approach is intended to demonstrate the possibility of manufacturing products directly from CAD via a standardized interface.

A KUKA KR 300 was chosen as a testbed for this purpose. The required manufacturing parameters were transferred from Fusion 360 CAD software over OPC UA to the robot. Therefore, skills on the robot controller and a Python CAD plug-in had to be programmed.

2. Related Work

The release of Companion Specifications, for example the Robotics Part 1 [21], as well as the work within the UMATI

* Corresponding author. Tel.: +49-631-205-5067 ; fax: +49-631-205-3705.
E-mail address: magnus.volkmann@mv.uni-kl.de (Magnus Volkmann).

2351-9789
10.1016/j.promfg.2020.10.059

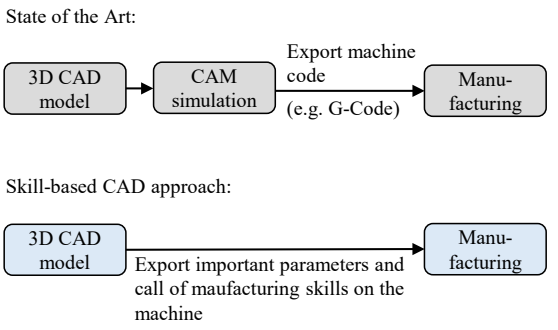

Fig. 1. Comparsion of the state of the art and the skill- and CAD-based approach

project [35] are a first step towards a standardized way of communication between production machines. Up to now, standardization has enabled a simplified vertical infrastructure which makes it possible to clearly identify a machine, specify it and read approved data from it. For the future, however, it will be interesting to see how production tasks can be distributed across several machines using component information. In [16] a skill-based approach is already described, whereby all capabilities of a production system can be mapped and a corresponding path to workpiece production can be found. A skill-based manufacturing framework which is able to adapt automatically to changing requirements in the production is implemented in [29]. The implementation and use of skills on the machine level has been investigated in several papers. The use of skills is crucial to reuse code for applications and prevents reprogramming of specific tasks [15]. In [31] it is shown that OPC UA is one of the best performing open source protocols over TCP, UDP at machine level and has its strength in semantic information modelling. Another advantage is the clearly defined semantic description, which is defined in Companion Specifications of the OPC Foundation. In [11, 18, 30] possibilities to describe an OPC UA skill semantically were already shown. By using simple movement skills, it was demonstrated in [30] that the concept of skill-based programming works. Our work relies heavily on the presented concepts for the development of OPC UA skills but so far mainly flexible skills for movement or handling (e. g. pick and place) have been investigated. For example in [26] pick and place and calibration skills are implemented on a collaborative robot mounted on a mobile platform. New tasks on the complex system can be programmed by non experts because the skills are very intuitive. In [2] a pick skill in combination with a depth sensor is developed. The teaching of a robot cell by mounting parts on an intuitive teaching interface is shown in [27]. In the evaluation of [27] the intuitive programming in combination with robot skills for mounting a gear or assembling a wood frame can save around 70-80% of the programming time. In [32] a plug and produce framework for collaborative robots is shown where the programming is based on skills. To implement the concept described in [16], where features of components are mapped on an production environment, skills which support the machining of components are required.

3. OPC UA manufacturing skills

OPC UA manufacturing skills are predefined manufacturing methods for a special task which are shared on an OPC UA server for example a programmable logic controller (PLC). We decided to use OPC UA as communication architecture for the skill-based manufacturing approach. The most important advantage from our point of view was the possibility to define unique semantics [22, 31] to be able to apply and orchestrate the skills on other machine types in the future. Further advantages are the high security standards, vendor independent interoperability as well as the possibility to use different transport protocols and standards for communication e. g. MQTT, AMQP or OPC UA over TSN [6, 12, 20, 24, 28]. This paper defines a three layer skill architecture (Fig. 2) for the robot control, like it is done e. g. in [1, 4, 14, 25, 26, 34]. The naming and definition of the layers in the architecture is orientated on literature which already used OPC UA skills and skill-based engineering [5, 8, 9, 10, 11, 19, 30, 36]:

- Atomic skills: Low level (mechatronic) actions like code for simple moving operations, sensor based moving or actions from tools e.g. move linear, open gripper, start milling spindle with defined speed or movement along edge in cooperation with a laser sensor.
- Composite skills: A combination (sequence or parallelization) of atomic skills and composite skills. In this paper we are looking for manufacturing skills which represent the ability to manufacture a geometric feature by executing a defined parametrized action on a robot or machine.
- Task: A task in this case is the whole manufacturing process of a part. A task is a sequence of skills. It can be programmed by a user or is coordinated e. g. by an edge device. A user should be able to easily program tasks by adding skills in a sequence.

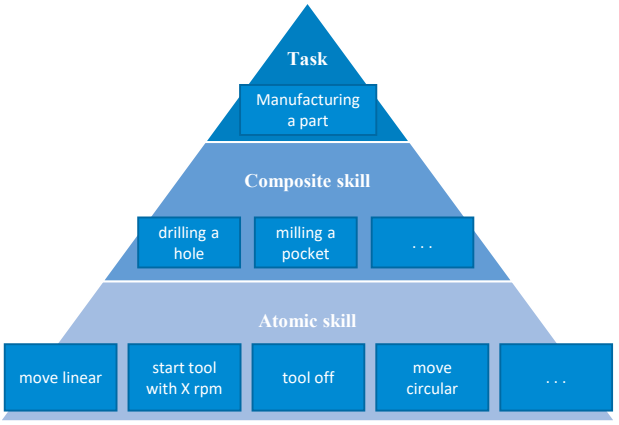

Fig. 2. Three layer architecture of tasks, composite skills and atomic skills

With the smallest possible number of defined parameters, the skills can be executed flexibly and collision-free by a client on a machine. A manufacturing skill could be, for example, the milling of pockets or drilling of holes. In the future, it is conceivable that machine or system manufacturers will provide a set of predefined skills for their customers on their controller. The benefits of defining skills on the machine are:

- Distribution of different production tasks for individual products over several machines is easy to implement.
- Production tasks can be easily adapted to changing factors based on a few parameters.
- The capabilities of a machine are mappable by its skills. Thus the skills become part of the administration shell of a machine.
- Easier programming of the systems by users as they can immediately access the predefined skills. This can save effort and money.
- Better overview of the manufacturing process, because skills are more intuitive (high level code) than for example G-Code (low level code).
- Ability to provide a standardized programming interface based on skills across the entire production. This way, a machine-independent service-oriented architecture (SOA) can be implemented.
- Skills can be reused on similar machines.

For the implementation of manufacturing skills, the initial focus was on functions, that are integrated on shop floor programming (SFP) interfaces of milling machines and are therefore already familiar. Due to SFP various manufacturing tasks can be programmed by the operator by entering just a few parameters directly at the control panel. In the future it will be investigated whether more complex tasks can be implemented. In this paper, the development of a manufacturing skill is shown using a drilling process as an example. The drilling process should run autonomously on the machine control system without a crash. A KUKA KR 300 with a KRC4 controller and mounted 2 kW milling spindle is available as a testbed for this purpose.

4. Implementation of a skill

To define the drilling skill mentioned above, first considerations were made about the motion sequences and the required parameters. It should be possible to drill holes on the upper surface and on all side surfaces of a component on the machine table. The drill holes need a defined diameter, depth, center point and direction vector. For this, it was evaluated whether the required parameters can be read from a 3D CAD model. Due to its open and well documented API and object based structure Autodesk's Fusion 360 was chosen as the first CAD system to test. The used file format is the *.f3d format. The API Object Model used in Fusion can be found in [3]. When using the *Hole Feature* the following parameters are stored in the saved file:

- Position of starting point / center of hole (X, Y, Z)
- Direction vector of the hole
- Diameter
- Depth
- Angle of a blind hole (depth of blind hole has to be added, the default value of 118° is used)

In addition if a thread is available all thread parameters can also be read out. Initially, however, the focus was on holes without threads. The parameters required for production are later read from a CAD project and transferred to the robot controller via OPC UA. To make the KUKA controller capable of processing the OPC UA NodeSet and to be able to transfer the data reliably with a modifiable server, it was extended by a Beckhoff controller of type CP2212-0000. The controller is used as an OPC UA adapter between the robot and clients connected to the robot. Although KUKA offers an OPC UA server as an upgrade, there are only a few configuration options and no possibilities to create own OPC UA methods. The Beckhoff controller is more configurable and additionally used as an extended Human-Machine-Interface (HMI). However, all motion commands belonging to a skill are implemented directly on the KRC4. The Beckhoff controller only provides the OPC UA server, carries out some calculations of the transmitted parameters and transfers them to the robot. The communication between the hardware is shown in Fig. 3.

In [11, 18] it was explained how an OPC UA skill can be implemented as a state machine. The approach is based on the Companion Specification 10: OPC UA Programs [23], because OPC UA methods should be used only for short operations. Each defined skill has a state machine with the states: *Running*, *Suspended*, *Halted* and *Ready*. By calling the released methods *Start*, *Suspend*, *Resume*, *Reset* or *Halt* the skill can be controlled. The corresponding diagram is shown in Fig. 4. This structure is used for the production of geometric features which are send from the CAD software to the robot. In a subfolder of the skill, all associated parameters are listed and can be changed by authorized users.

4.1. CAD add-in

Using the Fusion 360 API [3], a script was programmed that examines the CAD model for hole features and stores the required data for each hole. After scanning the component, the detected hole features with the corresponding parameters are shown on a GUI (graphical user interface). If a feature in the GUI is chosen, it is highlighted, and the CAD part rotates. This allows the operator to assign the listed features to the CAD model and to check the chosen data beforehand. Using Free OPC UA library for C++ and Python [13] a connection to the OPC UA server was established. The OPC UA library had to be installed in a virtual python environment, that is the same version as the one used by the Fusion 360 API to make it work reliably. Inside the python script, the folder in which the package is stored is then added to the PYTHONPATH variable. This installation method is mandatory to avoid deletion of the package during a Fusion 360 Update. The add-in is connecting as

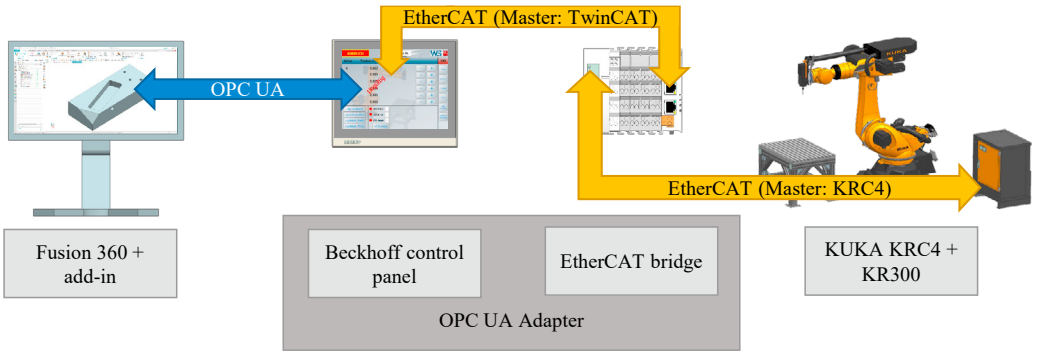

Fig. 3. Communication between the devices

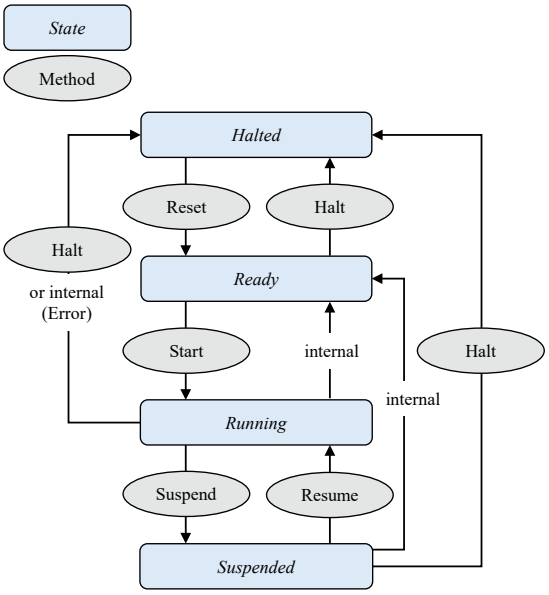

Fig. 4. State machine of a OPC UA program cf. [23]

a Client and asks the OPC UA server whether it can perform a drilling skill. If the machine has the required skill implemented, the machining parameters are stored in the variables assigned for the skill. By calling the methods described in Fig. 4, the client can control the robot. The methods can be called by clicking the referred GUI-Button. Whether a method is callable or not is dependent on the current program state of the robot. According to the state variable, that is repetitively retrieved from the server, the related GUI-Buttons are enabled or disabled. The current value of the program state variable as well as the program state transition variable is displayed in the GUI. Within the add-in´s current version, it is possible to send one feature at a time. To speed up the process it is conceivable to specify a manufacturing order, that can be created by the operator inside the GUI to send the next feature automatically, when the previous one is manufactured.

4.2. OPC UA adapter

As the Beckhoff controller is working as an OPC UA Adapter between any client and the robot, it coordinates the architecture of the OPC UA server, the user administration and provides the methods and skill parameters for the network. The adapter manages, calculates and forwards received parameters to the robot controller. Communication between the Beckhoff system and the KUKA KRC4 was established with the aid of a bridge terminal of type EL6695-1001. For this, a script had to be programmed that cyclically exchanges information between the robot controller and the adapter. If a client is connected to the OPC UA Server and has the appropriate rights, it is possible to call the methods of the drilling skill. In Fig. 5 the OPC UA structure of a drilling skill on the server is shown. The client transfers the predefined parameters to the server. On the OPC UA adapter all parameters are checked in advance and are converted if necessary for the robot controller. If the start method is called by a client, the individual parameters are transferred to the robot control and a corresponding KUKA Robotic Language (KRL) program is started, which executes the motion. The required parameters are transferred from a CAD model with the help of the plug-in described in section 4.1.

It is possible to intervene in the process at any time. The adapter monitors the actual skill state and the connected client is able to stop the movement by a halt or suspend call. At the moment the system does not monitor further process parameters or manipulates parameters during a skill process. Theoretically it would be possible, e.g. to constantly pass on a process variable to a condition monitoring system. Furthermore, the operator receives information about the current state of the system at any given time on the HMI provided on the Beckhoff system. Skills can also be started manually on the HMI. After an emergency stop or a detected fault, the operator can be offered instructions on the HMI to solve the problem or check the system. For example after an emergency stop occured during a skill process the operator has to check and confirm whether the home posi-

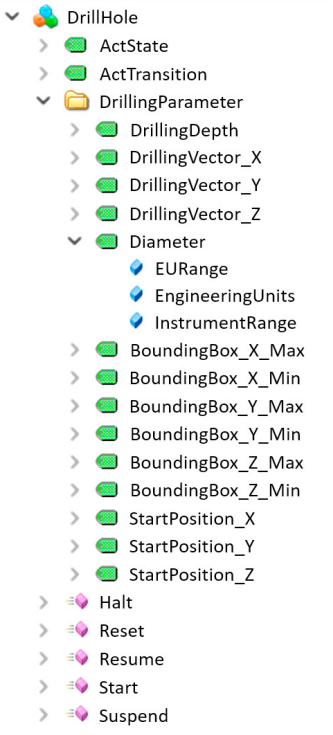

Fig. 5. OPC UA Structure of drilling skill

tion of the robot is reachable without any collision. If necessary the operator can move the robot into a safe position before confirming.

4.3. Robot programming

The motion sequence of the kinematic is implemented on the robot controller. The skills are parametrized KRL Code which receives the input parameters via the OPC UA adapter. The code of the skills on the robot controller is comparable with function blocks that are influenced in real-time by the OPC UA adapter. All movements during the skill are coordinated by the robot controller. The parametrized code can be executed very flexibly, so the same task can be used at different locations and in different orientations in the machine. A skill can consist of several atomic skills, which are stored on the robot controller as described in section 4.4. The robot starts a new machining process from its predefined home position. The home position was changed with respect to the KUKA standard home position, axis A3 and axis A4 are rotated by +10° and -10° respectively. This has been changed in order to exclude singularities in the home position and to prevent the hand axes from winding up while moving out of the home position. Previously, stronger rotations of the wrist axes occurred. At first the parameter for the diameter of the drill hole is called. A suitable drilling tool is chosen according to the diameter of the drill hole and taken from the tool magazine. In addition to the drilling parameters, the part size is sent from CAD using the coordinates stored in Fusion 360 as *boundingBox3D*. Later this information should be retrieved from the administration shell of the product. This can be done e. g. when the product is delivered to the cell and therefore the current part size is known before the process starts. At the beginning of production, the component is clamped at a defined zero-point position. With the information of the component size available, a more precise calibration of the component could be automated in the future with a further skill. Using the transferred component sizes, a dome is calculated from the center of the blank around the component. On this dome the Tool Center Point moves along to the drill hole to avoid collisions with the component (see Fig. 6). For drilling tools, the Tool Center Point is always defined at the end of the tool. With an implemented tool database and knowledge of the exact length of the current tool, the Tool Center Point can be adapted for each tool. This is already state of the art for classic machine tools.

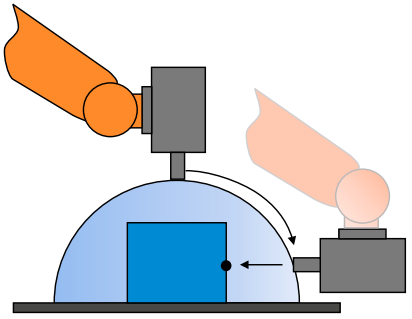

Fig. 6. Movement of the robot around an individual part

To select the target point for a drill hole, the X, Y, Z position of its center and its direction vector are read in. These parameters are used to determine the target point of the movement and the necessary rotation angles of the tool. Depending on the direction vector of the hole, an axis rotation around the tool axis must be calculated so that the robot wrist does not collide with itself, the periphery or the cable feeds to the spindle are not pinched. For this purpose, the Euler rotation angles between the Z-axis of the workpiece coordinate system and the direction vector of the hole are evaluated and a compensating rotation around the tool axis is calculated. Through a circular movement, the robot approaches the target point from the starting point via a calculated intermediate point. With a linear movement, the robot moves in front of the component surface to be machined and starts the drilling process. For this purpose, the spindle is started and the drilling to the defined depth is performed in several steps. After completing a skill, the robot waits a short time for a new skill command. After the complete machining of a component, the robot ends the process at the defined home position.

4.4. Skill structure

The three layer architecture for the skill-based programming is described in section 3. It is possible to divide a task in sev-

eral atomic and composite skills or a composite skill into several atomic and composite skills, so functioning programs can be reused to create new skills on a system. For example, the drilling skill could be divided into the following atomic and composite skills shown in Fig. 7.

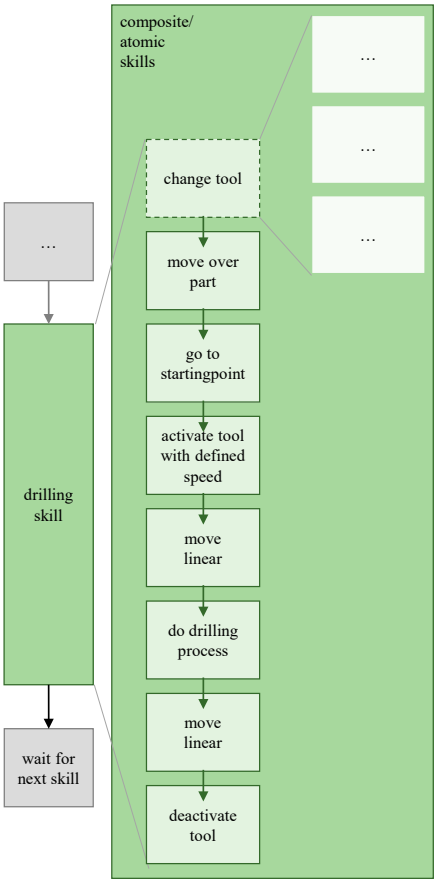

Fig. 7. Structure of the drilling skill

A further realized application example is the automated manufacturing of pockets. This skill supports milling of a rectangle in any size and orientation in a defined range of parameters. Since it was possible to reuse the code for approaching the starting point, we were able to focus directly on the implementation of the code for milling a pocket. The machine decides independently to use the largest possible end mill cutter. The tool path the machine travels to manufacture the pocket is calculated on the robot controller depending on the milling cutter diameter that is now available. The process starts in the middle of the pocket and moves outwards step by step. The corner radius has been neglected so far. Choosing a clear structure through skills is therefore useful to make already generated knowledge accessible and reusable. This structure can save time in the implementation of new skills.

Another advantage of skill-based programming is that dependencies can be described between skills and their corresponding features. Given a product that consists of several parts with a defined contact area, for example a hole in one and a cylinder on the other part, it is possible to intervene in the second process if the tolerances of the first are not met. For this adaption, the deviations from the target value are determined and saved. The manufacturing parameters of the corresponding counterpart is dynamically adjusted in a later manufacturing step with the help of the stored information. This is particularly suitable for applications in which the fit and not the individual dimensions are decisive, e. g. wood clamp connections. In principle, the dependencies of two skills to each other can also be determined from CAD, e. g. via extracting assembly constraints. This idea has not been practically implemented in the context of the paper. Research into solving this problem and how the dependencies can be stored in the OPC UA skills is in progress.

5. Evaluation

To test the skill-based approach, different CAD models were manufactured on the test bed. In Fig. 8 the evaluation process is shown. A CAD model with two holes and two pockets on different surfaces of a block and different direction vectors were created in Fusion 360. The zero point and the coordinate system in CAD must have the same reference to the component as the calibrated base in the production cell to the existing part. In this case the digital model was sketched from the origin in the positive X, Y plane and extruded in the positive Z direction. An unmachined block of identical outer geometry was positioned in the robot cell with the dimensions length = 200 mm, width = 100 mm and height = 50 mm. The origin to fix the component in a corresponding base coordinate system is predefined and marked in the robot cell. This ensures that the correct machining parameters are transferred. The plug-in was opened in Fusion 360 and the drill holes and pockets were scanned. With the help of the GUI the existing holes, pockets and values can be checked. The desired skill can be selected and started by calling the run method. The manufactured and described part is shown in Fig. 9. In addition to this part, many different tests were repeated with parts of different sizes and different positions of the holes and pockets for validation. For the surface and all side faces and for varying direction vectors it was verified that the manufacturing skills can be implemented. Neither a previous CAM simulation nor active programming or adjustment of the programming by a user was necessary. The functionality of the skill-based manufacturing could thus be successfully evaluated. It was possible to save a lot of time from CAD design to production for individual parts. An experienced CAM user needed around 10 minutes to simulate and export the required code. Thanks to the new approach this time can be reduced to selecting the features and clicking the start button. The more complex the components become, the greater the time difference will be and further potential for automation of the process is available. It was shown that path planning can be implemented even on complex systems with industrial robots without driving a crash. Special collision scenarios, such as collisions with

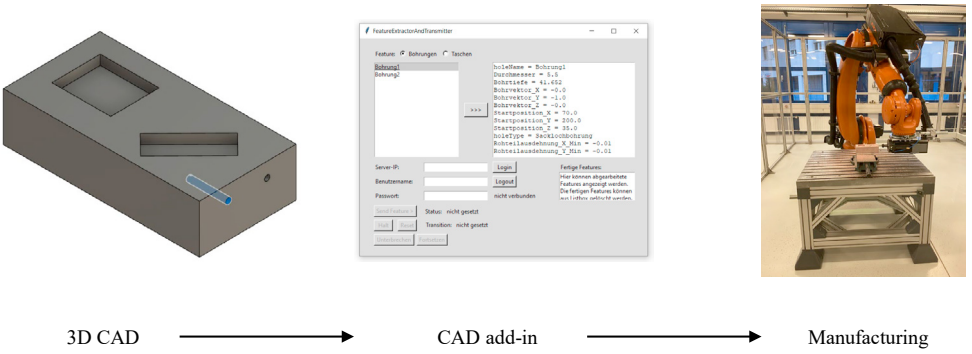

Fig. 8. Manufacturing of parts on the test bed

the vice on the workbench, cannot be ruled out. Methods solving this problem will be investigated in the future. The process reliability of skill-based manufacturing was thus demonstrated. Since the clamping point is defined in the manufacturing system, the skills can also be transferred easily to larger or smaller systems, if the controller can be programmed in KRL. Skill-based manufacturing could be a key technology for flexible and economical production of parts in small batch sizes.

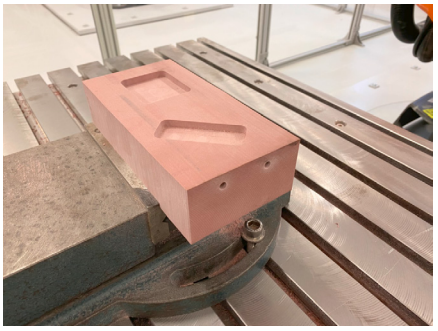

Fig. 9. Manufactured part

6. Conclusion

In this paper typical manufacturing processes were implemented which could be started flexibly and reliably with the help of CAD information. A drilling process and milling of a pocket was carried out on a KUKA Quantec KR 300. A flexible manufacturing code was stored on the robot controller, and its parameters were shared on an OPC UA server. The OPC UA server was created on a Beckhoff controller, which serves as an adapter between the robot and the clients connected to it. Using a CAD model, manufacturing parameters for holes and pockets were determined and transferred to the robot via OPC UA. The skills can be started, suspended, resumed, halted and reseted via OPC UA methods. During programming it was considered that the robot executes the manufacturing steps autonomously and in a safe manner. Until now, the manufacturing process cannot be checked in advance for unexpected collisions (e. g. tool collision with the vice). Different possibilities are still being evaluated to exclude collisions in advance, for example by simulation. The commissioning of the OPC UA server can be simplified if machine manufacturers offer the option of creating flexible OPC UA servers on their controllers and an additional adapter is no longer necessary. The time saved by skill-based manufacturing could be demonstrated for individual parts. The system will be expanded with additional skills and functions, e. g. material information is to be transferred from CAD in order to be able to adjust production speeds. Furthermore, future work will explore how dependencies of skills can be realized. This should open possibilities to adjust corresponding counterparts when individual production tolerances are exceeded and to minimize material waste. Skills enable a uniform programming interface in production. Workers could thus quickly create a program based on intuitively comprehensible sequences and could be trained faster to use the machine controls. Skills are an important step towards Production Level 4 [33], a production system that provides ideal support for humans with a high degree of automation in flexible process tasks. In this future development, it is likely that machines will perform certain tasks independently and thus become ”smarter”. The goal is no longer programming machines with low level code, but rather to use high level commands they solve on their own. Manufacturing based on skills could be a key technology to allow economical and individual production with small batches.

References

[1] Andersen, R.H., Solund, T., Hallam, J., 2014. Definition and initial case-based evaluation of hardware-independent robot skills for industrial robotic co-workers, in: Proceedings for the joint conference of ISR 2014, 45th International Symposium on Robotics, Robotik 2014, 8th German Conference on Robotics. VDE-Verl., Berlin and Offenbach.

[2] Andersen, R.S., Nalpantidis, L., Krüger, V., Madsen, O., Moeslund, T.B., 05.01.2014 - 08.01.2014. Using robot skills for flexible reprogramming of pick operations in industrial scenarios, in: Proceedings of the 9th International Conference on Computer Vision Theory and Applications, SCITEPRESS - Science and and Technology Publications. pp. 678–685. doi:10.5220/0004752306780685.

[3] Autodesk, . Fusion 360 api. URL: http://help.autodesk.com/view/fusion360/ENU/?guid=GUID-A92A4B10-3781-4925-94C6-47DA85A4F65A.

[4] Björkelund, A., Edstrom, L., Haage, M., Malec, J., Nilsson, K., Nugues, P., Robertz, S.G., Storkle, D., Blomdell, A., Johansson, R., Linderoth, M., Nilsson, A., Robertsson, A., Stolt, A., Bruyninckx, H., 2011. On the integration of skilled robot motions for productivity in manufacturing, in: 2011 International Symposium on Assembly and Manufacturing, IEEE, Piscataway, NJ. pp. 1–9. doi:10.1109/ISAM.2011.5942366.

[5] Brandenbourger, B., Durand, F., 2018. Design pattern for decomposition or aggregation of automation systems into hierarchy levels, in: Proceedings 2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE, Piscataway, NJ. pp. 895–901. doi:10.1109/ETFA.2018.8502627.

[6] Bundesamt für Sicherheit in der Informationstechnik, . Sicherheitsanalyse open platform communications unified architecture (opc ua). URL: https://www.bsi.bund.de/DE/Publikationen/Studien/OPCUA/OPCUA_node.html.

[7] Dai, F., Wahrburg, A., Matthias, B., Ding, H., 2016. Robot assembly skills based on compliant motion, in: Proceedings of ISR 2016: 47st International Symposium on Robotics. VDE and IEEE, Frankfurt am Main and Piscataway, NJ.

[8] Dorofeev, K., Cheng, C.H., Guedes, M., Ferreira, P., Profanter, S., Zoitl, A., 2017. Device adapter concept towards enabling plug&produce production environments, in: 2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation, IEEE, Piscataway, NJ. pp. 1–8. doi:10.1109/ETFA.2017.8247570.

[9] Dorofeev, K., Profanter, S., Cabral, J., Ferreira, P., Zoitl, A., 2019. Agile operational behavior for the control-level devices in plug&produce production environments, in: Proceedings, 2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE, Piscataway, NJ. pp. 49–56. doi:10.1109/ETFA.2019.8869208.

[10] Dorofeev, K., Wenger, M., 2019. Evaluating skill-based control architecture for flexible automation systems, in: Proceedings of the IEEE International Conference on Emerging Technologies And Factory Automation (ETFA). doi:10.1109/ETFA.2019.8869050.

[11] Dorofeev, K., Zoitl, A., 2018. Skill-based engineering approach using opc ua programs, in: Proceedings IEEE 16th International Conference on Industrial Informatics (INDIN), IEEE, Piscataway, NJ. pp. 1098–1103. doi:10.1109/INDIN.2018.8471978.

[12] Eckhardt, A., Muller, S., 2019. Analysis of the round trip time of opc ua and tsn based peer-to-peer communication, in: Proceedings, 2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE, Piscataway, NJ. pp. 161–167. doi:10.1109/ETFA.2019.8869060.

[13] FreeOpcUa, . Open source c++ and python opc-ua server and client libraries and tools. URL: http://freeopcua.github.io/.

[14] Gat, E., 1998. On three-layer architectures, in: Kortenkamp, D., Bonasso, R.P., Murphy, R. (Eds.), Artificial intelligence and mobile robots. AAAI Press, Menlo Park, Calif.

[15] Heikkila, T., Ahola, J.M., 2018. Robot skills - modeling and control aspects, in: Applications, I.I.C.o.M., and, E.S. (Eds.), 2018 14th IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications (MESA), IEEE, Piscataway, NJ. pp. 1–6. doi:10.1109/MESA.2018.8449184.

[16] Hermann, J., Rübel, P., Birtel, M., Mohr, F., Wagner, A., Ruskowski, M., 2019. Self-description of cyber-physical production modules for a product-driven manufacturing system. Procedia Manufacturing 38, 291–298. doi:10.1016/j.promfg.2020.01.038.

[17] Herrero, H., Moughlbay, A.A., Outón, J.L., Sallé, D., de Ipiña, K.L., 2017. Skill based robot programming: Assembly, vision and workspace monitoring skill interaction. Neurocomputing 255, 61–70. doi:10.1016/j.neucom.2016.09.133.

[18] Kaspar, M., Bock, J., Kogan, Y., Venet, P., Weser, M., Zimmermann, U.E., 2018. Tool and technology independent function interfaces by using a generic opc ua representation, in: Proceedings 2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE, Piscataway, NJ. pp. 1183–1186. doi:10.1109/ETFA.2018.8502647.

[19] Malakuti, S., Bock, J., Weser, M., Venet, P., Zimmermann, P., Wiegand, M., Grothoff, J., Wagner, C., Bayha, A., 2018. Challenges in skill-based engineering of industrial automation systems *, in: Proceedings 2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE, Piscataway, NJ. pp. 67–74. doi:10.1109/ETFA.2018.8502635.

[20] OPC Foundation, a. Opc specification part 14: Pubsub: Release 1.04.

[21] OPC Foundation, b. Opc ua robotics companion specification: Part 1.

[22] OPC Foundation, c. Opc ua specification part 5: Information model: Release 1.04. URL: https://opcfoundation.org/developer-tools/specifications-unified-architecture/part-5-information-model/.

[23] OPC Foundation, d. Opc ua specifikation part 10: Programs: Release 1.04. URL: https://opcfoundation.org/developer-tools/specifications-unified-architecture/part-10-programs/.

[24] OPC Foundation, e. Unified architecture (ua). URL: https://opcfoundation.org/about/opc-technologies/opc-ua/.

[25] Pedersen, M.R., 2014. Robot Skills for Transformable Manufacturing Systems. Ph.D. thesis. Aalborg University Copenhagen. Copenhagen.

[26] Pedersen, M.R., Nalpantidis, L., Andersen, R.S., Schou, C., Bøgh, S., Krüger, V., Madsen, O., 2016. Robot skills for manufacturing: From concept to industrial deployment. Robotics and Computer-Integrated Manufacturing 37, 282–291. URL: https://www.sciencedirect.com/science/article/pii/S0736584515000575, doi:10.1016/j.rcim.2015.04.002.

[27] Perzylo, A., Somani, N., Profanter, S., Kessler, I., Rickert, M., Knoll, A., . Intuitive instruction of industrial robots: Semantic process descriptions for small lot production, in: IROS 2016 2016, pp. 2293–2300. doi:10.1109/IROS.2016.7759358.

[28] Pfrommer, J., Ebner, A., Ravikumar, S., Karunakaran, B., 2018. Open source opc ua pubsub over tsn for realtime industrial communication, in: Proceedings 2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE, Piscataway, NJ. pp. 1087–1090. doi:10.1109/ETFA.2018.8502479.

[29] Pfrommer, J., Stogl, D., Aleksandrov, K., Schubert, V., Hein, B., 2014. Modelling and orchestration of service-based manufacturing systems via skills, in: IEEE [International Conference on] Emerging Technologies and Factory Automation (ETFA), 2014, IEEE, Piscataway, NJ. pp. 1–4. doi:10.1109/ETFA.2014.7005285.

[30] Profanter, S., Breitkreuz, A., Rickert, M., Knoll, A., . A hardware-agnostic opc ua skill model for robot manipulators and tools. doi:10.1109/ETFA.2019.8869205.

[31] Profanter, S., Tekat, A., Dorofeev, K., Rickert, M., Knoll, A., 2019. Opc ua versus ros, dds, and mqtt: Performance evaluation of industry 4.0 protocols, in: Proceedings, 2019 IEEE International Conference on Industrial Technology (ICIT), IEEE, [Piscataway, New Jersey]. pp. 955–962. doi:10.1109/ICIT.2019.8755050.

[32] Schou, C., Madsen, O., 2017. A plug and produce framework for industrial collaborative robots. International Journal of Advanced Robotic Systems 14, 172988141771747. doi:10.1177/1729881417717472.

[33] SmartFactoryKL e.V., . Production level 4. URL: https://smartfactory.de/production-level-4/.

[34] Steinmetz, F., Weitschat, R., 2016. Skill parametrization approaches and skill architecture for human-robot interaction, in: 2016 IEEE International Conference on Automation Science and Engineering (CASE), IEEE, Piscataway, NJ. pp. 280–285. doi:10.1109/COASE.2016.7743419.

[35] Verein Deutscher Werkzeugmaschinenfabriken e.V., . Umati. URL: https://vdw.de/technik-und-normung/umati/.

[36] Zimmermann, P., Axmann, E., Brandenbourger, B., Dorofeev, K., Mankowski, A., Zanini, P., . Skill-based engineering and control on field-device-level with opc ua, in: IEEE ETFA 2019. doi:10.1109/ETFA.2019.8869473.