



Feasibility Evaluation of Milling Designs Using Multi-Agent Systems

S. Plappert , C. Becker, P. C. Gembarski and R. Lachmayer

Leibniz Universität Hannover, Germany

 plappert@ipeg.uni-hannover.de

Abstract

During product development, many decisions have to be made that affect the entire product life cycle and often lead to errors that cause additional effort. To proactively support the engineer in evaluating his design in a CAD program, in this paper an approach to evaluate milling designs using a multi-agent system (MAS) is presented. The CommonKADS method is used and the MAS is validated against an application example of a gearbox housing that has been checked for design guidelines, standards, and tool or machine portfolios.

Keywords: multi-agent systems, design evaluation, CommonKADS, knowledge-based engineering (KBE), computer-aided design (CAD)

1. Introduction

How can errors in product development be avoided, which are caused by lack of knowledge about the manufacturing process or by oversight mistakes? A possible error in the design of milling components can be that the tool run-out is forgotten for pockets or that the tool must be changed for the radius used because it is not available in the standard magazine of the machine. These errors lead either to changes in the design or additional costs in production. One possibility to eliminate these errors is the 4-eyes-principle, where a second designer has to check the design again. This leads on the one hand to an additional effort, where it is not guaranteed that all mistakes are found, and on the other hand the designer also needs to know the complete process chain.

For this reason, in this paper, we take the approach of digitizing the 4-eyes-principle by proactively supporting the designer in checking his design in the CAD program. To do so, we will analyze the 3D product model using stored manufacturing knowledge and adjust the CAD model automatically.

Therefore, the sustainable storage, archiving of information and knowledge from the specific domains is becoming increasingly important (Huet *et al.*, 2007; Kratzer *et al.*, 2011). One possibility to represent specific domain knowledge are knowledge-based systems (KBS), which use digitally processed expert knowledge and simulate the actions of an expert in going through a solution process (Milton, 2008). Knowledge-based engineering systems (KBES), which are already increasingly used in the design phase of the product development process and for product configuration (Plappert *et al.*, 2020), are considered a special form of a KBS because they have advanced data processing and CAD support, so they support problem-solving through analysis and computation (Hirz *et al.*, 2013; Verhagen *et al.*, 2012). In addition, a shift in the focus of design from the creation of a single solution to a solution space or set of variants can be observed (Gembarski, 2020a).

Nevertheless, the complete solution space is usually represented by explicit domain knowledge, so that it can be regarded as *closed*. This approach makes the programming extensive and the adaptation of the system to new requirements is costly. For this reason, new approaches are being sought in research

to automatically check an *arbitrary* design for its compatibility with the solution space and then adapt the CAD design accordingly.

Multi-agent systems (MAS) and distributed artificial intelligence (AI) are assuming increasing importance in this regard, as the geographical distance between development and manufacturing increases (Li, 2007; Liu *et al.*, 2004). On the one hand, as a decentralized AI, the MAS can be more easily maintained and extended when changes occur since individual agents are considered separately and these can be exchanged. On the other hand, e.g., manufacturing knowledge can be determined locally and made available to the development department via agents. This can be done by using the expertise of experts from different domains such as development, manufacturing, suppliers, and other functional groups (Wetmore *et al.*, 2010).

This paper is organized as follows: Section 2 presents the theoretical background and related work for MAS and defines the research questions. The methodological approach and specification for MAS are presented and applied to the architecture in Section 3. Then, in Section 4, the MAS is applied to the application example of a gearbox housing. The application of MAS to the evaluation of milling designs is discussed in Section 5 and summarized in Section 6.

2. Related Work and Theoretical Background

Agent-based technologies have their beginnings in the 1990s and are based on the paradigm of distributed artificial intelligence and program agents (Dostatni *et al.*, 2016). In general, an agent can be assumed to be an autonomous, computational entity that perceives its environment through sensors and acts on it through effectors (Weiss, 2000). They represent, e.g., humans or machines and act autonomously within their boundaries, so that they serve as a work facilitator or aid for the user (Eymann, 2003). According to Weyns (Weyns, 2010), agents have three services: the *Perception Service* enables the sensing of the environment and representation of it, the *Action Service* coordinates the operations of the agents, and the *Communication Service* manages the interactions among the agents. Because conflict can arise when multiple stimuli are applied to the sensor for an agent's perception, Brooks (Brooks, 1987) designed a subsumption architecture that assigns subfunctions to the sensor, called *behaviours*. These act synchronously and terminate only if they are not interrupted by the other behaviours.

A multi-agent system consists of many agents that typically interact with each other by exchanging messages over a computer network infrastructure (Wooldridge, 2009). Characteristically, individual agents have limited knowledge about the system and problem, and thus can only contribute partial solutions (Eymann, 2003). For this reason, agents must cooperate and share knowledge when solving problems to produce a valid solution.

Three research directions can be identified in the literature on MAS in product development (Plappert *et al.*, 2021). First, MAS can be built from agent-based functions, where CAD models can autonomously adapt to new situations by representing agents as intelligent features (Fougères and Ostrosi, 2018). Second, Chu *et al.* (Chu *et al.*, 2009) discuss how a MAS can be used to support decentralized engineering work, e.g., by enabling a geographically distributed group of users to work synchronously and collaboratively on a 3D assembly over the Internet. Third, MAS are used as assistance systems for design engineers to support them in decision-making for DfX (Design for Excellence). This usually involves external intervention on the design, using different prior knowledge and expertise to evaluate a design (Gembariski, 2020b). To directly support the designer in modeling his design, MAS are increasingly used in combination with CAD systems (Dostatni *et al.*, 2016).

A few research works have focused on assisting the designer in evaluating the manufacturability of components using multi-agent systems. Feng (2005) identifies preliminary process planning in the early stages of product development to evaluate manufacturability in terms of primary manufacturing processes, selection of resources and equipment, and estimation of manufacturing costs. Medani and Ratchev (2006) analyze a STEP AP224-compliant product data model using design and manufacturing agents to perform rapid manufacturability of products in extended enterprises. Jia *et al.* (2004) and Mahesh *et al.* (2007) developed a web-based multiagent system to support distributed digital manufacturing. Here, manufacturability is evaluated and process planning and scheduling are performed, among other tasks. Existing literature only indicates potential manufacturing conflicts to

the designer, but it does not provide support for resolving the conflicts or resolving them independently.

Modeling MAS frameworks can be divided into two approaches, either the MAS framework is built for a specific area of interest or a holistic approach is taken (Palanca *et al.*, 2020). Probably the most widely used holistic MAS framework in the last two decades is JADE (Java Agent Development Framework) (Bergenti *et al.*, 2020), which is FIPA compliant and builds the network from agent containers (Bellifemine *et al.*, 2001). Among the newer representatives of MAS is the platform SPADE 3.0 (Smart Python Agent Development Environment), which is programmed in Python and uses open instant messaging protocols to enable distributed communication between humans, agents, and third-party elements (Palanca *et al.*, 2020). In addition, SPADE enables interaction between agents by exchanging information through predefined agent and behaviour classes.

Based on the theoretical background and related work on MAS, a possible architecture for MAS to evaluate CAD models using the platform SPADE will be investigated. The application area here is product development and in particular the design of milled components. For this purpose, the following research questions have been identified:

- How should a multi-agent system be designed to help the engineer evaluate milling designs for manufacturability and automatically adjust the CAD model?
- How can an architecture for the evaluation of CAD models be built using the CommonKADS method and SPADE as a MAS framework?

3. Multi-Agent System for Evaluation of Milling Designs

3.1. Methodological Approach and Specification

The MAS should overcome the hurdle of expert knowledge not being available centrally and the need to adapt to new conditions, e.g., in production due to new machines or tools, by storing knowledge in a decentralized manner and making it available to the engineer during the design process. To do this, the MAS has to meet requirements such as the perception of 3D models or the identification, storage, and processing of specific domain knowledge. Performing design reviews ensures that the product meets the requirements and thus serves as a mechanism to control the design.

To be able to map these requirements for a MAS in a structured way, the CommonKADS method is suitable, since it uses an approach to knowledge-based system development by capturing the most important characteristics of the system and its environment through the setup of separate models (Iglesias *et al.*, 1996). The CommonKADS methodology consists of six models (Fig. 1), which form different units of abstraction levels (De Hoog *et al.*, 1994; Kingston, 2001).

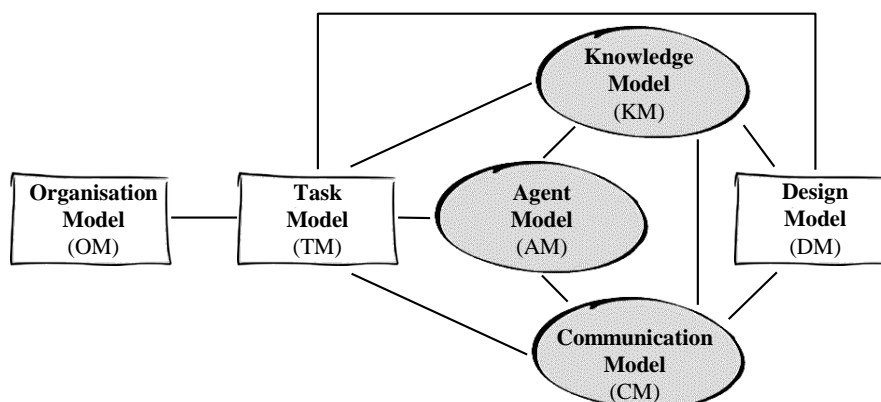


Figure 1. Relations between CommonKADS models (According to De Hoog *et al.*, 1994)

The *Organization Model* is a tool for analyzing the organization and identifying promising areas for knowledge-based system applications. The *Task Model* describes the tasks, independent of an agent, which are to be performed by the MAS. In the *Agent Model*, all relevant properties of agents are

described to perform the tasks. The agent can be a human, computer software, or other entity, which can perform the task. The exchange of information between the different agents is defined in the *Communication Model*. It has to be ensured that the sender and receiver are identifiable and that a control plan for transactions is developed. The *Knowledge Model* is the main focus of the CommonKADS methodology and describes the problem-solving knowledge that is assigned to an agent. Finally, the *Design Model* is developed, which combines the other models into a system and thus serves to describe the architecture and the technical design.

In the context of this research, the context or scope for the organizational model is in product development, particularly in the design phase of the product development process. The main task of the MAS is the virtual design review, in which 3D models are checked for their manufacturability based on standards and design guidelines. For this purpose, the main task is divided into subtasks and assigned to different agents.

For the agents to work, they need information. On the one hand, the knowledge to deal with situations, e.g., what can be manufactured with given tools. On the other hand, they need to know what the user wants. But the most important information is that of the model because it has to be matched with the external information. As part of this, the basic framework of the 3D model is examined, which consists of points/nodes, edges, surfaces, and relationships. These come from the *boundary representation* (B-Rep), which forms the geometric core of the CAD system. Feature recognition looks at the model from different viewpoints and has to read and adopt the information accordingly (Fougères and Ostrosi, 2018).

As the focus is on the evaluation of milling designs, the agents are provided with the specific domain knowledge for this purpose. When engineering milling designs, it is important to ensure that the design complies with standards and guidelines as well as that it can be manufactured with the available resources (tools, machines, material). In addition, the design rules for drilling have to be considered. The focus of the agents is on the machining of chamfers, radii, holes, slots, and steps. Here, on the one hand, it is checked whether the part can be manufactured with the given tools and if an optimization for tool selection is possible. On the other hand, design elements such as tool runouts, chamfers, or radii are checked and, if necessary, adjusted directly in the CAD model.

In reality, the experts are in constant contact, they discuss and exchange information. This behaviour is also aimed at the agents of the MAS. The agents must act independently and make decisions. Therefore, communication represents one of the fundamental properties of an agent (Bussmann *et al.*, 2004). To ensure the structure and function of the system, not only the hierarchy of the individual agents is important, but also their structure. Communication is the interface between the agents and the user. The exchange of information can take place via a *message-passing system* or a *shared-memory blackboard system*. The blackboard system operates over shared memory to which different agents have access. The more commonly used message-passing system transmits information directly to the receiver.

After applying the first five models of the CommonKADS methodology to the MAS use case for evaluating milling designs, the next chapter discusses the design model and presents the architecture.

3.2. Architecture of the Multi-Agent System

The agents from the agent model have several interfaces to the environment, as can be seen in Figure 2, to perform the tasks from the task model, e.g., direct contact with the user or access to external knowledge repositories. At the same time, the architecture shows the organization of the system in terms of knowledge sharing and represents the general structure of the MAS. The MAS for design evaluation of milling designs is based on the MAS platform SPADE 3.0 developed by Palanca *et al.* (Palanca *et al.*, 2020). A major advantage of SPADE is that it is programmed in Python, as it contains numerous artificial intelligence libraries and thus offers interesting extension possibilities for the MAS. Furthermore, FIPA compliance ensures a standardized structure of the development environment, which can be extended with additional FIPA-compliant agents. To enable the agents to communicate, a messaging platform must first be set up. In this case, a Prosody XMPP (Extensible Messaging and Presence Protocol) server, is set up on a virtual machine running the Ubuntu operating

system. The XMPP server also provides an opportunity for the MAS to communicate directly with the user via messengers, such as Spark or Pidgin, to have ambiguities resolved by the designer. The modularity of the agent templates is exploited when building the MAS by programming the resulting agents with them. They use different behaviour types depending on the function. Especially the *Cyclic-Behaviour* is useful when several messages are sent one after the other. For example, with user queries to approve changes. This is useful in a MAS for evaluating milling designs, which targets changes due to design errors or problems in manufacturing.

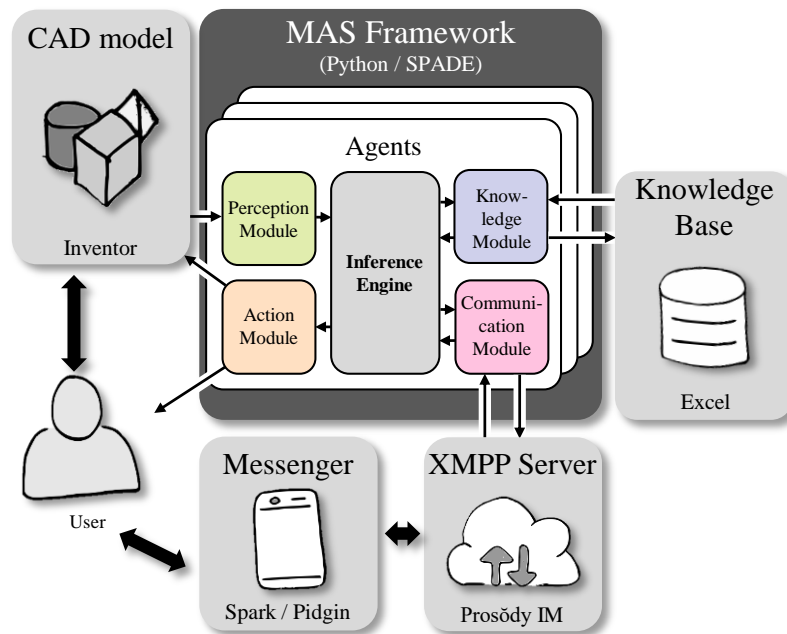


Figure 2. The architecture of the Multi-Agent System and its Interfaces

For the evaluation of CAD models and decision support for designers, an interface to a CAD program must be provided (Plappert *et al.*, 2022). Autodesk Inventor Professional is used as the CAD program since the B-Rep of the CAD model can be accessed via the API (Application Programming Interface). Each agent is itself a knowledge-based system, so the required knowledge in the form of standards or tool lists is stored as a relational database in Excel spreadsheets. Additionally, design guidelines are used as rules and aggregated in the inference engine. On the one hand, the result of the evaluation is documented within Python and saved as a text file, and on the other hand, the CAD model has been adapted accordingly.

3.3. Program Sequence of the Multi-Agent System

The evaluation of designs contains individual work areas, which are divided among experts. Based on this, the tasks are distributed to the agents according to the topic. For this purpose, the *Feedback Agent* is responsible for collecting and then passing on the information to the user. The information from the CAD model is read out and transferred by the *Info Agent*. The *Chamfer/ Fillet Agent* checks the chamfers and radii of the 3D model. This information is used by the *Design Agent* to request features from the user via XMPP messenger.

The *Manufacturing Agent* is available for planning the manufacturing steps. This runs through all features in the feature tree in the CAD model and checks whether they can be combined into one manufacturing step. These manufacturing steps are stored in an Action-Item-List (AIL).

With this information about the machining and a tool list, the *Tool Agent* is responsible for selecting the tools. For this purpose, the tool list is first read out and then the dimensions and surface specifications of the cuts are compared with the tools. The possible tool with the largest diameter is inserted into the AIL as the selected tool, and all others are appended as alternatives.

The optimization of the tools is taken over by the *Planning Agent*. For this purpose, the different tools in the AIL are counted and, depending on the number and possibility of use in the AIL, are used as the selected tool, and the alternative tools are adjusted. Since certain cuts require a tool runout so that they can be manufactured, the design agent checks this. Primarily, grooves are checked by examining the cut for radii at the end of the tool path. The tool agent checks the tools of the changed machining steps in the AIL and makes corrections if necessary. The AIL is then sent to the feedback agent.

The sequence of the MAS is shown in the form of a sequence diagram in Figure 3 for better understanding and represents the tasks and relationships of the agents.

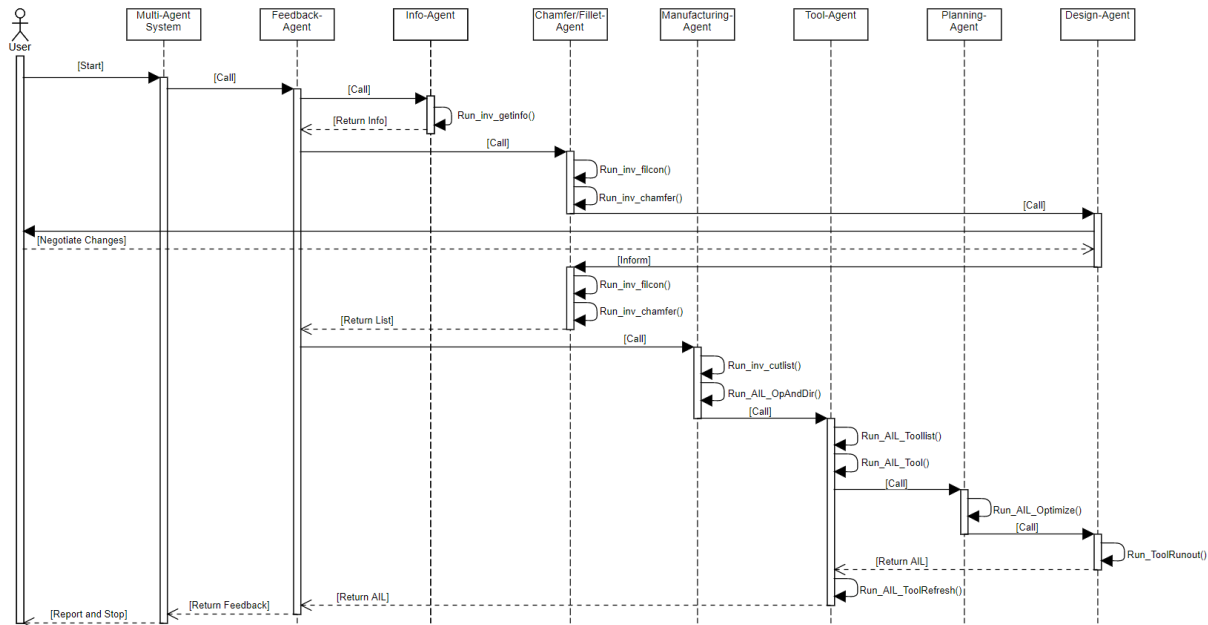


Figure 3. Sequence Diagram of the Multi-Agent System

4. Application Example

After reviewing the general form and function of the MAS architecture with the described interfaces, the next step is to build a prototype for evaluating a CAD design for milling designs. As an application example, the CAD model of a gearbox housing, which is completely machined on a milling machine, is analyzed. For this purpose, it is compared with the applicable standards, and it is checked whether the manufacturing is possible with the tools in the tool magazine of the machine. The tool list is stored in an Excel spreadsheet and can be edited by the user so that if changes are made, they can be easily updated or special tools for specific parts can be added.

In the following, the evaluation of the gearbox housing is carried out and the application of the MAS is presented from the engineer's point of view. First, the engineer designs the gearbox housing with the relevant functional surfaces and then starts the MAS. When the MAS starts it opens Inventor and loads the transferred model. After collecting metadata from the CAD model, the model is checked for radii and chamfers.

The user is also involved in this detection, as it is necessary to check whether these have been inserted for purely aesthetic reasons or if they are relevant for the function of the part. The communication to the user is done by highlighting the feature in the CAD program and by a query in the XMPP messenger. So, the agent communicates via a chat client with the user and continues the execution depending on his or her answers. If the features were only inserted for aesthetic reasons, they are suppressed in the CAD model and disregarded for further evaluation.

After the model has been prepared for the manufacturability evaluation, the next step is to define the machining steps using the feature tree. To do this, a list of machining steps is first created and the necessary milling tool position and machining direction are added. Then the possible tools for the

individual cuts are added, depending on dimensions and surface specifications. This table forms the first section of the Action-Item-List (AIL), as shown in Figure 4.

ID	Feature	Tool position	Tool direction	Alt. operations
1	['Extrusion2', 'Fillet4']	Vertical Cutting	+Z	['No Alternative']
2	['Hole1']	Vertical Drilling	+Y	[['Vertical Cutting', '+Y']]
3	['Extrusion3', 'Fillet5']	Horizontal Cutting	-X	['No Alternative']
4	['Hole2']	Horizontal Drilling	+X	[['Horizontal Cutting', '+X']]
5	['Hole3']	Horizontal Cutting	+X	[['Horizontal Drilling', '+X']]
6	['Extrusion4', 'Fillet6']	Horizontal Cutting	+Y	['No Alternative']
7	['Hole4']	Horizontal Drilling	+Z	[['Horizontal Cutting', '+Z']]
8	['Hole5']	Horizontal Cutting	+Z	[['Horizontal Cutting', '+Z']]
9	['Hole6']	Horizontal Drilling	-Z	[['Horizontal Cutting', '-Z']]

ID	Used tool	Alt. tool	Design Change
1	Shank_RF_20	['Mk 20', 'Shank_R_20', 'Shank_R_15', 'Shank_RF_15', 'Mk 10', 'Shank_RF_10', 'Shank_RF_5']	Added "Fillet4" for tool runout
2	Shank_RF_5	['Drill_8']	None
3	Shank_RF_20	['Mk 20', 'Shank_F_20', 'Shank_R_20', 'Shank_F_15', 'Shank_R_15', 'Shank_RF_15', 'Mk 10', 'Shank_RF_10', 'Shank_RF_5']	Added "Fillet5" for tool runout
4	Drill_3.3	['No Alternative']	None
5	Shank_RF_20	['Shank_R_20', 'Shank_F_15', 'Shank_R_15', 'Shank_RF_15', 'Shank_RF_10', 'Shank_F_20']	None
6	Shank_RF_20	['Shank_F_20', 'Shank_R_20']	Added "Fillet6" for tool runout
7	Drill_3.3	['No Alternative']	None
8	Shank_RF_20	['Shank_F_15', 'Shank_RF_15', 'Shank_RF_10', 'Shank_F_20']	None
9	Drill_5	['No Alternative']	None

The MAS needs 213.2741107940674s.

Figure 4. Action-Item-List for the Design as Output in a Console

As soon as the machining steps have been defined, the tools that can be used for this machining are checked. In addition, optimization is carried out to reduce the number of tools, so that the tool change can be minimized, which in turn leads to time savings in production. Finally, the model is checked for missing tool runouts, which are inserted directly into the CAD model. In addition, the AIL is extended so that the tools used are expanded in the lower part and the changes in the design are documented.

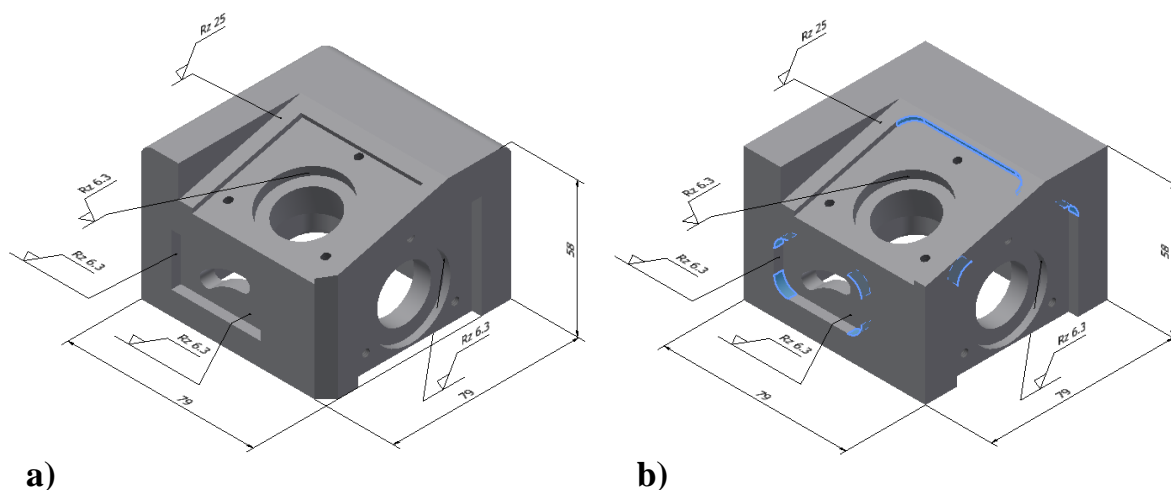


Figure 5. Gearbox Housing before (a) and after (b) the Evaluation of the Design by the MAS

For the application example of the gearbox housing, the review by the MAS revealed the following possibilities for improvement. The front chamfer and the rear rounded edges were removed, after consultation with the designer, as they were only inserted for aesthetic reasons. The missing radii were added to the pocket on the front left side. In addition, the tool outlets on the top and the right front side were also added. To ensure that the pockets could be produced with the same milling tool, the sizes of

the radii were matched. The MAS also detects that the rear hole in the upper pocket is placed too close to the pocket wall. The designer is then asked whether the pocket should be extended to the rear and the CAD model is adjusted accordingly. To optimize the machining time of the gearbox housing, the surfaces are either roughed or finished, depending on their surface specification. Figure 5 shows the gearbox housing before (Fig. 5a) and after (Fig. 5b) machining, in which the added surfaces are highlighted.

5. Discussion

The CommonKADS method provides a good structure for the development of MAS in product development and helps to divide tasks among different agents for checking the CAD models. Dividing the models into agents, communication, and knowledge ensures that all areas are considered at an early stage. Defining the form of communication and the division of tasks between agents, as well as the data structure for the knowledge to be stored, is critical to the robust functioning of the MAS. Once the specification and planning of the MAS has been completed, the MAS can be programmed and the required infrastructure implemented. The MAS framework SPADE was used for the implementation, as it can utilize numerous algorithms programmed in Python, especially the AI libraries, and the communication via the open messaging protocol XMPP offers the possibility to integrate further agents, humans, and even third-party tools.

In comparison to related literature, the focus was placed on the embodiment design phase, in which a design close to manufacturing is available, so that, for example, the radii for tool runouts are considered. In addition, an attempt was made to read as much information from the CAD model directly, so that only questions about function or context need to be asked by the user. It is also worth mentioning that the MAS presented here makes changes to the CAD model independently, whereas existing work only formulates suggested changes.

At this moment, the system can evaluate features such as holes, chamfers, fillets, pockets, slots, and bosses. However, the system does not yet recognize extrusions where several features are blended, so the system can only check parts that have been created completely from the raw part. Another difficulty is the evaluation of free-form surfaces since the traverse path of the tool or machine also has to be considered here. Furthermore, the MAS is not able to adapt the model in case of missing tools or to submit a proposal to extend the tool list. A priority list for tools would also be conceivable. To make the system even more efficient, one possibility would be to review multiple parts synchronously. About virtual design reviews, the existing system so far covers only a part, since on the one hand the specific product requirements have not yet been implemented and on the other hand only formalized expert knowledge is used for evaluation.

6. Conclusion and Further Research

In this paper, a MAS for the evaluation of milling designs was presented. Here, the CommonKADS method was used, and architecture was developed consisting of the CAD program Autodesk Inventor, the MAS framework SPADE, and a communication server. A gearbox housing served as an application example, which was checked against design guidelines, standards, and tool lists. Subsequently, the model was divided into machining steps and a possible tool was assigned to each step and checked about design errors, e.g., tool runouts, and the CAD model has adjusted accordingly. It could be shown that MAS can be a promising tool to represent expert knowledge decentrally in the form of agents and that it is easy to maintain and extend due to its modular structure.

The presented MAS offers a first step for the evaluation of CAD designs about virtual design reviews or the digitization of the 4-eyes principle. In addition, the system can be used to simplify and digitize the approval process by allowing the reviewer to use the Action-Item-List for review. Compared to manual checking, this can save time as the system automatically reads out the machining steps, checks the manufacturability by the tools, and automatically adjusts the CAD model. An extension of the system is easily possible and is simplified by the templates for the agents. Furthermore, a check of the design about the available machine tools is possible by analyzing the part with different stored tool lists, so that an automatic adaptation of the design in case of a short-term change to another machine

tool is possible without much effort. About the machine tools, it is also interesting whether the required form and position tolerances of the workpiece can be produced on these. Furthermore, a conceivable extension would be the use of an artificial neural network through which CAD models can be clustered (Müller *et al.*, 2020), e.g., whether the part is turned or milled, and accordingly, the agents are assigned for detailed evaluation.

For further research, the focus will be on representing virtual design reviews with MAS. For this purpose, requirements need to be operationalized so that they can be used by the system and to evaluate the product's fulfillment of these requirements. Furthermore, the system can be extended by using coordination, collaboration, and negotiation techniques to reach a common solution based on the different perspectives between the agents and their knowledge. Since these processes are very knowledge-intensive, there is an increasing search for ways to provide agents with learning capabilities, such as case-based reasoning, where agents draw on experience in a case base and apply this knowledge to a new problem. In addition, reasoning can be done in the presence of uncertainty by using Bayesian decision networks, where the knowledge representation is given by conditional probabilities.

References

- Bellifemine, F., Poggi, A. and Rimassa, G. (2001), "Developing Multi-agent Systems with JADE", in Castelfranchi, C. and Lespérance, Y. (Eds.), *Intelligent Agents VII Agent Theories Architectures and Languages*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 89–103. https://doi.org/10.1007/3-540-44631-1_7
- Bergenti, F., Caire, G., Monica, S. and Poggi, A. (2020), "The first twenty years of agent-based software development with JADE", *Autonomous Agents and Multi-Agent Systems*, Vol. 34 No. 2, p. 36. <https://doi.org/10.1007/s10458-020-09460-z>
- Brooks, R. (1987), "A hardware retargetable distributed layered architecture for mobile robot control", *Proceedings. 1987 IEEE International Conference on Robotics and Automation*, Vol. 4, Institute of Electrical and Electronics Engineers, pp. 106–110. <https://doi.org/10.1109/ROBOT.1987.1088016>
- Bussmann, S., Jennings, N.R. and Wooldridge, M. (2004), *Multiagent Systems for Manufacturing Control, Media*, Springer Berlin Heidelberg, Berlin, Heidelberg. <https://doi.org/10.1007/978-3-662-08872-2>.
- Chu, C.-H.H., Wu, P.-H.H. and Hsu, Y.-C.C. (2009), "Multi-agent collaborative 3D design with geometric model at different levels of detail", *Robotics and Computer-Integrated Manufacturing*, John Wiley & Sons, Vol. 25 No. 2, pp. 334–347. <https://doi.org/10.1016/j.rcim.2007.01.005>
- Dostatni, E., Diakun, J., Grajewski, D., Wichniarek, R. and Karwasz, A. (2016), "Multi-agent system to support decision-making process in design for recycling", *Soft Computing*, Springer, Vol. 20 No. 11, pp. 4347–4361. <https://doi.org/10.1007/s00500-016-2302-z>
- Eymann, T. (2003), "Grundlagen der Software-Agenten", *Digitale Geschäftsagenten: Softwareagenten Im Einsatz*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 17–107. https://doi.org/10.1007/978-3-642-55622-7_2
- Feng, S.C. (2005), "Preliminary design and manufacturing planning integration using web-based intelligent agents", *Journal of Intelligent Manufacturing*, Springer, Vol. 16 No. 4–5, pp. 423–437. <https://doi.org/10.1007/s10845-005-1655-4>
- Fougères, A.J. and Ostrosi, E. (2018), "Intelligent agents for feature modelling in computer aided design", *Journal of Computational Design and Engineering*, Vol. 5 No. 1, pp. 19–40. <https://doi.org/10.1016/j.jcde.2017.11.001>
- Gembarski, P.C. (2020a), "Three Ways of Integrating Computer-Aided Design and Knowledge-Based Engineering", *Proceedings of the Design Society: DESIGN Conference*, Vol. 1 No. 2017, pp. 1255–1264. <https://doi.org/10.1017/dsd.2020.313>
- Gembarski, P.C. (2020b), "On the Conception of a Multi-agent Analysis and Optimization Tool for Mechanical Engineering Parts", in Jezic, G., Chen-Burger, J., Kusek, M., Sperka, R., Howlett, R.J. and Jain, L.C. (Eds.), *Agents and Multi-Agent Systems: Technologies and Applications 2020*, Vol. 186, John Wiley & sons, Singapore, pp. 93–102. https://doi.org/10.1007/978-981-15-5764-4_9
- Hirz, M., Wilhelm, D., Anton, G. and Johann, L. (2013), *Integrated Computer-Aided Design in Automotive Development, Integrated Computer-Aided Design in Automotive Development*, Vol. 10, Springer Berlin Heidelberg, Berlin, Heidelberg. <https://doi.org/10.1007/978-3-642-11940-8>.
- De Hoog, R., Martil, R. and Wielinga, B. (1994), "The Common KADS model set", No. February 1998.

- Huet, G., Culley, S.J., McMahon, C.A. and Fortin, C. (2007), “Making sense of engineering design review activities”, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, Vol. 21 No. 3, pp. 243–266. <https://doi.org/10.1017/S0890060407000261>
- Iglesias, C.A., Garijo, M., González, J.C. and Velasco, J.R. (1996), “A methodological proposal for multiagent systems development extending CommonKADS”, *Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop*, Vol. 1, pp. 21–25.
- Jia, H.Z.Z., Ong, S.K.K., Fuh, J.Y.H.Y.H., Zhang, Y.F.F. and Nee, A.Y.C.Y.C. (2004), “An adaptive and upgradable agent-based system for coordinated product development and manufacture”, *Robotics and Computer-Integrated Manufacturing*, Vol. 20 No. 2, pp. 79–90. <https://doi.org/10.1016/j.rcim.2003.08.001>
- Kingston, J. (2001), “Modelling Agents and Communication using CommonKADS”, *Research and Development in Intelligent Systems XVII*, Springer London, pp. 301–319. https://doi.org/10.1007/978-1-4471-0269-4_22
- Kratzer, M., Rauscher, M., Binz, H. and Goehner, P. (2011), “An agent-based system for supporting design engineers in the embodiment design phase”, *ICED 11 - 18th International Conference on Engineering Design - Impacting Society Through Engineering Design*, John Wiley & Sons, Vol. 10 No. PART 2, pp. 178–189.
- Li, Y. (2007), “Application of Multi-Agent for Collaborative Product Design Engineering”, Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD 2007), Vol. 2, John Wiley & Sons, pp. 450–454. <https://doi.org/10.1109/SNPD.2007.37>
- Liu, H., Tang, M. and Frazer, J.H. (2004), “Supporting dynamic management in a multi-agent collaborative design system”, *Advances in Engineering Software*, Vol. 35 No. 8–9, pp. 493–502. <https://doi.org/10.1016/j.advengsoft.2004.06.007>
- Mahesh, M., Ong, S.K. and Nee, A.Y.C. (2007), “A web-based multi-agent system for distributed digital manufacturing”, *International Journal of Computer Integrated Manufacturing*, Vol. 20 No. 1, pp. 11–27. <https://doi.org/10.1080/09511920600710927>
- Medani, O. and Ratchev, S.M. (2006), “A STEP AP224 agent-based early manufacturability assessment environment using XML”, *The International Journal of Advanced Manufacturing Technology*, Springer, Vol. 27 No. 9–10, pp. 854–864. <https://doi.org/10.1007/s00170-004-2279-0>
- Milton, N. (2008), Knowledge Technologies.
- Müller, P., Gembariski, P.C. and Lachmayer, R. (2020), “Detektion von Konstruktionsfehlern durch eine automatisierte Objekterkennung mittels Deep Learning”, *Proceedings of the 18th Joint Colloquium on Design Engineering, Duisburg*. <https://doi.org/10.15488/11539>
- Palanca, J., Terrasa, A.A., Julian, V. and Carrascosa, C. (2020), “Spade 3: Supporting the new generation of multi-agent systems”, *IEEE Access*, Vol. 8, pp. 182537–182549. <https://doi.org/10.1109/ACCESS.2020.3027357>
- Plappert, S., Gembariski, P.C. and Lachmayer, R. (2020), “The Use of Knowledge-Based Engineering Systems and Artificial Intelligence in Product Development: A Snapshot”, in Świątek, J., Borzemski, L. and Wilimowska, Z. (Eds.), *Advances in Intelligent Systems and Computing*, Vol. 1051, Springer International Publishing, Cham, pp. 62–73. https://doi.org/10.1007/978-3-030-30604-5_6
- Plappert, S., Gembariski, P.C. and Lachmayer, R. (2022), “Knowledge-Based Design Evaluation of Rotational CAD-Models with a Multi-Agent System”, in Borzemski, L., Selvaraj, H. and Świątek, J. (Eds.), *Advances in Systems Engineering*, Springer International Publishing, Cham, pp. 47–56. https://doi.org/10.1007/978-3-030-92604-5_5
- Plappert, S., Gembariski, P.C.P.C. and Lachmayer, R. (2021), “Multi-Agent Systems in Mechanical Engineering: A Review”, in Jezic, G., Chen-Burger, J., Kusek, M., Sperka, R., Howlett, R.J. and Jain, L.C. (Eds.), *Agents and Multi-Agent Systems: Technologies and Applications 2021*, Vol. 241, Springer Singapore, Singapore, pp. 193–203. https://doi.org/10.1007/978-981-16-2994-5_16
- Verhagen, W.J.C., Bermell-Garcia, P., van Dijk, R.E.C. and Curran, R. (2012), “A critical review of Knowledge-Based Engineering: An identification of research challenges”, *Advanced Engineering Informatics*, Elsevier Ltd, Vol. 26 No. 1, pp. 5–15. <https://doi.org/10.1016/j.aei.2011.06.004>
- Weiss, G. (2000), *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, MIT Press.
- Wetmore, W.R., Summers, J.D. and Greenstein, J.S. (2010), “Experimental study of influence of group familiarity and information sharing on design review effectiveness”, *Journal of Engineering Design*, Taylor & Francis, Vol. 21 No. 1, pp. 111–126. <https://doi.org/10.1080/09544820802238217>
- Weyns, D. (2010), *Architecture-Based Design of Multi-Agent Systems*, Architecture-Based Design of Multi-Agent Systems, Springer Berlin Heidelberg. <https://doi.org/10.1007/978-3-642-01064-4>.
- Wooldridge, M. (2009), *An Introduction to Multiagent Systems*, John Wiley & Sons.