Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 134 (2025) 109–114

www.elsevier.com/locate/procedia

Proceedings of the 58th CIRP Conference on Manufacturing Systems 2025

A Kanban-based Approach to Manage Machine Learning Projects in Manufacturing

Ulf Schreier*,[a], Peter Reimann*,[b,d], Bernhard Mitschang[c]

[a]*Business Information Systems - Furtwangen University of Applied Science, Robert-Gerwig-Platz 1, D-78120 Furtwangen, Germany*
[b]*Graduate School of advanced Manufacturing Engineering (GSaME), University of Stuttgart, Nobelstr. 12, D-70569 Stuttgart, Germany*
[c]*Institute of Parallel and Distributed Systems (IPVS), University of Stuttgart, Universitätsstr. 38, D-70569 Stuttgart, Germany*
[d]*Institute for Program Structures and Data Organization (IPD), Karlsruhe Institute of Technology (KIT), Am Fasanengarten 5, D-76131 Karlsruhe, Germany*

* Corresponding authors. *E-mail addresses:* ulf.schreier@hs-furtwangen.de; peter.reimann@gsame.uni-stuttgart.de

Abstract

A growing number of machine learning (ML) projects in manufacturing require the collaboration of various experts. In addition to data scientists, stakeholders with production engineering knowledge have to specify and prioritize individual project tasks. Data engineers prepare input data, while machine learning operations (MLOps) engineers ensure that trained models are deployed and monitored within IT landscapes. Existing project management approaches, e.g., Scrum, have problems for ML projects, as they do not consider various expert roles or ML project stages. We propose a project management approach defining a Kanban workflow by readjusting stages of ML development lifecycles, e.g., CRISP DM. This makes it possible to map expert roles to stages of the Kanban workflow. An adapted Kanban board allows visualizing and reviewing the status of all project tasks. We validate our approach with specific use cases, showing that it facilitates ML project management in manufacturing.

Keywords: Machine learning (ML); ML project management, machine learning operations (MLOps); Kanban; Scrum

1. Introduction

Due to advances in data availability and compute infrastructures, recent literature discusses a growing number of machine learning (ML) projects in manufacturing [1, 2, 3]. A common form is supervised learning, where ML models are trained with historical data and used as a function to predict a value for new input. The outcomes of these projects are not only experimental, but are used as business applications or as part of these applications, e.g., for product design, manufacturing and maintenance of goods. In practice, many specialized ML models can be used in an enterprise, which leads to concurrent development of new and revision of existing models [4, 5].

The successful execution of ML projects requires the collaboration of various experts for certain kinds of specialized tasks. The core work is the development of ML models and their careful evaluation. In many rounds of experiments, data scientists have to transform data to features, find the best ML algorithm, and tune its hyperparameters. As a result, data scientists deliver a prototypical piece of software, e.g., written in Python or R.

However, this piece of software is often only a prototype which cannot be deployed into an operational IT infrastructure. Therefore, deployment engineers have to refine the software prototype of the experimental phase to an efficient, maintainable, error-free software product. Finally, support for monitoring and change of ML models is essential, since ML models may become outdated, e.g., due to concept shifts [1]. This whole area is known as machine learning operations (MLOps) and its major tasks are carried out by MLOps engineers [4].

Moreover, the data needed for ML projects is often not readily available in an enterprise. Typically, it is stored in databases hidden behind software applications. For manufacturing, data from several ISA-95 levels [6], e.g., planning, operations, monitoring and field, with many systems involved, have to be considered. Hence, data scientists also have to collaborate with data engineers to get needed datasets and to prepare them for ML.

All in all, the development and deployment of ML models in a software infrastructure of an enterprise is more than only data science. It is a mixture of complex and diverse activities requiring teamwork and special knowledge in several fields. All these

10.1016/j.procir.2025.03.011

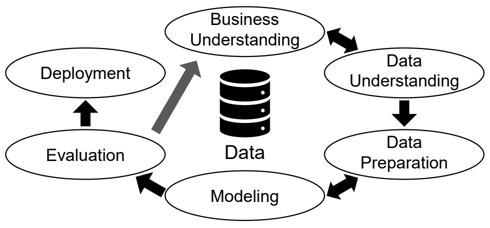

Fig. 1. Cross-Industry Standard Process for Data Mining, cf. [7].

activities include the development of software. The complexity of the activities and the diversity of involved expert roles and their knowledge calls for an adequate approach to software project management that is tailored to ML projects.

CRISP-DM [7] is the pioneering process framework to organize data mining projects introducing six project phases (see Figure 1). However, it assumes a classic, non-agile project management. Moreover, CRISP-DM does not describe any expert roles and how to distribute work among them. Scrum [8] is a favorite choice for an agile project management framework in software development. But although Scrum is successfully used for many software projects in general, some severe problems arise when applied for ML projects.

First, ML projects have only a few generic user stories that correspond to common stages of ML processes [7], e.g., 1) provide data, 2) prepare features and train a model, 3) use this model to predict values and monitor its prediction quality. The only aspects that differ between these similar user stories are so-called acceptance criteria, i.e., conditions under which a user story is successfully implemented. Scrum, however, does not exploit this specific structure of user stories, but assumes each story to be a unique software function.

Second, before starting a specific task in ML projects, it is hard to estimate the time needed to finish this task [9, 10]. For instance, the data quality cannot be assessed in advance, so that it is not clear how much effort will be needed for data cleaning, or model training might need to iteratively adjust model parameters. Scrum however expects project tasks to be small, i.e., executable in short so-called sprints with a fixed time length, which requires an easy prediction of the effort.

Third, multiple expert roles are involved in ML projects. Data engineers, data scientists, MLOps engineers and software engineers are specialized experts with diverse educational backgrounds, knowledge and management cultures. In Scrum, it is left to the team how to organize itself. In particular, Scrum does neither define roles of team members nor any project workflow.

Finally, a ML model is a monolith, i.e., learned approximation of a function implemented as a software component with only one service for making a prediction. Scrum however assumes that a software application can be decomposed into many independent and diverse functions, i.e., user stories on which a team can work concurrently and incrementally.

In this paper, we propose a project management approach that uses Kanban [11] – another agile method originating from the manufacturing area — to master these problems. Nevertheless, Kanban is a general analysis method and we still have to find, discuss and assess a specific solution setup for reasonable project management. To this end, we define a Kanban workflow by readjusting stages of mature ML development lifecycles. This makes it possible to map expert roles to stages of the Kanban workflow. An adapted Kanban board allows visualizing the status of all project tasks. We validate our approach by illustrating how to apply it to specific manufacturing use cases and how this facilitates ML project management in manufacturing.

In the following, we discuss related work in Section 2, before we propose our approach to Kanban-based project management in Section 3. A validation via a meaningful use case scenario follows in Section 4, while Section 5 discusses the outcomes of this validation and concludes this paper.

2. Related work

In Scrum [8], a product owner defines and prioritizes requirements and lists them in a product backlog. For software development, each backlog item contains a user story and its acceptance criteria with testable conditions the software has to satisfy [12]. A team of developers works in a self-determined way. A Scrum project is structured into iterations, i.e., sprints, having the same fixed time length of a few weeks. Scrum is widely accepted for general software development, but it does not fit well for ML projects, as described in Section 1.

Kanban is a method originally developed for process control in manufacturing, but which has been adapted for software development projects [11]. While Scrum highlights the team working together, Kanban emphasizes the workflow activities that need to be done. This organization schema is especially helpful for associating specific expert roles to individual workflow steps. Kanban also relies on buffers between activities. In software development, the buffer contents correspond to virtual artifacts, e.g, software code and documentation. Making buffers visible with a Kanban board is at the focal point of the method. To reduce costs and to improve lead time, Kanban introduces work-in-progress (WIP) limits on buffers and the pulling principle, i.e., the experts can move forward only a limited number of tasks to the next buffer [11]. Kanban is a generic method, but it does not define a concrete development workflow.

Scrumban [13] augments Scrum via some ideas of Kanban, e.g., it uses a Kanban board and WIP limits. However, Scrumban still relies on most of the Scrum rules, e.g., it does not define a workflow with associated expert roles, and it still relies on sprint iterations with a fixed duration.

CRISP-ML(Q) [14] extends CRISP-DM with quality checking methods for all phases and refines them for machine learning. QM-CRISP-DM [15] specializes the process model for manufacturing by defining which of the six sigma methods may be applied at each phase. These are valuable contributions, but they do not focus on project organization.

Based on interviews with experts, Kreuzberger et al. [4] propose a more detailed workflow and associated expert roles for ML projects. A special emphasis is on ML operations (MLOps) and on repositories and tools assisting ML and data scientists. This is a good foundation for teamwork considerations, but the authors do not address questions of project organization.

Saltz et al. [9] compare data science project management methods via a controlled experiment, where student groups worked on the same ML use case with several project methodologies. A major outcome is that Scrum did not work well, mainly due to the reasons mentioned in Section 1. The groups with Kanban and CRISP-DM had better results. The Kanban group used workflow steps for data preparation, data analysis and model deployment. However, the workflow employed by this group did not include all details of how ML projects are carried out in real-world. For instance, the students start with pre-defined requirements and with data files that have already been extracted from data sources. Moreover, deployment is reduced to communicating results, i.e., they did not deploy a ML model into an IT infrastructure. Expert roles for requirements, data or MLOps engineering are hence not defined.

In another work [10], the authors use a literature research and a field study with a large enterprise to discuss challenges of Scrum and Kanban for Big Data projects. The main critical points for Scrum are that reacting to issues during operation of ML models, time estimation of user storys, and frequent project meetings constitute problems for ML projects. Kanban was also used by teams, but only with a simple Scrum-like board or with WIP limits. A positive effect of using Kanban was that time estimation was no longer necessary. However, the authors only rudimentarily describe how team members need to collaborate, and they do not define any specific expert roles.

A later paper of this group provides a detailed definition of an agile project management approach for data science, which is called Data Driven Scrum (DDS) [16]. It is based on Scrum and changes some rules which do not fit to data science projects. For instance, iterations no longer have a fixed time length. The DDS method also applies some Kanban principles. It uses a task board and WIP limits, but does not specify the columns of the board. Likewise, DDS does not define how team members with diverse expert roles need to work together in ML projects.

3. Kanban-based organization of ML project management

ML project members have to carry out diverse activities in sequence, which may correspond roughly to the phases of process frameworks such as CRISP-DM. To this end, the group of ML project members needs to include diverse experts with specialized knowledge in either data engineering, data science, or MLOps engineering. On this account, it makes sense to apply Kanban to find the best workflow organization. We therefore propose a Kanban workflow as basis for ML project organization and discuss the design rationale, such as which expert roles are associated with each workflow step. This is extended by a Kanban board to visualize the status of project tasks, and we discuss the influence of WIP limits and input cadence.

3.1. Workflow definition

Depending on project conditions, many different workflow structures may be useful. Therefore, we restrict the considered field of ML projects via the following five assumptions. First,

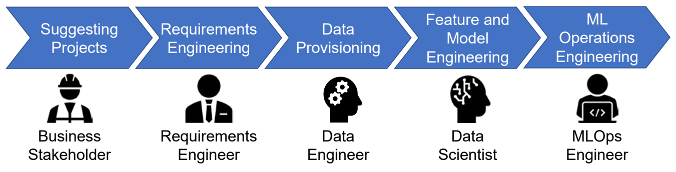

Fig. 2. Main steps and associated expert roles of the workflow used in the Kanban-based approach to ML project management.

we assume that ML is applied in a medium- to large-sized company with a typical software landscape of enterprise applications and IT infrastructures. In these cases, data is accessed using software applications, many data sources exist, and the data have complex and heterogeneous data structures. Moreover, ownership of data plays a role, so that not everyone can access all data. Presumably, data platforms, such as data warehouses or data lakes, are available, and data engineers are already operating in this environment. Second, ML solutions have to be embedded into the software and IT infrastructure. Note that we limit our approach to the integration of ML models, i.e., we do not consider client software applications invoking ML models. These clients may be developed via a standard software development lifecycle. Third, many applications of well-established ML algorithms, e.g., for clustering, classification, or regression, are possible within a considered enterprise. Consequently, the organization has established a group of data scientists in order to exploit ML. Fourth, MLOps engineers have an understanding of data provision and can develop and integrate software to prepare the input used by ML models to make predictions. Obviously, MLOps engineers also need to have an understanding of the data science solution. Fifth, all experts involved have at least medium skills in software design, so that an additional software engineer is not required.

Figure 2 shows our workflow definition and assigns expert roles to each individual workflow step. In step 1, problems have to be identified and a basic idea how ML can help to solve these problems is present. This is done by a business stakeholder, who can be anyone with domain understanding, e.g., an industrial engineer. This business stakeholder also prioritizes the problems and corresponding projects to solve them, i.e., s/he suggests which of these projects the other experts have to carry out in the following workflow steps.

The start of requirements engineering as second step already follows the pull principles of Kanban. When a requirement engineer is available for a new project, s/he will pick the project with highest priority and defines the detailed acceptance criteria for preparing data, features, models, and inputs for predictions.

Step 3 comprises tasks for data provisioning, e.g., to extract data from the data sources and to improve data quality. In our assumed socio-technical setup of an enterprise IT landscape, this is not simple, but requires data engineers as experts who have sufficient understanding of database systems and data engineering. Furthermore, these data engineers need to be able to negotiate with IT platform specialists and with data source owners in order to obtain access rights to the data.

Suggesting Projects	Requirements Engineering		Data Provisioning		Feature and Model Engineering		MLOps Engineering	
To do	Doing	Done	Doing	Done	Doing	Done	Doing	Done
P7	P6		P5	P4	P2		P1	
P8					P3			
P9								

Fig. 3. Abstract example of a Kanban board in our approach. Individual projects are indicated via their project numbers and associated to columns of the board.

Once the data is prepared and made available, data scientists can start to work on the core activity of feature and model engineering (step 4). This is an exploratory task [5], where data scientists have to try out several ML algorithms and associated hyperparameters to train and test several candidates of ML models. The task is finished when an ML model is found that satisfies all previously defined acceptance criteria. A Definition of Done for this activity may also contain reviews by requirements engineers or business stakeholders, who may assert that the final ML model satisfies all relevant business requirements. The result contains data used for learning and data used for testing model quality, source code, and the actual ML model.

Finally, the selected ML model needs to be deployed as an IT service into the IT infrastructure of the enterprise or of its manufacturing environment. This last step of our workflow needs special care, since enterprises usually have complex IT infrastructures. For instance, they may use a Kubernetes environment to support elastic scaling on a cloud platform. This workflow step comprises not only model deployment on an IT server, but also data preprocessing for model use. That is, adequate data and feature engineering is not only needed for batch-oriented learning in steps 3 and 4, but also for new input data used for actual predictions in step 5. Depending on the use case, the batch software used in the previous steps can be reused for step 5, or it needs to be adapted to special needs, e.g., to handle data streams [4]. In addition, this step comprises the development of a solution for monitoring model quality, e.g., to be able to adapt ML models to data set shifts [1, 5]. Here, MLOps engineers are required having the necessary competences of model deployment and monitoring and to adapt data provision to prepare input data used for model predictions.

3.2. Kanban board, WIP limits and cadence

Based on our workflow definition, we design a Kanban board with the columns shown in Figure 3. In this Kanban board, the first step of our workflow for suggesting projects gets a column *to do*, while each of the four subsequent steps is associated with a column *doing* and a column *done*. In a practical setting, all user stories are shown as cards and moved between the columns indicating the status of the tasks. Figure 3 shows an exemplified multi-project situation, where cards are only indicated as project numbers P1, P2, etc. New projects are inserted as new cards into the *to do* column of the first step, e.g., as shown for projects P7, P8 and P9 in Figure 3. The order of the cards indicates the priority assigned to them by the business stakeholder, i.e., project P7 has a higher priority than project P8. When a requirements engineer is ready to work, s/he picks the card with the highest priority from this column *to do* and moves it to the *doing* column of the second workflow step. Figure 3 shows that a requirements engineer is currently defining acceptance criteria for project P6. When s/he finishes work, s/he moves this card to the *done* column of this workflow step.

In general, a *done* column of the three steps for requirements engineering, data provisioning, or feature and model engineering implicitly corresponds to the list of tasks to be worked on in the respective next step. For instance, data engineers may pick cards from the *done* column of requirements engineering and move them to the *doing* column of data provisioning as soon as they have time to prepare the data. Work continues in the same way until it reaches the done column of MLOps engineering. Cards in this column indicate that ML models for the projects are already deployed and ready to make predictions.

Further key decisions in a Kanban project are limits for work-in-progress (WIP). The concrete limit size usually depends on the number of experts available and on the difficulty of the ML problem at hand. The principle of WIP limits is very important for ML projects as a substitute for planning. This especially holds for the step of feature and model engineering [4]. It can happen that the number of cards in the buffer of data scientists, i.e., in the *done* column of data provisioning, increases strongly. In this case, the Kanban pull strategy and WIP limits ensure that no new data provisioning tasks and in consequence requirements engineering tasks will be started. When this overflow scenario happens, the experts at earlier stages may look for other useful work, e.g., for improving the IT infrastructure.

The input cadence of Kanban projects depends mainly on the need for meetings of the experts involved and on the cost of meetings at project start time. If for instance detailed planning is useful and experts with tight schedules have to meet, it makes sense to come together with low frequency and to release a large bunch of tasks in each meeting. However, it is difficult to plan an ML project in detail. Hence, meetings may be used only to focus on prioritizing ML projects instead of planning them because Kanban already provides time control via WIP limits. Thus, a high cadence may be achieved by only suggesting a new ML project without immediately working on its details.

4. Validation via a use case scenario in manufacturing

The use case scenario to validate our approach is based on a generic production process of a manufacturing company. We derived this scenario from several real-world use cases for ML-based quality control, fault detection and fault diagnosis that we explored in previous publications [1, 17, 18, 19] or that are described by other authors [2, 3, 20]. As shown in Figure 4, the general production process consists of multiple steps, each of which is carried out either automatically by machines or by human workers using manufacturing tools. Between some of the process steps, inline quality gates are used for early product testing [3, 21]. The process ends with an End-of-Line (EoL) test bench, which carries out a more detailed product test [2, 19].

The individual process steps, quality gates, and the EoL test bench generate various data. This includes MES data as well as sensor data from quality gates at the manufacturing operations

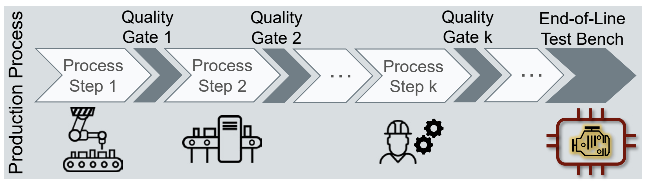

Fig. 4. Use case scenario: A manufacturing process with multiple process steps, quality gates and an End-of-Line test bench that may each be supported by ML approaches to enhance quality control, fault detection and fault diagnosis.

management level of ISA-95. Other relevant data are sensor data from machines at the ISA-95 field level, data from SCADA systems at the monitoring level, as well as textual data documented by workers [1, 3]. The availability of various data motivates the manufacturing company to launch projects for implementing ML-based solutions to enhance quality control, fault detection and fault diagnosis in the production process.

As discussed in Section 3, we assume a medium- to large-sized company that employs several experts of the roles required by our Kanban-based approach to ML project management (see Figure 2), i.e., requirements engineers, data engineers, data scientists, and MLOps engineers. In addition, production quality engineers are available having in-depth knowledge of quality control, fault detection and fault diagnosis. These quality engineers correspond to the business stakeholders of our Kanban-based approach. In our scenario, they first suggest several projects to introduce ML into the production process. Furthermore, they prioritize these projects according to the following list, i.e., the first projects have higher priority:

1. *Fault detection at EoL testing (FDet-EoL):* The first project aims to realize a ML solution to support fault detection in the EoL test bench [17, 18]. The quality engineers give this project the highest priority due to the high business need to only deliver correctly working products.
2. *Fault detection at inline quality gates (FDet-QG)*: The project with the second highest priority also involves ML-based fault detection, but earlier in the production process at inline quality gates [1, 17, 20]. In our use case, the quality engineers give this project a slightly lower priority, as it is a more detailed view than fault detection in EoL testing.
3. *Anomaly detection at process steps (ADet-PS)*: ML may also be used to detect process anomalies, i.e., abnormal process conditions within individual steps of a production process [3, 22]. These abnormal process conditions may be an early indicator of a poor final product quality.
4. *Fault diagnosis at EoL testing (FDia-EoL)*: ML-based fault diagnosis at the EoL test bench [2, 18, 19] constitutes a more sophisticated project, which is thus given the lowest priority. The goal is to identify the cause of a previously detected fault or quality problem.

Our scenario focuses on only four projects, but note that this number increases quickly in practice, because we define project types which may be carried out on each product, on each production process, or even on each quality gate and process step.

Suggesting Projects	Requirements Engineering		Data Provisioning		Feature and Model Engineering		MLOps Engineering	
To do	Doing	Done	Doing	Done	Doing	Done	Doing	Done
ADet-PS *FDia-EoL*	*FDet-QG*		*FDet-EoL*					

Fig. 5. Kanban board for some of the projects of the use case scenario.

Suggesting Projects	Requirements Engineering		Data Provisioning		Feature and Model Engineering		MLOps Engineering	
To do	Doing	Done	Doing	Done	Doing	Done	Doing	Done
	FDia-EoL		*ADet-PS*		*FDet-QG*		*FDet-EoL*	

Fig. 6. Kanban board for the projects of the use case scenario after the projects have already passed some of the workflow steps.

Initially, the quality engineers place all four projects in the *to do* column 'Suggesting Projects' of our Kanban board, in the order given by their priority. Afterwards, requirements engineers pick the first project *FDet-EoL* from this column in order to define acceptance criteria for it. As soon as they are finished, they move the project to their *done* column and pick the next project from the column 'Suggesting Projects'. Data engineers may then pick the first project *FDet-EoL* from the *done* column of requirements engineering in order to start preparing and providing sensor data of the EoL test bench. This situation is illustrated by the Kanban board in Figure 5. When this data provisioning step is completed, the project is similarly moved to the columns of the subsequent workflow steps for feature and model engineering and then for MLOps Engineering.

After a while, the Kanban board looks like the one shown in Figure 6. Here, the data for the first project *FDet-EoL* has already been prepared, and an accurate classification model has been trained based on this data, i.e., all workflow steps up to feature and model engineering have already been completed. MLOps engineers are currently preparing this classification model to deploy it into the IT infrastructure of the EoL testing area. Thereby, they are mainly implementing an IT service that uses the classification model to predict whether new products that pass the test bench are either functional or defective. All other projects are currently undergoing one of the other workflow steps, i.e., project *FDet-QG* is in the *doing* column for feature and model engineering, project *ADet-PS* for data provisioning, and project *FDia-EoL* for requirements engineering. These projects are passed through the subsequent steps as soon as the respective work in the current steps has been completed.

5. Discussion, Lessons Learned and Conclusion

The major benefit of our Kanban-based approach is the structured workflow. This leads to efficiency improvements and a reduced likelihood of errors mainly due to the following advantages: (a) The workflow makes it possible to explicitly map each step to specific expert roles, whose knowledge match the skill set required for each individual project task. (b) The workflow can be visualized by Kanban boards providing high transparency of project status for all stakeholders. (c) Each workflow step has its own Definition of Done, so that the experts in individual steps know explicitly what they need to do to fulfill the acceptance criteria. (d) Reaching a WIP limit of a process step is an indication that this step needs far more time than usual and is a bottleneck. In such a case, experts at previous workflow steps get a signal to look for work not increasing the bottleneck. This way, WIP limits reduce average lead time of projects and waste on work which will not be finished soon [11].

Our experience shows that ML projects face challenges for accessing and preparing manufacturing data. This is evident in particular because the considered use cases are based on the analysis of multiple heterogeneous data sources, e.g., data from MES, sensor data from machines or from test benches [1], databases with historical test and diagnosis data [19], or even unstructured text data [3]. This increases the effort of data engineers. Here, implementation of ISA-95 interface standards [6] may support unified solutions to access and integrate data. In addition, it is important to incorporate domain knowledge into data integration and preparation, especially to cope with challenges associated with complex data characteristics [19]. This is also facilitated by our approach, as business stakeholders, e.g., production or quality engineers, and requirements engineers are involved right from the start in the first two steps of our workflow and can immediately contribute their domain knowledge.

Another observation is that feedback loops are needed due to the exploratory nature of ML projects. Kanban in general and our approach supports this practice. It is possible that an expert at a later workflow step discovers that a result of a previous step has to be changed. Minor changes can be handled directly by data scientists or MLOps engineers, e.g., by adding data transformations, without looping back to a previous step. Major changes require that projects have to move back to earlier steps. For this purpose, Kanban boards have to permit the backward movement of cards, which is supported by the Kanban method and by state-of-the-art tools. For instance, if requirements have to be changed, a card may move back to the first board column or if new data provisioning is needed, the card may move to the Done column of requirements engineering. Here, it is useful to have an exception rule to increment WIP limits temporarily by one, since the buffers could be full already.

Altogether, our Kanban approach addresses some of the disadvantages of standard attempts such as Scrum to facilitate the organization of ML projects, as needed for complex industry IT landscapes. As part of future work, we plan to extend our approach for other scenarios, e.g, for smaller enterprises, and to further validate it via more concrete case studies.

Acknowledgements. This work was partially supported by the Ministry of Science, Research and Arts of the State of Baden-Wurttemberg within the sustainability support of the Excellence Initiative II.

References

[1] Y. Wilhelm, U. Schreier, P. Reimann, B. Mitschang, H. Ziekow, Data Science Approaches to Quality Control in Manufacturing: A Review of Problems, Challenges and Architecture, in: Proc. of the 14th Symposium on Service-Oriented Computing (SummerSOC), 2020, pp. 45–65.

[2] L. Leitner, A. Lagrange, C. Endisch, End-of-line Fault Detection for Combustion Engines Using One-Class Classification, in: Proc. of the IEEE International Conference on Advanced Intelligent Mechatronics (AIM), Banff, AB, Canada, 2016, pp. 207–213.

[3] C. Gröger, Industrial Analytics — An Overview, it – Information Technology 64 (1–2) (2022) 55–65.

[4] D. Kreuzberger, N. Kühl, S. Hirschl, Machine Learning Operations (MLOps): Overview, Definition, and Architecture, IEEE Access 11 (2023) 31866–31879.

[5] C. Weber, P. Hirmer, P. Reimann, H. Schwarz, A New Process Model for the Comprehensive Management of Machine Learning Models, in: Proc. of the 21st International Conference on Enterprise Information Systems, SCITEPRESS, Heraklion, Greece, 2019, pp. 415–422.

[6] B. Scholten, The Road to Integration: A Guide to Applying the ISA-95 Standard in Manufacturing, ISA, 2007.

[7] P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, R. Wirth, CRISP-DM 1.0: Step-by-step Data Mining Guide (2000).

[8] K. Schwaber, J. Sutherland, The Scrum Guide, scrumguides.org, 2020.

[9] J. S. Saltz, I. Shamshurin, K. Crowston, Comparing Data Science Project Management Methodologies via a Controlled Experiment, in: Proc. 50th Hawaii International Conf. on System Sciences, 2017, pp. 1014–1022.

[10] J. S. Saltz, I. Shamshurin, Achieving Agile Big Data Science: The Evolution of a Team's Agile Process Methodology, in: Proc. of the 2019 IEEE International Conference on Big Data, 2019, pp. 3477–3485.

[11] D. J. Anderson, Kanban: Successful Evoluationary Change for Your Technology Business, Blue Hole Press, 2010.

[12] K. S. Rubin, Essential Scrum: A Practical Guide to the Most Popular Agile Process, Addison-Wesley, 2012.

[13] C. Ladas, What is Scrumban?, www.agilealliance.org/scrumban, 2021.

[14] S. Studer, T. B. Bui, C. Drescher, A. Hanuschkin, L. Winkler, S. Peters, K.-R. Müller, Towards CRISP-ML(Q): A Machine Learning Process Model with Quality Assurance Methodology, Machine Learning and Knowledge Extraction 3 (2) (2021) 392 – 413.

[15] F. Schäfer, C. Zeiselmair, J. Becker, H. Otten, Synthesizing CRISP-DM and Quality Management: A Data Mining Approach for Production Processes, in: Proc. of the 2018 IEEE International Conference on Technology Management, Operations, and Decisions (ICTMOD), 2018, pp. 190–195.

[16] J. S. Saltz, N. Hotz, A. Sutherland, Achieving Lean Data Science Agility Via Data Driven Scrum, in: Proc. of the 55th Hawaii International Conference on System Sciences, 2022, pp. 7287–7296.

[17] A. Gerling, O. Kamper, C. Seiffer, H. Ziekow, U. Schreier, A. Hess, D. Ould Abdeslam, Results from using an AutoML Tool for Error Analysis in Manufacturing, in: Proc. of the 22nd International Conference on Enterprise Information Systems, 2022, pp. 100–111.

[18] V. Hirsch, P. Reimann, O. Kirn, B. Mitschang, Analytical Approach to Support Fault Diagnosis and Quality Control in End-Of-Line Testing, Procedia CIRP 72 (2018) 1333–1338.

[19] V. Hirsch, P. Reimann, D. Treder-Tschechlov, H. Schwarz, B. Mitschang, Exploiting Domain Knowledge to Address Class Imbalance and a Heterogeneous Feature Space in Multi-Class Classification, International Journal on Very Large Data Bases (VLDB-Journal) 32 (5) (2023) 1037–1064.

[20] S. Thalmann, H. G. Gursch, J. Suschnigg, M. Gashi, H. Ennsbrunner, A. K. Fuchs, T. Schreck, B. Mutlu, J. Mangler, G. Kappl, C. Huemer, S. Lindstaedt, Cognitive Decision Support for Industrial Product Life Cycles: A Position Paper, in: Proc. of the 11th International Conference on Advanced Cognitive Technologies and Applications (COGNITIVE), IARIA, Venice, Italy, 2019, pp. 3–9.

[21] A. Gerling, U. Schreier, A. Hess, A. Saleh, H. Ziekow, D. Ould Abdeslam, A Reference Process Model For Machine Learning Aided Production Quality Management, in: Proc. of the 22nd International Conference on Enterprise Information Systems, Prague, Czech Republic, 2020, pp. 515–523.

[22] Y. Wilhelm, P. Reimann, W. Gauchel, B. Mitschang, Overview on Hybrid Approaches to Fault Detection and Diagnosis: Combining Data-driven, Physics-based and Knowledge-based Models, Procedia CIRP 99 (2021) 278–283.