Complex Systems Informatics and Modeling Quarterly (CSIMQ)
eISSN: 2255-9922
Published online by RTU Press, https://csimq-journals.rtu.lv
Article 208, Issue 38, March/April 2024, Pages 27–53
https://doi.org/10.7250/csimq.2024-38.02

Enable Flexibilisation in FAIRWork's Democratic AI-based Decision Support System by Applying Conceptual Models Using ADOxx

Robert Woitsch[1], Christian Muck[2⋆], Wilfrid Utz[2], and Herwig Zeiner[3]

[1]BOC Products and Services AG, Operngasse 20b, 1040 Wien, Austria
[2]OMiLAB gGmbH, Lützowufer 1, 10785 Berlin, Germany
[3]JOANNEUM RESEARCH Forschungsgesellschaft mbH, Steyrergasse 17, 8010 Graz, Austria

robert.woitsch@boc-group.com, christian.muck@omilab.org, wilfrid.utz@omilab.org, herwig.zeiner@joanneum.at

Abstract. Decision-making in complex production environments is challenging as the information and knowledge requirements must be constantly observed since the ecosystems they operate in are continuously changing. Artificial intelligence (AI) can tackle complexity in decision-making by making machines more intelligent. But reacting to changing or new problems and related decision processes to facilitate the understanding of the involved humans is an equally important problem. Therefore, decision support systems are required to assist complex decisions and enable flexibility to support the decision-makers. Within this scope, we will introduce the *Democratic AI-based Decision Support System (DAI-DSS)*, which is designed and implemented within the EU-funded *FAIRWork* project, considering both human and machine actors during decision-making. The *FAIRWork* project proposes a model-based approach to both express high-level decision scenarios and formally describe the decision processes, which are then used as input for configuring the decision support system to meet concrete decision problems.

Keywords: Complex Industrial Processes, Decentralized Decision Support, Artificial Intelligence, Conceptual Modeling, Metamodeling, ADOxx.

1 Introduction

European manufacturing companies are experiencing growing pressure due to the effects of international competition [1]. They have to act in global production ecosystems and proactively adapt to changing circumstances on various levels (e.g., in the design of their supply chain network [2], legal regulation, market needs, and novel and sustainable production techniques [3]). Integration of advanced technological capabilities within the management of these production

⋆ Corresponding author

Reference: R. Woitsch, C. Muck, W. Utz, and H. Zeiner, "Enable Flexibilisation in FAIRWork's Democratic AI-based Decision Support System by Applying Conceptual Models Using ADOxx," Complex Systems Informatics and Modeling Quarterly, CSIMQ, no. 38, pp. 27–53, 2024. Available: https://doi.org/10.7250/csimq.2024-38.02

Additional information. Authors ORCID iD: R. Woitsch – https://orcid.org/0000-0002-4783-4999, C. Muck – https://orcid.org/0000-0002-2654-1070, W. Utz – https://orcid.org/0000-0002-6661-4668, and H. Zeiner – https://orcid.org/0000-0002-6913-8046. PII S225599222400208X. Article received: 4 February 2024. Accepted: 9 April 2024. Available online: 30 April 2024.

environments [4], [5] has played a vital role and affects the way how decision makers (re-)act in these situations, considering the complexity of such a system where human and virtual/artificial actors collaborate [6] to achieve a common goal.

The increased complexity within the manufacturing domain impacts decision-making processes as coordination mechanisms are required, and a shared understanding between the involved actors is needed. Decision support is evolving from a central, potentially hierarchical structure towards a decentralized [4], networked setting, bringing it closer to the context where solutions are applied. Complexity also increases through the number of parameters that influence the decision-making process. Market competitiveness is enhanced by optimizing production time or machine utilization within production lines and integrating factors such as resource efficiency, sustainability, energy consumption or worker well-being.

One example of a decision problem is assigning workers to production lines by considering various parameters, such as their availability, whether the workers have the required qualifications, and their current physical condition [7]. This not only includes the planning of the worker allocation but also quickly adapting the plan to urgent changes, e.g., if a worker is absent or materials are missing. In such cases, the plan of produced products must be changed, and subsequently, the workers must be newly assigned. This gets even more complex if the workers' preferences should be considered. Therefore, sophisticated decision support is needed.

To tackle such challenges, the *Democratic AI-based Decision Support System (DAI-DSS)* is prepared, designed, and implemented within the EU-funded *FAIRWork* project. *FAIRWork* is a collaborative project within the Horizon Europe Programme involving eight partners from six countries with an overall budget of 3.5 million EUR.

The DAI-DSS supports the configuration of domain-specific decision strategies enabled by AI-based decision services, which can be applied to concrete decision problems during the design and operation of production processes.

This article is an extension of [8], providing additional details on the model-based design process established in relation to the DAI-DSS and how conceptual models can support the configuration of the DAI-DSS. As the project is still running, this article introduces the first results by presenting the envisioned architecture and a first prototype for the model-based configuration and orchestration of decision services.

The development of the architecture and the prototype was facilitated by design thinking workshops organized in collaboration with the industrial pilot project partners Stellantis/CRF[4] and Flex[5], following *OMiLAB's Digital Innovation Environment (DIEn)* approach [9] [10] and using the Scene2Model tool [11] as a means to digitally capture the design results as input for the requirements engineering process. The models created for the model-based configuration prototype are derived from the design process and use well-established and standardized modeling methods to enable exchange with other systems and organizations.

The remainder of this article is structured as follows: Section 2 discusses related work and the theoretical background. Section 3 introduces the project context and idea in relation to the DAI-DSS. This section is followed by the discussion of the design procedure in Section 4, resulting in an elaboration of decision support challenges (see Section 4.2). These challenges are input for designing the DAI-DSS, from the perspective of model-based alignment of decision problems (in Section 4.3) to the proposal of the architecture of DAI-DSS (in Section 5) and the first model-based configuration prototype (in Section 6). The article concludes by highlighting the next steps in the implementation and the industrial evaluation of the results at *FAIRWork* partners Flex and CRF.

[4] https://www.stellantis.com/

[5] https://flex.com/

2 Theoretical Background and Related Work

Multi-Agent Systems. Distributed decision-making is a process where the decision-making responsibility is distributed across a group of individuals or agents instead of centralizing it with a single agent or group [12]. Such distributed decision-making is typically modeled using multi-agent systems. The global system behavior emerges from the interaction of the individual agents. Each agent contributes to its objectives and has defined knowledge, behavior, and skills. This allows the development of a management system from a bottom-up perspective. Each agent has a partial view of the system and must interact with others to achieve its objectives as described in [13]. Agents operate by an explicit definition of their behavior. Each agent has a formally modeled strategy and acts according to this definition.

Conceptual Models. As a means to externalize the knowledge in an environment, such as agents' strategies or definition of the decision logic for a service, hybrid models [14] are required. Hybrid models address the needs for human expert interpretation such as intuitive readability, graphical representation, ease of use, and machine interpretation (e.g., formal correctness, completeness, and expressiveness). This can be achieved by using models of different degrees of formalization to act as "mediators". This means that strictly formalized, machine-interpretable models are linked with less formalized but intuitive and graphically represented human-interpretable models. For instance, the Business Process Model and Notation (BPMN) is typically used to model business processes and can act as such a "mediator" as it can be extended, e.g., with Business Process Feature Model (see [15]) to deal with process variability or as suggested by [16] a separation of business process and the decision models can be achieved. This is specifically relevant for the DAI-DSS as the mechanisms are required to move between the formalization/abstraction levels to enable decentralization (involvement of stakeholders with different domain expertise) and democratization (transparency for communication). A possibility to represent these hybrid models is conceptual modeling.

Conceptual models can not only be used as mediators between machine-readable models and human interpretable models but can also be used to connect high-level design models of human experts with abstractions of runtime systems (e.g., using Cyber-Physical Systems, CPS), as introduced in the approach of [17]. This approach implies that the mediation is not only on the level of mapping and communication but that – assuming well-defined execution semantics of the underlying metamodel – these models also impact run-time configuration. This means that they bridge representations of business ecosystems with technical aspects of digital or physical twins. A prototyping environment, in the context of a product-service system, is presented and discussed in [9]. Models on different abstraction levels are used and linked together, enhancing the model value. However, to utilize model value, not only modeling methods but also modeling tools are needed to implement processing and interaction capabilities. This results in a more understandable planning process for involved stakeholders in a setting where people have autonomy and use agile methods to ensure the alignment [18].

Design Thinking. To allow support for designing innovative solutions in complex systems, such modeling tools must not only be able to manage and process digital models but also support the co-creation by different stakeholders [19]. For the creation of high-level representations of complex systems, OMiLAB offers the Scene2Model tool to support design thinking workshops. In these workshops, physical objects (e.g., article figures) are used to represent the system under study on an abstraction level that is suitable and understandable. Scene2Model offers an automated transformation from the physical objects to a digital model: the physical objects are identified and mapped to a digital model using an ontological lookup mechanism. The identification is feasible using different approaches, for instance, by enhancing the physical objects with pre-defined tags [11] or by using AI-based methods to recognize the used objects utilizing trained, domain-specific neural networks [20]. This Scene2Model supported design thinking approach is specifically

applicable in the *FAIRWork* project as it involves the expertise of various stakeholders (production engineers, shop-floor operators, business management, potentially customers, legal experts) during the design but also retrospectively during the assessment of the decision-support system.

Artificial Intelligence Services. In today's industry, individual and complex decisions are made using AI-based methods [21]. This starts with multi-objective methods [22] that distinguish between different system concerns, such as interacting with a digital twin or applying individual decisions. Other AI techniques are used to improve the multi-dimensional optimization of these multi-objective solutions, such as reinforcement learning, supervised learning, unsupervised learning, and symbolic AI [23]. The decision support system then presents multiple recommendations to the user. This enables informed and collaborative decision-making [24]. To provide the context to AI services, (i) an adequate knowledge representation is required, and (ii) the data asset services [5] must be available to train the system based on historical cases. For these aspects, digital shadows and digital twins are applied.

Digital Twins. Building upon the definition of digital shadows [25] and digital twins [26], in *FAIRWork*, these concepts are understood from a knowledge and data representation perspective, specifically in the context of complex industrial processes. In FAIRWork, these concepts are used to gather real-time information and make it available within the knowledge base as input for decision-making.

An adapted version of the definition and approach of [27] is relevant:

- Digital shadow (DS) and digital twin (DT) exist for physical and virtual assets, i.e., objects that change their state over the asset's life cycle.
- The digital shadow is defined as an assignment of status data to a specific asset at a particular time.
- There is only one and no second digital shadow of an object, including and combining all relevant properties along the specific asset's life cycle.
- The structure of twins is application-oriented and not universally valid.
- In contrast to a digital shadow, the digital twin allows for an adaptation of the asset via the twin (bi-directional).

In the *FAIRWork* project, the concepts are applied via model-based techniques. As discussed above, conceptual models are considered digital shadows of the production environment, enhanced with historical operation data. The digital twin exists through model-processing techniques and abstraction/decomposition approaches. Based on model-level learning, scenarios are developed and can be deployed directly on the production infrastructure.

3 Project Idea

Manufacturers can no longer afford to use only time and costs for production planning; due to the global scope of today's economy, environmental and social factors must be considered, too. Through the EU-funded *FAIRWork* project, novel technologies are applied to tackle these challenges and improve the decision-making processes. These technologies can gather and pre-process data, allow multiple actors to participate in the decision process, and use AI to identify the solution space for complex problems. One project result will be the *Democratic AI-based Decision Support System (DAI-DSS)*, integrating different technologies to support decision makers. This section will discuss the influencing factors for the development of the DAI-DSS.

One goal for the DAI-DSS is the facilitation of democratic and flexible decision-making. Democratization in this context means that we focus on the participation aspect and do not delve into the political aspects of democratization. Therefore, we support democratization through participation, which means in our context that all involved decision stakeholders can contribute and are considered in the decision-making process. Consequently, participation-based democratization

requires decentralized decision-making, where the involved stakeholders can contribute, in contrast to centralized decision-making, where a central person or small group makes the decisions on behalf of the organization.

Decentralization can be accomplished by directly involving the actors or using agents representing them and acting on their behalf. Using agents allows for further support, as the agents can negotiate between themselves and find solutions benefiting the involved actors. Information is needed to define the decision process, independent of whether a decision is made centrally or de-centrally. Therefore, a knowledge base is required to save, offer, and manage data from different sources, encode it in various formats, and provide static, dynamic, or even real-time input.

The *FAIRWork* decision support system utilizes AI algorithms capable of (i) understanding complex decision problems and (ii)) providing feasible solutions as decision support. The goal is not to identify a single AI algorithm that can help with every decision problem but to offer different implementations of AI algorithms within the decision support system configured for specific purposes to support concrete problems. The decision support system can manage and control the different modules and make them work together. It orchestrates the modules (e.g., various implementations of AI algorithms, data from the knowledge base, input from involved stakeholders, etc.) to identify solutions to concrete decision problems. This granular structure of the system and the reconfigurability of the decision services facilitate the flexibility of the overall system and will be supported through model-based configuration.

An example of such a decision problem is allocating workers to production lines. Considering their qualifications or physical capabilities is essential, and integrating them with their individual needs establishes the knowledge base to be considered (e.g., how much experience the worker has with a specific production line). This example will be used in Section 6 for our prototype. Another scenario in the project targets the support of choosing an alternative for a given decision problem. The most adequate solution may be a combination of various parameters such as lead time, worker satisfaction, and environmental conditions.

These cases are examples identified within the *FAIRWork* project to provide input for the system's design. The cases stem from the two manufacturing companies Stellantis/CRF and Flex. The concrete use cases have been abstracted to *Decision Support Challenges*, introduced in Section 4.2. A more detailed description of the use cases can be found in [7].

4 DAI-DSS Design Procedure

In this section, we introduce the procedure which guides the creation of the DAI-DSS in the *FAIRWork* project and how the DAI-DSS will be applied in real-world scenarios.

Our approach is in line with the research methodology of design science [28]. We start with an industry-relevant decision-making problem. From this application-oriented scenario, an 'artifact' is created. In our case, this is the decision process, which consists of several individual decisions. This is used to implement the solution approach. The solution is then tested. In this way, we understand how the solution works.

These phases follow a *Plan-Do-Check-Act (PDCA)* procedure [29], which consists of four phases that are circular in nature, i.e., they are repeated several times to improve the overall result (design iterations, continuous improvement). This procedure was created to fulfill two main tasks within FAIRWork.

Firstly, it served as a tool to guide the user through the different phases of the selected artifact (i.e., the decision-making process with the proposed solution approach). We modeled our approach by using PDCA, as it provides an iterative approach to continuously improve the current state in each phase. This fits our approach as we want to enable feedback loops with the stakeholders involved. If the proposed solution approach leads to an improvement, the approach can be improved in further iterations, e.g., different configurations.

Secondly, if the solution approach does not lead to an improvement and therefore no benefit, the process starts with an alternative solution approach for the chosen decision process. This new solution (e.g., a new artifact) will also go through all the phases of the PDCA approach.

4.1 Phases of the Design Procedure

The **Plan** phase analyses the problem and creates one or more solution strategies to solve it. In this article's context, the type of decision and its context must be understood to provide "fitting" and "adequate" decision support. After it is understood, the essential aspects of the decision are captured and described comprehensibly. This requires a domain-specific vocabulary in the form of a metamodel to derive conceptual models. The model-based approach for understanding and capturing the needed knowledge of the decision problems and their context is described in Section 4.3 and visualized in the top layer of Figure 1. This approach was established and used in two manufacturing companies to guide the design of the DAI-DSS, and the results were abstracted into the decision support challenges, introduced in Section 4.2.

After the decision problem is understood, a solution strategy is designed and configured utilizing the models. The configuration contains the preparation and orchestration of different decision support modules and the connection to the knowledge base to gather the needed information. Therefore, after the *Plan* phase, the DAI-DSS is established to offer support for the identified decision problem.

In the **Do** phase, the prepared system is used in real-world environments to support the production process containing the identified decision problems. In the project, an experimental approach is applied to evaluate DAI-DSS first in a small-scale setting under laboratory conditions. This enables the project's industrial partners to assess its applicability. This includes real-time data processing of the system, containing actors' input, to provide contextual information, or start a decision support process. During the execution of a decision support process, data is collected as input for a later evaluation and planning of adaptations. Which data is collected depends on the concrete usage. Still, the system enables mechanisms to persist this data, for instance, through log files or directly save it to the knowledge base as input for calculating the system's key performance indicators (KPI).

After executing one or multiple decision support processes, the **Check** phase is applied. First, KPIs are defined to evaluate the DAI-DSS configuration for the specific decision problem. Afterward, the KPIs are calculated and prepared based on the available data to assess the decision support system as a whole or to analyze specific parts, e.g., determining if the chosen AI approach was feasible or not. The KPIs themselves focus on different aspects, e.g., analyzing the influence of the decision support system on the production process, whether the available data is sufficient, whether all needed stakeholders could participate, or whether the decision support tool is adequate for the task.

Last but not least, in the **Act** phase, reflection on the results from the previous phase is triggered to learn how the decision support can be improved for future usage for the same or similar decision problems. Different starting points for improvement can be identified based on the identified problems with the current solution, e.g., the specification of the problem, the used AI algorithms, or other parts of the configuration. The identified problems and starting points are then used as input for the *Plan* phase of the next iteration. The findings from the *Act* phase can be applied not only to upcoming iterations of the configured system, as they can also be utilized as lessons learned for similar decision support processes, or insights can be generated on how new decision problems can be supported with the DAI-DSS.

4.2 Decision Support Challenges

To guide the design of the DAI-DSS, multiple decision problems of the two manufacturing companies CRF and Flex, were thoroughly investigated during the *Plan* phase of the

above-introduced design procedure (see Section 4.1). Following the set objective to support flexible adaptation to changing or newly identified decision problems, three categories of decision support challenges were abstracted from the concrete decision problems of the industrial partners. These challenges are used to guide the design of the DAI-DSS and are introduced in this section. More information on the underlying use cases from the manufacturing companies can be found in [7].

The idea behind using abstracted decision support challenges and not concrete decision problems is attributed to the project's aim of achieving system flexibility. The DAI-DSS should be configurable to allow its use in changed or new decision problems. Therefore, the design of the overall system and the individual decision services must consider this adaptability, which leads to the definition of the decision problems on a higher abstraction level, hence the decision support challenges. Different decision services can then be usable for one or more decision support challenges and be adapted to work in specific cases belonging to the challenge.

Adapting a generic version of a decision service to a concrete instance should be done preferably through model-based configuration and only through (re-)implementing a service where necessary. Using conceptual models should lower the barrier to entry so that not only technical experts but also domain experts can establish their decision services.

The goal is not to design one decision service for one decision challenge but to ensure that users of the DAI-DSS can find and use services that best fit their needs. Additionally, as the decision services are designed to be more generic, they may not cover a whole decision challenge. Multiple services should be orchestrated to fulfill one concrete decision problem.

Below, the identified decision support challenges will be introduced.

1) Resource Mapping is the category of decisions where a requester and a provider of artifacts or services must be matched to achieve a common goal. For instance, workers (providers) with specific capabilities must be matched to machines (requesters) to fulfill the orders. Here, the characteristics of the provider and the requester must be considered and compared to each other. This does not necessarily lead to the decision of whether an allocation is a match but how fitting it is so that different allocations can be compared. This mapping goes beyond a one-to-one mapping towards mappings between sets of requesters and sets of providers.

2) Solution Configuration is applied in the context of an adaptive system (e.g., a production line), and the decision support helps to decide which configuration of the system is feasible or allows a comparison between the configurations. The resulting configuration must be feasible within the company and the current restrictions, based on available resources and orders, therefore considering the production ecosystem as a whole.

3) Selection is the decision support challenge for decisions that are applied after *resource mapping* or *solution configuration*, which result in a list of possible alternatives from which one must be chosen. *Selection* contains algorithms that support a comparison of different possible solutions and rank them accordingly. Therefore, context information of the solution proposals (from *resource mapping* and *solution configuration*) is used as input to be used for comparison, including the human in the loop.

Lastly, it should be mentioned that the objective of the established decision support challenges is not to cover every possible decision problem in every production environment and provide a complete set of all possible decision support problems. They are abstractions of the project's pilot cases and are used as the foundation of the DAI-DSS so that the design is not too focused on the specific cases but allows for decision support for a broader range of problems. Therefore, additional decision-support challenges may be identified in the future.

4.3 Aligning Decision Problems to Decision Support Configurations

The common requirements from the above challenges result in the following aspects of the DAI-DSS to be considered in this article:

- Common understanding of the decision scenario: The decision problem scenario and its context must be sufficiently understood. This requires means to have an adequate knowledge representation in place.
- Decision support configuration: The adequate decision services and their algorithms need to be identified (based on the common understanding), configured, and then orchestrated to fulfill the needs of the decision scenarios.

Within the DAI-DSS, these two aspects are tackled through model-based alignment, visualized in Figure 1. The figure shows three layers, where the top layer represents how the decision problem is understood and formalized (different levels of abstraction are applicable). In contrast, the bottom layer shows what the DAI-DSS offers concerning decision support. The middle layer mediates between the needs and the available resources. The layers are introduced below, starting from the top and followed by the bottom layer, and then the middle layer of DAI-DSS that is concerned with the configuration environment for decision support challenges.

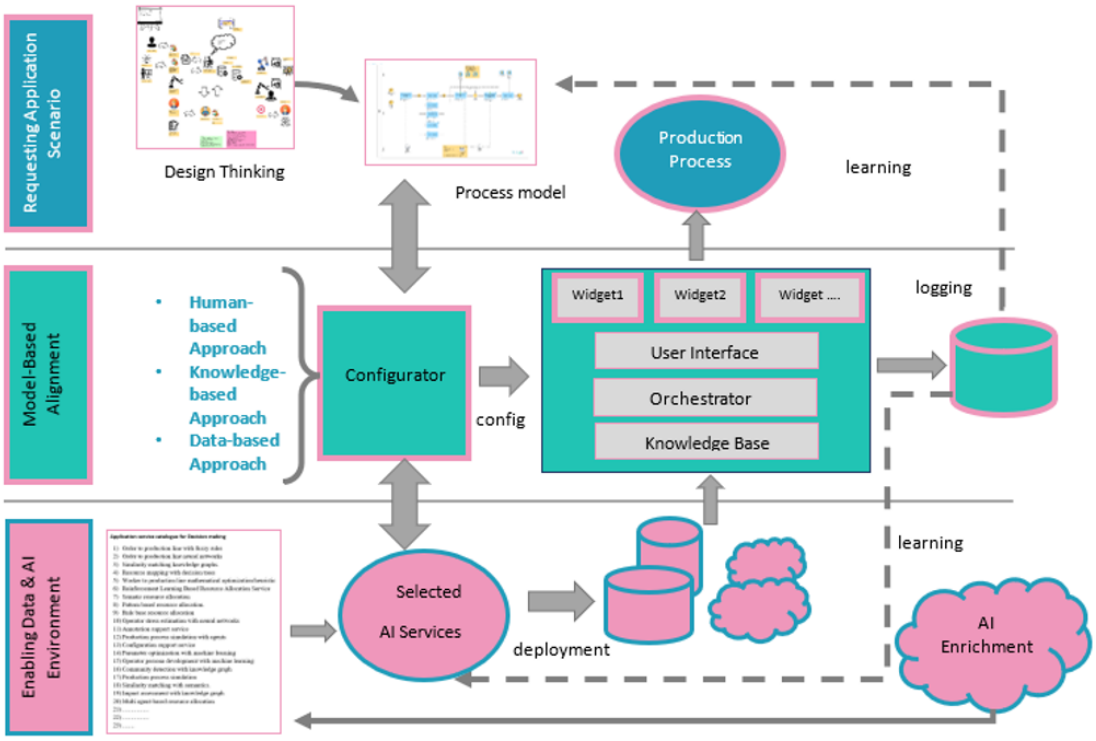


Figure 1. Model-based alignment of decisions problems and DAI-DSS configuration (figure taken from figure 3 of [30, p.25], accessed 29.01.2024)

Application Scenario. As a means to understand the decision problem scenario, which should be supported by the DAI-DSS (represented on the top layer of Figure 1), a model-based approach is used. The first step in establishing automated decision support is understanding the decision problem, its context, and the goal. Without this information, no sufficient solution can be found.

The following decision elicitation and design methods were created and used in the FAIRWork project to support this crucial first step. This method was made to analyze and understand the decision problems of FAIRWork's use case partners, from which the decision support challenges were later derived. Additionally, it will be later used as part of the DAI-DSS and guide its usage. Therefore, the method was used in two manufacturing companies, one from the automotive sector and one focusing on creating small lot-size products for varying companies.

The method aims to facilitate stakeholders' understanding of the decision, its goal, its parameters, and its context to establish sophisticated decision support. The method uses conceptual models as

the primary artifacts to capture and communicate information about the decision problem. Different conceptual modeling methods with varying levels of abstraction are used in various stages of the method. The created models will start on a high abstraction level focused on understanding, and over time, more formal and exact models containing the decision information will be made. The five stages of the method are visualized in Figure 2 and were first introduced in our deliverable [7]. The models for *High level scenario*, *Process landscape* and *Decision process* shown in Figure 2 are shown in larger size in Section 6.

The first two stages focus on exploring and understanding the problem domain by defining concrete decision scenarios that must be made in the company. They are based on applying physical workshops to capture the needed information from involved and knowledgeable people. For the workshops, we used the Scene2Model tool (cf. [11]), which uses paper figures based on SAP Scenes[TM6]. The physical paper figures are placed on a table and then used to visualize and discuss the participants' understanding. The physical interaction facilitates cooperation and participation during the workshop.

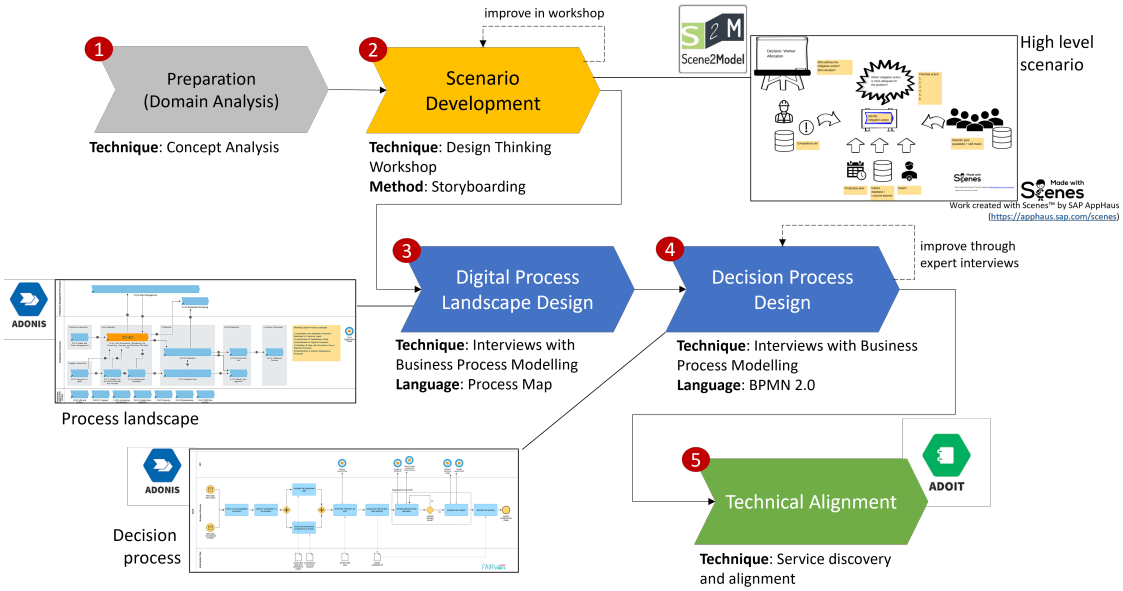


Figure 2. Visualisation of the method for understanding the decision process (based on Figure 1 from [7], accessed 17.01.2024)

The *Preparation* stage prepares the workshop. Therefore, the workshop's goal, the participants, and the used paper figures must be specified. Even though the workshops should be explorative and allow for creativity, the overall context should be set by knowing which decisions should be analyzed, what the problem is, and what should be solved. This information is essential in preparing and running the workshop smoothly. The preparation stage includes finding and preparing the paper figures and the workshop environment to be used. For the workshop environment, Scene2Model's technical infrastructure must be set up, e.g., preparing the modeling tool to work with the selected paper figures. More information is available in [11]. The paper figures must fit the problem domain so the participants can use and understand them.

The next stage is called *Scenario Development*, and it is based on physical workshops where stakeholders with different backgrounds who are influenced by the decision problem participate. The workshops are used to externalize the participants' knowledge through high-level-abstraction representations with paper figures. It focuses on a visually understandable representation of the decision problems through the provided figures and not a rigorous formalization. The formalization

[6] https://apphaus.sap.com/scenes, accessed 17.01.2024

is needed and will be created later, but in this stage, understanding and communication are the focus, and, therefore, high-level-abstraction visual representations are used. An example of the results of a workshop can be seen in Figure 2 on the top right.

After a workshop, the digitalized knowledge is concretized using fitting conceptual modeling methods. This formalization is done over the following two stages of the method, which are called *Digital Process Landscape Design* and *Decision Process Design*. In these stages, the context of the decision problem and the definition from the second stage are formalized in more detail. The knowledge needed to implement the decision support later is specified through this formalization. In FAIRWork, we used the BPMN modeling language because it could represent the context and steps, and most project partners were familiar with it.

To increase the available knowledge and fill in the blanks that could not be provided during the workshops, interviews and discussions with the experts on the decision problem were held as follow-ups. These were used to improve the models created in stages three and four.

In the *Digital Process Landscape Design* stage, an overview in the form of a process landscape is created. These models represent the high-level processes related to the decision problems and provide context for the concrete decisions that should be made. An example of a process landscape can be seen in Figure 2.

Afterward, in the *Decision Process Design* step, the concrete processes containing the decision problems are modeled to capture the semantics of the decision in dedicated BPMN models. Here, capturing the decision, the sub-decisions, and the influencing factors is essential. An example of such a BPMN model can be seen in Figure 2. After the initial version of the decision process design is created, it is evaluated by interviewing experts who know this decision, e.g., production line managers. If the model is not correct, it is adapted and re-evaluated until it reaches a state where the experts accept it.

The last stage visualized in Figure 2 aligns the identified decision processes with the DAI-DSS by configuring it to support instantiate decision support for the concrete decision problem. However, a semantic gap exists between the modeled decision processes and the knowledge needed to execute them in a decision support system such as the DAI-DSS. To close this gap, we used decision models to capture the decision knowledge within FAIRWork. No concrete modeling method is defined here, but one that fits the decision problem should be chosen.

The decision models are derived from the decision processes and specify parts of them where decisions must be made. For one decision process, it may be necessary to create multiple decision models if the process contains various sub-decisions. The decision models are used to capture the decision knowledge, and the models, as artifacts, can be used as input for a configuration environment if designed this way. However, not any modeling method can be chosen, as it must be understandable by the DAI-DSS and the used decision services. More information on the decision services and how they fit into the DAI-DSS can be found in the next layer of the model-based alignment. The prototype for using models as configuration input for decision services can be found in Section 6.

Enabling Data and AI Environment. This layer focuses on how the DAI-DSS can provide decision support using different AI algorithms. The goal is to provide AI algorithms in the form of various adaptable services, which can then be bundled to offer decision support for a specific problem. Therefore, the configuration of the DAI-DSS consists of adapting the AI-enriched decision services to concrete decision problems and combining fitting decision services. But first, the services must be selected, configured, and deployed to make them usable.

The services that should be used influence the possible modeling methods for creating the decision models, as they must provide the decision knowledge in a way that is understandable to the decision service.

As decision services are technical components that consume data and provide possible solutions, they must be integrated into the infrastructure to enable information exchange with other components. To allow interaction with the production environment, we use digital twins and digital shadows connected to the DAI-DSS and the AI services. In this way, actual data can be accessed, and the decision result can be integrated into the production processes.

The integration within the DAI-DSS of the configured services means that they can be integrated using the orchestrator; the data consumed and created comes and goes from the knowledge base and the result and input interfaces can be made over the user interfaces provided through the DAI-DSS.

DAI-DSS Layer: Model-based Alignment. AI services can only lay the foundation for the desired flexibility. Users must be supported in utilizing the flexibility and easing their workload for configuring the system to meet their needs. From the bottom layer, the services' interfaces are provided so they can be integrated. From the top layer, the knowledge of the decision problem domain is provided. In the middle layer, the configurator combines the other two layers to allow the system to provide decision support, as visualized in Figure 1.

The configurator component of the DAI-DSS offers interfaces, which can be graphical for users or APIs, for flexible adaptation. The configuration does not only influence the used decision services but also how they are combined, how the data is gathered from the knowledge base, how it is saved there, how the end users interact with the system to provide input for a decision, and how they can access the result of a decision. Therefore, data exchange between these components must be possible, and the configurator must provide the configuration information so that the needed components can be deployed if necessary and then configured. More information on the different components and how they work together is discussed in the architecture section (see Section 5).

The configurator and the orchestrator must be aware of the available services in a decision service repository. However, the input on the decision and how the different services are configured and orchestrated comes from the top level, where the information is encoded in models. The goal is not to manually translate the modeled knowledge in a DAI-DSS configuration but to reuse the information from the models to support the users. Depending on the decision service and how it is implemented, more or less additional manual configuration must be provided. The better the models created fit into the decision support environment, the better the user can be supported.

Using an instantiated configuration for the decision support bears the additional benefit that the instantiation can be made in different environments, such as experimentation or production. An experimentation environment can be used to evaluate the defined decision support, and only if everything works as anticipated, will it be used in a real system. The experimentation environment can be used to explain how the decision is made and how different parameters influence the final decision.

Last but not least, a configured decision can be reused, adapted to similar cases, and then deployed. This further increases the flexibility, allowing parts of the decision to be reused if they are similar, rather than starting from scratch.

Based on this conceptual perspective on alignment, the technical architecture is presented in the next section.

5 DAI-DSS Architecture

As an initial result, the technical architecture of DAI-DSS has been established. Based on the scenarios/challenges (see Section 4.2) and alignment consideration, the architecture is presented as a distributed, service-oriented environment that utilizes models as a configuration, interaction, and operational artifact. As such, it acts as a blueprint for implementation.

This section will provide an overview of the architecture's main components. A detailed description of the architecture can be found in [30]. The architecture is visualized in Figure 3, and the main components are highlighted with red boxes.

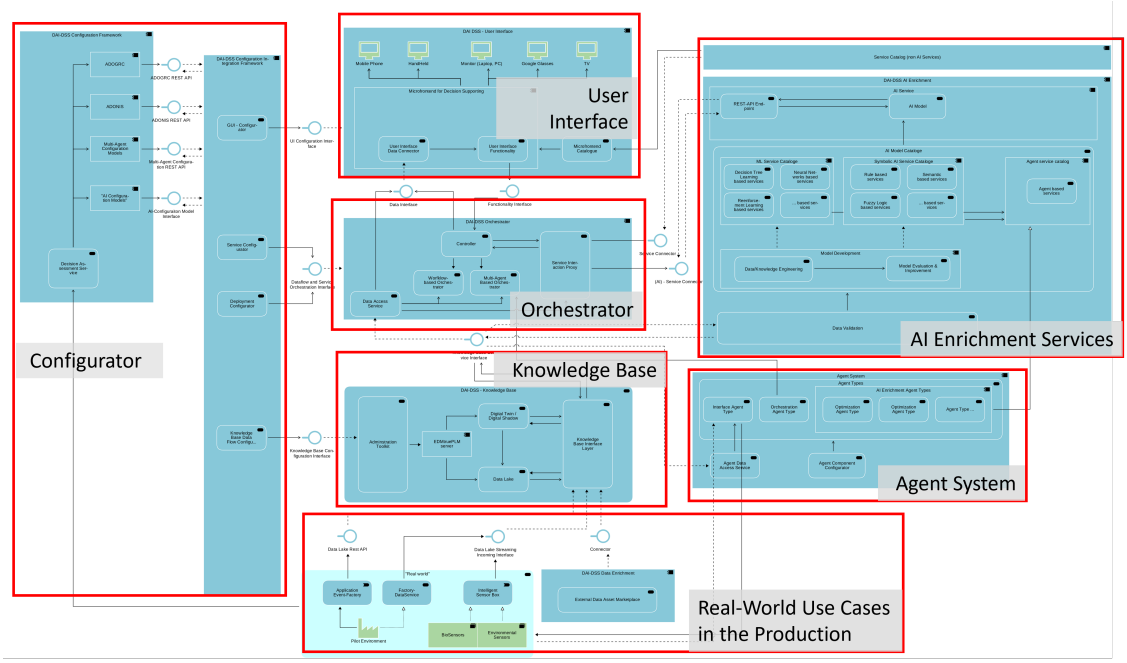


Figure 3. DAI-DSS architecture overview (based on Figure 2 of [30], accessed 29.01.2024)

User Interface. The user interface enables involved stakeholders to interact with the DAI-DSS. Different interfaces will be provided to achieve various tasks. For instance, decision-makers need an overview of possible alternative solutions and their differences. Other users need an overview of the KPIs of the decisions so that they can evaluate their applicability, or actors need an interface to trigger the required decision process. The user interface is defined as its own component, as interfaces for different tasks and also various devices (e.g., laptops, smartphones, TVs, AR glasses, etc.) are needed. Therefore, the user interfaces are implemented separately and linked to the core system via defined interfaces. An important aspect related to the interface layer is provenance and trust. This means that the presented results or visualization are required to enable the drill-down mechanisms to assess how results (algorithm, data) have been achieved.

Configurator. The configurator component allows the adoption of the DAI-DSS and its components so that it can be flexibly adapted to various decision problems. This component will be model-based, meaning it can use models created with dedicated tools to configure the DAI-DSS and its decision services. Additionally, the configurator will offer functionality to asses the configured decisions based on usage data (e.g., logged information or user feedback via questionnaires).

AI Enrichment Services. The AI enrichment services component represents the usage of AI algorithms to provide solutions for decision problems. Different AI algorithms are implemented as the backbone for provided services, following a microservice architecture (cf. [31]). The services themselves can be configured to support different decision problems and are then used by the orchestrator to support concrete decision problems with the DAI-DSS.

AI algorithms of different types, such as symbolic, sub-symbolic, or hybrid approaches (cf. [32]), can be used by this component. It is essential that the algorithm allows an adaptation and usage in the context of the abstracted decision support challenges, which implies that (i) the input/output requirements are clearly defined and (ii) configuration possibilities are foreseen.

The AI enrichment service component aims to create an extensible catalog of AI services, which can then be configured to fit concrete decision problems. Within the service, we use different types of techniques, such as AI-based optimization ([33] or [34]) and reinforcement learning [35]. If a new problem should be solved, the preferred strategy is to first check if fitting decision services are in the catalogue; if not, a new one is developed based on existing libraries or APIs, if applicable, or implemented from scratch, if necessary.

The data for the AI algorithms were provided by the companies involved in FAIRWork, Stellantis/CRF, and FLEX. The usual data quality assurance processes (e.g., missing value checks, consistency checks, etc.) were then carried out. This data was adapted to be used for AI solutions (e.g., Reinforcement Learning). All company data was documented accordingly. In addition, the companies defined KPIs for the use cases and FAIRWork. The aim is always to improve the status quo over time. These KPIs are then also used to evaluate the purely data-driven AI algorithms.

The services are implemented following the microservice principle so that they can be integrated into the DAI-DSS. A discussion of all planned and prototyped services would go beyond the scope of this article.

Agent System. This component represents the environment that enables agents which will be created to represent human and virtual actors within the DAI-DSS. The agents themselves will negotiate for the represented actors to support the decision-making process. Additionally, agents can be used by the orchestrator to support the orchestration of the different services, not through centrally designed workflows, but by cooperating to find feasible solutions.

Orchestrator. The orchestrator is a critical component of the DAI-DSS, as it controls the configured microservices and facilitates decision support. This includes, on the one hand, managing the life cycle of the microservices, like starting, stopping, restarting, or decommissioning them. On the other hand, the orchestrator also manages the information flow between different services. For instance, one decision in a production line may be separated into two sub-decisions that are executed separately utilizing independent implementations of microservices. The results are later combined into one proposed solution.

The orchestration can be done by using workflows or multi-agent systems. Users define workflows and specify which service or other components are used at which stage of the decision support process and how data is exchanged. If a multi-agent approach is chosen, the individual agents will coordinate their actions to find solutions for the decision problem.

Knowledge base. The knowledge base of the DAI-DSS will be the central repository for saving and retrieving data. The DAI-DSS requires data from different sources, which are used as input, e.g., real-time data from a production line or context information such as workers' qualifications. Additionally, the data created by the decision support system is stored and persisted in the knowledge base. The structure of the knowledge base is based on two concepts: digital twins/ digital shadows and data lakes. Digital shadows/twins have been introduced in Section 2. The question of the differences between the digital shadow and the digital twin arises. The digital shadow can be seen as a preliminary stage of the digital twin. The digital shadow forms the basis for the digital twin, which describes measurement data or metadata associated with a specific object with a spatial or temporal reference, but not yet the physical object itself with all its properties. Data lakes (cf. [36]) are relevant to storing data from multiple sources combined to provide access in a distributed system, independent of the actual source system/provider.

The knowledge base is implemented using the industry proven *EDMTruePLM*^{TM7} tool, which can handle various heterogeneous data and offers REST APIs to retrieve, add, and change the available data [7]. It allows the storage of data over its lifecycle in the form of documents, files, data sets, and so on, whereby the implementation of *EDMTruePLMM*^{TM} is based on various standards easing the data transfer with other applications. In addition, the tool also allows the integration of additional protocols (e.g., MQTT) to support the creation of digital twins.

Within the knowledge base, an individual structure for each supported use case will be created to tailor the structure to the concrete needs. For instance, for the worker allocation use case used for the prototype, the types for the production line and the worker were created. For the production line, information such as the geometry that should be produced, the importance of the product, and the due date is saved. Identified workers and attributes for mapping them to production lines are saved,

[7] https://jotneit.com/products/edmtrueplm/0, accessed 23.04.2020

for instance, the workers' experience with the geometry, their endurance, their availability, their medical condition, and so on. More details on the knowledge base can be found in our deliverable [37].

Real-World Use Cases in the Production. This part of the architecture does not represent a component of the DAI-DSS like the others but represents a collection of interfaces and data sources that will be accessible within the DAI-DSS. This can either be data created directly by machines from production lines, from external data catalogues or from laboratories. The goal is to support the gathering of data needed to establish decision support through the decision services. Therefore, the DAI-DSS should be capable of using other data sources, e.g., external data catalogues, if the production environment does not possess the data on its own.

The architecture itself is designed to support the system's flexibility and allow for an adaptation to concrete, domain-specific decision problems. Data can be saved and exchanged using the knowledge base component, which decouples the singular components and further supports flexibility.

6 Model-Based Service Configuration Prototype

This section will introduce FAIRWork's first DAI-DSS prototype, focusing on how model-based configuration can support decision-making. Therefore, not all aspects of the DAI-DSS prototype and components of the above-introduced architecture are described. Within this section, we focus on the *Configurator* and *AI Enrichment Services* components and discuss their connection to the *Orchestrator*.

To create the prototype, we took one of the identified decision problems from our use case partners and created the prototype to support the design methodology introduced in section 4.3. Therefore, the remainder of this section is structured in accordance with this methodology. It will first introduce the procedure and tools which were used to identify the application scenario, and subsequently, the concrete scenario used for this prototype will be described. Afterwards, the environment for creating the configurable decision service will be introduced. Next, the main focus of this prototype, the model-based configuration, will be discussed. Finally, the usage of the decision service in the orchestrator will be introduced.

6.1 Application Scenario

As a **use case** for the prototype, we took the *CRF Workload Balance* case from our deliverable [7]. In short, this use case focuses on assigning workers to production lines based on the current orders that the production lines must fulfil. It belongs to the *Resource Mapping* (as introduced in Section 4.2) decision support challenge.

To define this use case, the design method introduced in Section 4.3 (and visualized in Figure 2) was used in the project. First, a workshop for creating the high-level representation was held at the location of the project partner CRF. For this workshop, the Scene2Model tool was used, representing the decision problems with paper figures and using the modeling tool to digitally capture and enrich the visualization. Project partners traveled to CRF as workshop organizers, preparing and leading the workshop. They also knew how to use Scene2Model and what objectives should be analyzed. To provide domain knowledge, experts from the production environments were invited to the workshop as participants.

The result of the workshop in the form of the enriched digital model can be seen in Figure 4. This model was then used to share the created knowledge with the other project partners.

Based on the high-level scenario model, the process landscape model (shown in Figure 5) and the decision process model (shown in Figure 6) were created with the Adonis[8] business process

[8] https://www.boc-group.com/en/adonis/, accessed 24.04.2024

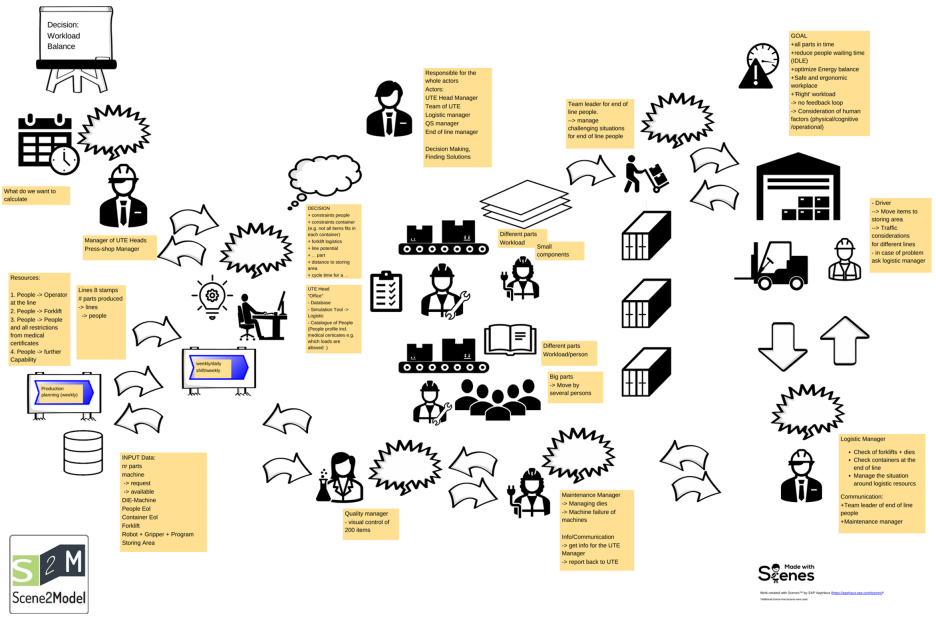


Figure 4. High-level scenario model for the *CRF Workload Balance* case (Figure 17 from [7], accessed 29.01.2024)

management suite. A model expert first created these models and then discussed them with the domain experts from the production line during interviews. Based on the feedback from the domain experts, the process model was adapted and reevaluated in interviews with the experts, leading to multiple interview rounds. The resulting models helped to deepen the understanding of the decision problem and communicate it to the partners so that all the needed information could be derived to implement the prototype.

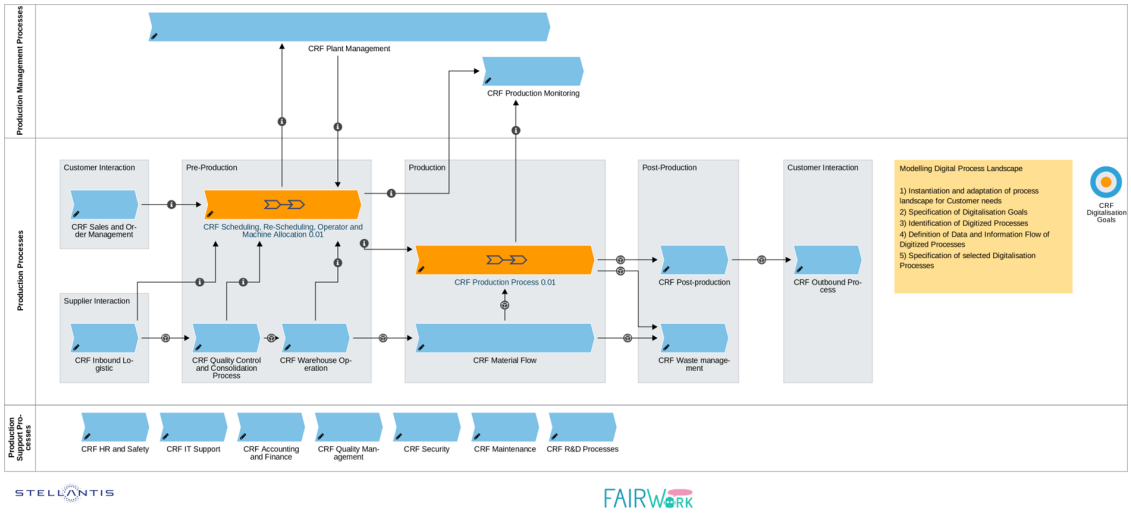


Figure 5. Process landscape model for the *CRF Workload Balance* case (Figure 21 from [7], accessed 29.01.2024)

From the models, it was derived that the available orders influence which products or parts are produced on certain production lines, further influencing which workers can be allocated to specific production lines. First, which workers are allowed on production lines must be decided, especially considering their knowledge about the machines used and the product to be created. Further, the workers' physical capabilities must be considered, especially if heavy and bulky parts

must be manually moved. Within the company, knowledge exists about which characteristics of the workers must be considered to allow workers to work on specific production lines and products.

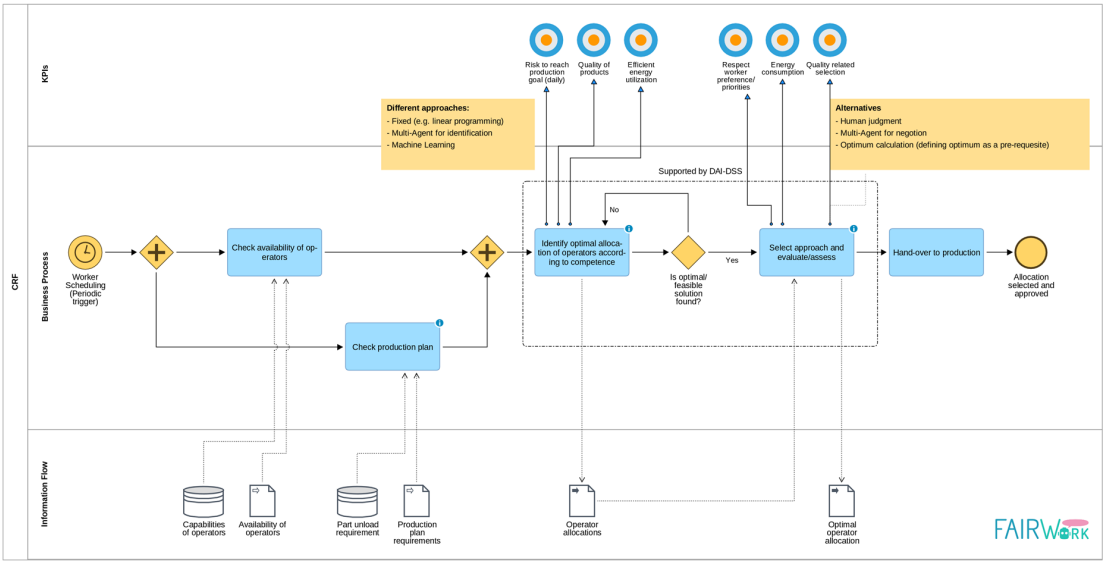


Figure 6. Decision process model for the *CRF Workload Balance* case (Figure 22 from [7], accessed 29.01.2024)

For the use case, the following parameters are needed to decide if a worker is allowed to work on a specific production line (in this list, we also provide how this parameter is later called in the configuration of the service to fit the FAIRWork project and its initial prototype [38] in which context it was created):

- *Medical Condition*: Are there any permanent or temporary medical restrictions why a worker cannot be used for a specific order? The medical condition is checked in a true or false manner. This parameter is called *MedicalCondition* in the prototype.
- *Availability*: Is the worker available during the time when the product should be produced? The availability is checked in a true or false manner. This parameter is called *Availability* in the prototype.
- *Experience*: Is the worker experienced with the products that should be produced? The experience is checked in a true or false manner. This parameter is called *UTEExperience* in the prototype.
- *Resilience*: Can a worker withstand the physical strain of working on the product? The resilience is scored between 0 and 1. This parameter is called *Resilience* in the prototype.
- *Production Priority*: Is the order prioritized? The production priority is checked in a true or false manner. This parameter is called *ProductionPriority* in the prototype.
- *Due Date*: In how many days is the order due? The due date is evaluated as an integer in how many days the order is due or as a negative integer if it is already due. This parameter is called *DueDate* in the prototype.
- *Preference*: This is a rating of the production line, which the workers themselves provide to the system and, in this way, signal their preferences to the system.

All except the last parameter are used to decide if a worker can be assigned to a production line. The last parameter, which is called *preference*, is used to make the final assignment of the possible workers to the production line. This parameter was added to the decision problem by the FAIRWork project and illustrates how the workers' preferences can be considered during decision-making. The parameter presents a vote on which production line and order the workers prefer to work on. The vote is encoded by a value between zero and one, which the worker provides for the different production lines, signaling which production line they prefer.

6.2 Decision Service Environment

To enable decision support for this case, fitting **decision services** must be utilized. Therefore, the DAI-DSS must offer an environment where different services can be created and used. This subsection discusses the decision service environment used for the prototype.

Based on the identified use case, we separated the decision into two parts for the prototype; wherefore two services are needed. One is the decision if a worker can generally be assigned to a production line producing a specific order, and the second is the concrete assignment of possible workers to the production line, based on their provided preferences.

Two decision services will be used to provide decision support. One will be a rule-based and configurable AI service, and the other will be a service that was specially created for the case. Their combination will be done through a workflow-based orchestrator. Last but not least, a model-based configuration of the AI-based service will be used to encode the already available decision knowledge, which is the used parameters and how their values influence the decision.

We chose a rule-based AI approach as we did not have enough data to train a machine-learning model for the use case. However, we were able to extract the expert knowledge with the procedure introduced in Section 4.3. With a symbolic AI approach, expert knowledge can be encoded in a machine-understandable way [32].

A fitting framework is needed to allow the instantiation of the AI-based decision service and support a configuration. For this prototype, we chose the Olive[9] microservice framework [39], which allows the configuration and starting of microservices. The framework has already been applied in multiple EU-funded projects, where it was used together with conceptual models to configure it or represent information created by its services.

Olive allows the creation of new microservices within a single installation through the configuration. Here, a microservice can be created containing different operations, which are configured instances of functionality offered by the microservice. The operation then offers an endpoint where REST calls can be received. The operation processes the information and provides the result based on the configuration.

The framework allows to add to its functionality or introduce new types of operations by implementing what is called a *connector* in Olive terms [39]. Simply put, a connector implements a generic functionality, which can be instantiated within a microservice to create an operation. Therefore, the generic implementation must consider configuration capabilities, which are part of the implementation. Then, users of the Olive framework can create their own microservices and use the available connectors to configure their operations.

We used Olive as an environment where the AI-based decision service can be configured using information from a model. Therefore, we prepared a connector which takes a model containing the definition of the rules, which are used as input for the configuration. After the configuration is done and the connector is instantiated, a REST interface is available, which can be called by other applications to use the decision logic.

In the prototype, the decision models provide the decision logic and the needed parameters for the decision. The modeled parameters are then configured to be parameters of the decision service endpoint so that another application can gather the required information, send it to the endpoint, and then retrieve the proposed solution for the decision. Olive offers a web interface to configure the service where the decision model can be uploaded as input. The other parameters needed for the service usage, such as the endpoint parameters' names, can also be configured in the graphical interface.

The second service for assigning the allowed work to a production line based on the preference was implemented from scratch and integrated into the decision service environment as an endpoint.

[9] https://www.adoxx.org/live/olive, accessed 29.01.2024

Therefore, all the services, including the Olive environment, are offered over web applications and offer their endpoint.

The expertise needed in this step depends on the concrete situation, where configurable services are available for the concrete use case or if a dedicated service must be implemented. If a dedicated service must be created, software engineers who know about the DAI-DSS and how to integrate new services are needed. In addition, experts on how the specific case can be solved through decision service are needed; therefore, they are knowledgeable about applicable algorithms.

No software engineer or expert on decision support algorithms is needed if a configurable service is available. Here, someone familiar with the DAI-DSS and its model-based configuration environment is required. This person must know which services can be used and how they can be configured to fit the concrete case.

Independent if a service must be implemented or configured, a domain expert, understanding the decision scenario, must be available to provide input and evaluate if the decision service is working correctly.

6.3 Model-Based Alignment

This section discusses how the model-based alignment, as it was introduced in section 4.3, was implemented in the prototype. The discussion is starting with the modeling method used for the decisions, followed by the configuration of the decision service, and ends with a short introduction to how this relates to orchestration.

Decision Model. In this first prototype, we used a well-established modeling language for creating the decision models as the basis for configuring decision services. We chose the *Decision Model and Notation (DMN)* method (cf. [40]), as it was dedicated to decision modeling (cf. [41]). To represent the model, the Bee-Up tool was chosen, as it has the DMN method already implemented, it is free to use and adaptable. We then adapted it to define the rules more efficiently and create an export for the modeled knowledge to be used in the Olive-based decision service.

As the DMN method is already established and known, we did not have to start from scratch to create a specialized modeling method, which is a difficult task. In the later stages of the FAIRWork project, tailored modeling methods could be used as input for the configuration if no fitting and well-established methods can be identified.

To create the DMN model, we took the input from the use case and first structured the available decision model. The goal was to model the rules for deciding if a worker is allowed on a specific production line. For this decision, two knock-out criteria were identified. Workers can never be assigned if they are unavailable or do not fit the medical condition. Within the definition of the decision structure, we considered this by creating a sub-decision with these two parameters, which we called *Worker Able*. This allows us later to keep the rules more understandable. The result of the sub-decision is then used as input for the final decision, which we called *Worker at Line* in the model. The decision model is shown in Figure 7.

The other identified parameters for the decision were used as direct input for the final decision. For the parameters, not only the name but also their data type must be provided. For instance, if a ”1” or a ”0” is then used, it must be defined if it should be interpreted as a boolean or an integer.

The model was created for our project deliverable, where more technical aspects can be found [38]. The input parameter names are those we used within the project.

The next step in defining the decision logic was to establish the rules. To further reuse existing and evaluated knowledge to fit the DMN modeling method, we aligned the rule definition within the prototype with the DMN standard (cf. [40]). Here, we used the syntax and semantics for the rules defined in decision tables. For each decision, the rules must be defined in its own decision table, where each line defines then one rule. The service takes these definitions and then searches for fitting answers to the posed question.

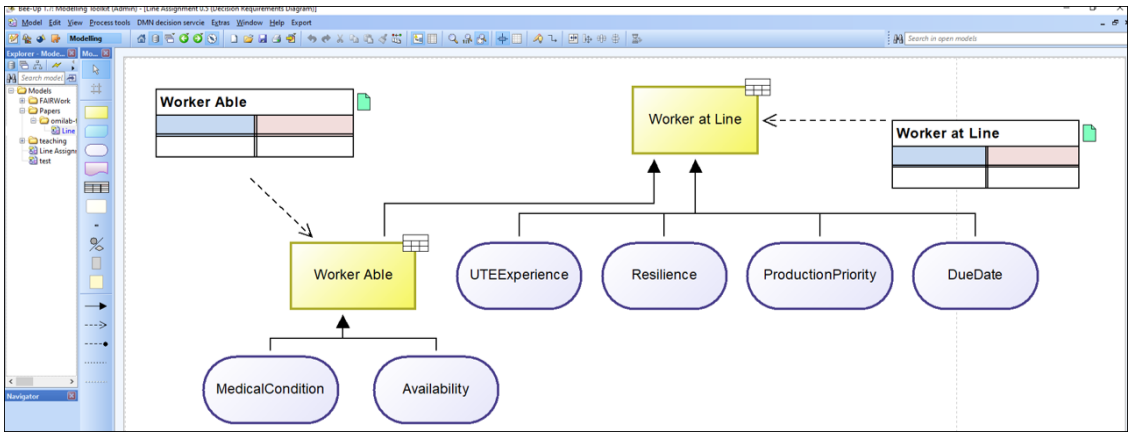


Figure 7. Screenshot of the Bee-Up tool with the decision model created for the prototype (picture based on Figure 32 from [38], accessed 29.01.2024)

The decision table for the sub-decision ensures that it returns true if the medical condition and the worker's availability are true and otherwise returns false as input for the main decision. The rules defined for the main decision can be seen in Figure 8. To ease the interaction of the modeler with creating the decision rules, they are created in CSV format and then imported into the model. For the creation of the decision table, a local program like *Microsoft Excel* or *Libre Office Calc* can be used, whereby the modeling tool can make the basic structure of the CSV file so that the parameter names fit the ones used in the models.

In the decision table, the values which should be considered in one rule are written in one line, and the most right column specifies the outcome of the rule. The other values are connected with *and* logic. Additionally, to state the input parameters' value, the wanted value can be provided directly or short expressions can be used. For instance, for numbers, we used comparison operators > or <=. For numbers, one can also specify intervals, which a number is allowed in this rule. The syntax is here *[<lower-bound>..<upper-bound>]*, as it was used in the due date column.

Worker Able	UTEExperience	Resilience	ProductionPriority	DueDates	Worker at Line
TRUE	TRUE	[0.0..1.0]	FALSE	[-10..10]	1
TRUE	TRUE	[0.0..1.0]	TRUE	[-10..10]	1
TRUE	FALSE	>=0.5	FALSE	[-10..10]	1
TRUE	FALSE	>=0.5	TRUE	[-10..10]	1
TRUE	FALSE	<0.5	TRUE	[-10..10]	1
TRUE	FALSE	<0.5	FALSE	<=1	1
TRUE	FALSE	<0.5	FALSE	>1	0
FALSE	FALSE	[0.0..1.0]	TRUE	[-10..10]	0
FALSE	TRUE	[0.0..1.0]	TRUE	[-10..10]	0
FALSE	FALSE	[0.0..1.0]	FALSE	[-10..10]	0
FALSE	TRUE	[0.0..1.0]	FALSE	[-10..10]	0

Figure 8. Decision table for the main decision of the experiment

To create decision models, persons knowledgeable of the used modeling method and the decision which should be supported are needed. They then have to translate the knowledge from the use case into decision models.

Model-based Configuration of the Decision Service. After all the rules and input parameters are defined, the information must be made available to the configuration of the decision service. For this, the modeled information is exported in an XML file, which will be uploaded to Olive's configuration interface. Additionally, the modeling tool provides the configuration information

needed for the implemented Olive controller. A screenshot of the configuration interface can be seen in Figure 9.

The configuration consists of three main steps. Upload the file with the decision logic from the model, define which of the decisions in the model should be returned, and which input parameter the endpoint offers. The first step is done by uploading the file exported from the modeling tool. This is done in the *DMN File Path* field of the configuration interface. In the next step, in the *Decision Key*, the identifier of the main decision, as defined in the uploaded file, must be provided. This key is shown by the modeling tool at the end of the file export and can just be copied and pasted.

In the third step, the parameters used in the model must be mapped to parameters used in the service's endpoint. The *Decision Variables* and *Call Configuration Inputs* fields are used for this. The modeling tool also provides the configuration string for *Decision Variables* and they can be copied and pasted. Last but not least, in the *Input ID* field, the name offered by the endpoint must be provided, and in *Matching Name*, the string used in *Decision Variables*. This establishes the connection between the endpoint parameters and the decision parameters. A more detailed description of how this can be used and the source code can be found in the OMiLAB GitLab repository[10].

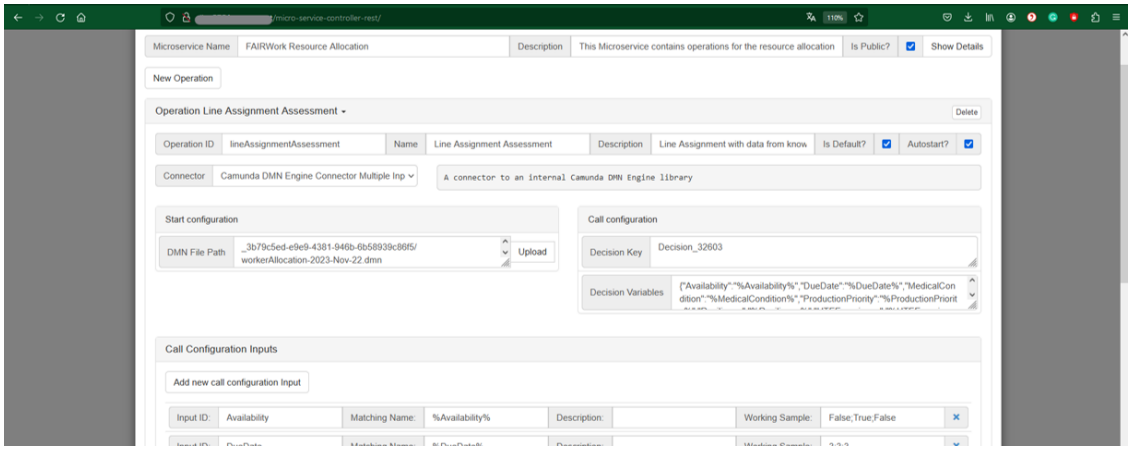


Figure 9. Screenshot of the Olive Configuration Interface for the decision service

To configure the system, the user needs to know about the Olive framework as a whole and the concrete implemented configurable services.

Orchestration of the Decision Services. After the configuration of the decision service, the endpoint for this decision is ready to be used. In the context of the DAI-DSS, the **orchestrator** can use the service to support users in their decisions. The orchestrator is essential, as it manages the order of the decision services that must be called and transforms the data between the services, the knowledge base, and the user interface. In our prototype, the orchestrator first gathers the input information, such as the current production line with the products and the information about the workers from the knowledge base, transforms it, and sends it to the above-described decision service to get a list of all the workers that can be assigned to the needed production lines. Then, it takes this list and sends it to the next service, assigning possible workers to the production line based on their preference. This service also ensures that no worker is assigned to multiple production lines.

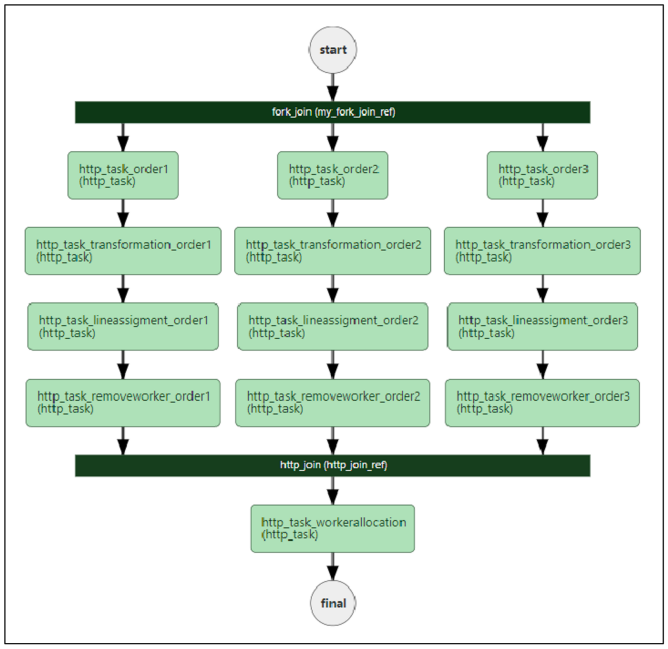
This service represents tailored services specifically developed for decision problems. This is possible as the DAI-DSS architecture uses the decision service to support the decision-making.

[10] https://code.omilab.org/research-projects/fairwork/decision-services/bee-up-dmn-extension, accessed 27.01.2024

These can be configured to fit the needs or specific services can be implemented and added to the DAI-DSS.

For the orchestration in this experiment, we used a workflow-based orchestrator, based on the *Conducter OSS*[11]. We used the orchestrator, which could connect to the knowledge base, the decision services, and the user interfaces. In this case, the user interfaces were used to show the decision results. As the focus of this article is using conceptual modeling within FAIRWork to support the configuration of the decision-making within the DAI-DSS, we are not going into the details of the knowledge base, the orchestrator, or providing the information to the users of the DAI-DSS so that they can receive the result of the decision support.

To explain how the orchestration can work, Figure 10 shows under a) what the orchestration workflow used for the prototype can look like. Here, workers for three production lines needed to be assigned. Therefore, the orchestrator gathers the information for each line separately and calls the service configured with the model to decide which workers are allowed on a production line. Then, the orchestrator joins the three lists and sends this information to the overall allocation, where a section of the result can be seen in the Figure 10. Under b), it shows the service returns a JSON code containing information on the production line and which workers, identified by their ID number, are assigned. Additionally, to these service returns, the input parameters are available in the result, as it is later provided to the users of the DAI-DSS. The result is shown to the decision maker, who can then decide if this allocation should be taken or something should be changed.




```
{
  "allocations": [
    {
      "LineId": "17",
      "WorkersRequired": 6,
      "Workers": [
        {
          "Id": "100060",
          "WorkerPreference": 0.9,
          "UTEExperience": "False",
          "WorkerResilience": 0.87,
          "Availability": "True",
          "MedicalCondition": "True"
        },
        {
          "Id": "100045",
          "WorkerPreference": 0.83,
          "UTEExperience": "False",
          "WorkerResilience": 0.66,
          "Availability": "True",
          "MedicalCondition": "True"
        },
        {
          "Id": "100070",
          "WorkerPreference": 0.82,
          "UTEExperience": "False",
          "WorkerResilience": 0.7,
          "Availability": "True",
          "MedicalCondition": "True"
        },
        {
          "Id": "100011",
          "WorkerPreference": 0.8,
          "UTEExperience": "True",
          "WorkerResilience": 0.7,
          "Availability": "True",
          "MedicalCondition": "True"
        },
```


a) Orchestration Workflow b) Result Example

Figure 10. The figure shows a) visualization of the orchestration workflow from our deliverable [38, p 21] (Figure 12) and b) an example of the result of the workflow for one production line

To use the orchestrator, a user knowledgeable of the overall design for the decision support, how to configure the DAI-DSS orchestrator, and the instantiated decision services is needed.

7 Conclusion

In this article, we extended [8] by providing deeper insights into how we use conceptual modeling in the EU-funded *FAIRWork* project. Within the project, we develop the *democratic AI-based decision*

[11] https://github.com/conductor-oss, accessed 30.01.2024

support system (DAI-DSS) to improve decision-making in production environments. This article first introduces the project's goal, which decision challenges should be supported by the DAI-DSS in its final state, the initial architecture, and how we use conceptual modeling to support both the understanding of the decision and the definition of the decision logic.

Within FAIRWork, conceptual modeling supports the elicitation and definition of the decision logic, facilitating the knowledge exchange between the involved actors, who can be human or artificial agents. Different modeling methods are used based on the stage of the elicitation procedure, ranging from high-level representation of the decision scenario over the processes in which the decisions are made to the definition of the decision itself.

The configuration component is the basis for increasing the flexibility of the DAI-DSS. But also, the orchestrator and the decision services must be designed and implemented to support the configuration and collaboration with the configurator. In our first prototype, we prepared a rule-based decision service to enable the creation of endpoints to make the rule-based decisions based on the configured rules. By combining configurable decision services with decision models, the flexibility of the DAI-DSS is further improved, meaning that the modeled knowledge is reused to configure and instantiate decision services. Using comprehensible modeling methods to define the knowledge and using it to configure and instantiate a service reduces the effort to provide decision support. This is further supported by separating a complex decision problem into multiple sub-problems, which can be solved individually. These are then combined to offer decision support for complex problems.

For the introduced prototype (see Section 6), we used the well-established DMN modeling language to represent the core of the decision logic. But here, we also had to extend the basic functionality to support the configuration of the decision service. The functionality for the export of the decision logic and additional attributes in the concepts were needed. However, not all decision algorithms or configuration aspects of the DAI-DSS can be covered with DMN. Therefore, further adaptations or new modeling methods are needed.

This also poses a challenge during the development, as the underlying metamodels must be appropriate for the purpose. This becomes specifically relevant when utilizing domain-specific modeling languages where execution semantics need to be considered as part of the conceptualization of the modeling method. Consequently, techniques are assessed that allow for dynamic modification of metamodels (potentially at runtime) to capture these aspects and classify modeling techniques according to their specific purpose as defined in [42].

Future work is scheduled according to the project lifecycle to evaluate the environment's feasibility and trigger the components' implementation and integration. The OMiLAB infrastructure is available at *FAIRWork* partners BOC and JOANNEUM RESEARCH and is used to perform small-scale experiments and trigger community involvement to extend and contribute services and/or model-based approaches based on the challenges identified. Through this approach, community-based innovation processes are made feasible.

The project's ongoing and future work regarding this article's topics can be categorized into different branches. One branch is the improvement of the AI-based decision services regarding the covered decision support challenges. The introduced prototype focused on a use case from the *Resource Mapping* decision support challenge (as introduced in Section 4.2). It was chosen as the first case in the FAIRWork project, as it had a high priority for the use case partners, and feasible approaches to handle this problem, like rule-based system and DMN, were identified. But within the project, we also identified other cases for *Resource Mapping* and two more decision support challenges, *Solution Configuration* and *Selection*, which pose their own use cases and challenges.

Therefore, part of FAIRWork's future work is to add decision support for new cases, including ones from the other two decision support challenges.

For instance, another use case from *Resource Mapping* would be to support the maintenance of machines by matching the currently found issues in the production line semantically to

already-applied solutions. Therefore, the description of the problem must be possible in an ad-hoc manner and mapped to possible solutions that can fit exactly or loosely to the current situation. The collection of solutions should be extendible.

A case from *Solution Configuration* is that the configuration for a system should be designed and evaluated to find the best solution. Therefore, the decision support must help describe such a configuration and allow the system to understand and change the parameters to optimize towards a defined goal.

Finally, the system must be enhanced to find solutions for concrete decisions by using the decision services with various parameters and then comparing them. Afterward, it should provide the list with a recommendation based on a predefined objective function to the user so that the user has the last say. This falls into the *Selection* decision support challenge.

In addition, multiple AI algorithms should be made available to support different types of problems. These different AI methods (e.g., symbolic AI, decision trees, heuristics, machine learning, deep learning) can be combined for individual decisions. Orchestration can be centralized or distributed with agent-based orchestration. This service-oriented approach enables good integration with legacy systems in our industrial application domain. In this way, we create different decision services that fit the identified use cases and support different types of AI algorithms. Here future work is to evaluate and implement additional decision services, which can be used in the DAI-DSS. We have one rule-based prototype described in more detail in Section 6. Additionally, we have created first prototypes for services using neural networks, multi-agent systems, fuzzy logic, or decision trees. In this context, future work in the project concerns the identification and implementation of additional AI services, as briefly outlined above.

Another branch of future work will relate to how decisions can be visualized with models to make them comprehensible. Modeling is used not only for configuring decision services but also for capturing more abstract context information for the decisions. The models are used as input to the decision models and can further be used to explain the decisions to be made. Whether the models are used for the decision model or for capturing general knowledge about the decision, the modeling methods must use domain concepts to improve their comprehensibility.

The aim is to use the information available within the DAI-DSS and encode it in models, facilitating a comprehensible representation and explainability. For instance, models can be used to assess and certify solutions, making them better understandable and more reliable [43].

The last branch focuses on investigating how the design procedure (see Section 4.3) can be better supported to ease users' work. Therefore, introduced steps and their resulting artifacts, in the form of models or decision services, must be integrated. How this integration can be supported is still ongoing research in the project. For instance, an early prototype was created that allows technical aspects of the decision service environment to be added to the decision model, allowing the modeling tool to instantiate a decision service directly. Other efforts for automating the steps go toward supporting knowledge exchange between the modeling methods used in the different steps.

Acknowledgements

This work has been supported by the *FAIRWork* project (www.fairwork-project.eu) and has been funded within the European Commission's Horizon Europe Programme under contract number 101069499. This article expresses the opinions of the authors and not necessarily those of the European Commission. The European Commission is not liable for any use that may be made of the information contained in this article. The authors express their gratitude to the project consortium for their input in writing this article, specifically the industrial partners Stellantis/CRF and Flex for the valuable time during the requirements elicitation phase.

References

[1] S. Eckert, "Business power and the geoeconomic turn in the single european market," *JCMS: Journal of Common Market Studies*, pp. 1–20, 2024. Available: https://doi.org/10.1111/jcms.13604

[2] G. Ascari, D. Bonam, and A. Smadu, "Global supply chain pressures, inflation, and implications for monetary policy," *Journal of International Money and Finance*, vol. 142, p. 103029, 2024. Available: https://doi.org/10.1016/j.jimonfin.2024.103029

[3] J.-P. Schöggl, R. J. Baumgartner, C. J. O'Reilly, H. Bouchouireb, and P. Göransson, "Barriers to sustainable and circular product design–a theoretical and empirical prioritisation in the european automotive industry," *Journal of Cleaner Production*, vol. 434, p. 140250, 2024. Available: https://doi.org/10.1016/j.jclepro.2023.140250

[4] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld, and M. Hoffmann, "Industrie 4.0," *Wirtschaftsinformatik*, vol. 56, pp. 261–264, 2014, (in German). Available: https://doi.org/10.1007/s11576-014-0424-4

[5] W. Utz and D. Falcioni, "Data assets for decision support in multi -stage production systems industrial business process management using adoxx," in *2018 IEEE 16th International Conference on Industrial Informatics (INDIN)*, 2018, pp. 809–814. Available: https://doi.org/10.1109/INDIN.2018.8472033

[6] B. Wang, "The Future of Manufacturing: A New Perspective," *Engineering*, vol. 4, no. 5, pp. 722–728, 2018. Available: https://doi.org/10.1016/j.eng.2018.07.020

[7] H. Zeiner, "Specification of FAIRWork Use Case and DAI-DSS Prototype Report." Available: https://fairwork-project.eu/deliverables/D2.1_SpecificationofFAIRWork-v1.0a-preliminary.pdf

[8] R. Woitsch, C. Muck, W. Utz, and H. Zeiner, "Towards a democratic ai-based decision support system to improve decision making in complexecosystems," in *Joint Proceedings of the BIR 2023 Workshops and Doctoral Consortiumco-located with 22nd International Conference on Perspectives in Business Informatics Research (BIR 2023)*, vol. 3514. CEUR Workshop Proceedings, 2023, pp. 209–223. Available: https://ceur-ws.org/Vol-3514/paper94.pdf

[9] D. Karagiannis, R. A. Buchmann, and W. Utz, "The OMiLAB Digital Innovation environment: Agile conceptual models to bridge business value with Digital and Physical Twins for Product-Service Systems development," *Computers in Industry*, vol. 138, p. 103631, 2022. Available: https://doi.org/10.1016/j.compind.2022.103631

[10] R. Woitsch, "Industrial Digital Environments in Action: The OMiLAB Innovation Corner," in *The Practice of Enterprise Modeling*, J. Grabis and D. Bork, Eds., vol. 400. Springer, 2020, pp. 8–22. Available: https://doi.org/10.1007/978-3-030-63479-7_2

[11] C. Muck and S. Palkovits-Rauter, "Conceptualizing Design Thinking Artefacts: The Scene2Model Storyboard Approach," in *Domain-Specific Conceptual Modeling: Concepts, Methods and ADOxx Tools*, D. Karagiannis, M. Lee, K. Hinkelmann, and W. Utz, Eds. Springer, 2022, pp. 567–587. Available: https://doi.org/10.1007/978-3-030-93547-4_25

[12] T. W. Sandholm, "Distributed Rational Decision Making," in *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press, 1999, p. 201–258.

[13] P. Leitão, "Agent-based distributed manufacturing control: A state-of-the-art survey," *Engineering Applications of Artificial Intelligence*, vol. 22, no. 7, pp. 979–991, 2009. Available: https://doi.org/10.1016/j.engappai.2008.09.005

[14] D. Karagiannis and R. Woitsch, "Knowledge Engineering in Business Process Management," in *Handbook on Business Process Management 2: Strategic Alignment, Governance, People and Culture*, J. vom Brocke and M. Rosemann, Eds. Springer, 2014, pp. 623–648. Available: https://doi.org/10.1007/978-3-642-45103-4_26

[15] R. Cognini, F. Corradini, A. Polini, and B. Re, "Business Process Feature Model: An Approach to Deal with Variability of Business Processes," in *Domain-Specific Conceptual Modeling: Concepts, Methods and Tools*, D. Karagiannis, H. C. Mayr, and J. Mylopoulos, Eds. Springer, 2016, pp. 171–194. Available: https://doi.org/10.1007/978-3-319-39417-6_8

[16] T. Biard, A. Le Mauff, M. Bigand, and J.-P. Bourey, "Separation of decision modeling from business process modeling using new "Decision Model and Notation" (DMN) for automating operational decision-making," in *Risks and Resilience of Collaborative Networks. PRO-VE 2015. IFIP Advances in Information and Communication Technology*, vol. 463. Springer, 2015, pp. 489–496. Available: https://doi.org/10.1007/978-3-319-24141-8_45

[17] M. Walch and D. Karagiannis, "How to connect design thinking and cyber-physical systems: the s* IoT conceptual modelling approach," in *Proceedings of the 52nd Hawaii International Conference on System Sciences*, 2019. Available: https://doi.org/10.24251/HICSS.2019.870

[18] X. Xu, Y. Lu, B. Vogel-Heuser, and L. Wang, "Industry 4.0 and Industry 5.0—Inception, conception and perception," *Journal of Manufacturing Systems*, vol. 61, pp. 530–535, 2021. Available: https://doi.org/10.1016/j.jmsy.2021.10.006

[19] I. Vaidian, A. Jurczuk, Z. Misiak, M. Neidow, M. Petry, and M. Nemetz, "Challenging Digital Innovation Through the OMiLAB Community of Practice," in *Domain-Specific Conceptual Modeling: Concepts, Methods and ADOxx Tools*, D. Karagiannis, M. Lee, K. Hinkelmann, and W. Utz, Eds. Springer, 2022, pp. 41–64. Available: https://doi.org/10.1007/978-3-030-93547-4_3

[20] C. Muck and W. Utz, "A Recognition Service for Haptic Modelling in Scene2Model," in *Proceedings of the AAAI2023 Spring Symposium on Challenges Requiring the Combination of Machine Learning and Knowledge Engineering(AAAI-MAKE 2023)*, A. Martin, H.-G. Fill, A. Gerber, K. Hinkelmann, D. Lenat, R. Stolle, and F. van Harmelen, Eds., vol. 3433. CEUR Workshop Proceedings, 2023. Available: https://ceur-ws.org/Vol-3433/short4.pdf

[21] A. Felsberger, B. Oberegger, and G. Reiner, "A Review of Decision Support Systems for Manufacturing Systems," in *Proceedings of the 1st International Workshop on Science, Application and Methods in Industry 4.0co-located with (i-KNOW 2016)*, R. Kern, G. Reiner, and O. Bluder, Eds., vol. 1793. CEUR Workshop Proceedings, 2016. Available: https://ceur-ws.org/Vol-1793/paper6.pdf

[22] S. Zhang, F. Tang, X. Li, J. Liu, and B. Zhang, "A hybrid multi-objective approach for real-time flexible production scheduling and rescheduling under dynamic environment in Industry 4.0 context," *Computers & Operations Research*, vol. 132, p. 105267, 2021. Available: https://doi.org/10.1016/j.cor.2021.105267

[23] A. del Real Torres, D. S. Andreiana, Á. Ojeda Roldán, A. Hernández Bustos, and L. E. Acevedo Galicia, "A Review of Deep Reinforcement Learning Approaches for Smart Manufacturing in Industry 4.0 and 5.0 Framework," *Applied Sciences*, vol. 12, no. 23, p. 12377, 2022. Available: https://doi.org/10.3390/app122312377

[24] J. E. Hernández, A. C. Lyons, P. Zarate, and F. Dargam, "Collaborative decision-making and decision support systems for enhancing operations management in industrial environments," *Production Planning & Control*, vol. 25, no. 8, pp. 636–638, 2014. Available: https://doi.org/10.1080/09537287.2013.798083

[25] F. Becker, P. Bibow, M. Dalibor, A. Gannouni, V. Hahn, C. Hopmann, M. Jarke, I. Koren, M. Kröger, J. Lipp *et al.*, "A conceptual model for digital shadows in industry and its application," in *Conceptual Modeling. ER 2021. Lecture Notes in Computer Science*, vol. 13011. Springer, 2021, pp. 271–281. Available: https://doi.org/10.1007/978-3-030-89022-3_22

[26] W. Kritzinger, M. Karner, G. Traar, J. Henjes, and W. Sihn, "Digital Twin in manufacturing: A categorical literature review and classification," *Ifac-PapersOnline*, vol. 51, no. 11, pp. 1016–1022, 2018. Available: https://doi.org/10.1016/j.ifacol.2018.08.474

[27] T. Bergs, S. Gierlings, T. Auerbach, A. Klink, D. Schraknepper, and T. Augspurger, "The concept of digital twin and digital shadow in manufacturing," *Procedia CIRP*, vol. 101, pp. 81–84, 2021. Available: https://doi.org/10.1016/j.procir.2021.02.010

[28] R. J. Wieringa, *Design science methodology for information systems and software engineering*. Springer, 2014. Available: https://doi.org/10.1007/978-3-662-43839-8

[29] M. T. Schmidt, F. Elezi, I. D. Tommelein, and U. Lindemann, "Towards recursive plan-do-check-act cycles for continuous improvement," in *2014 IEEE International Conference on Industrial Engineering and Engineering Management*, 2014, pp. 1486–1490. Available: https://doi.org/10.1109/IEEM.2014.7058886

[30] R. Chevuri, "DAI-DSS Architecture and Initial Documentation and Test Report," 2023. Available: https://fairwork-project.eu/deliverables/D4.1_DAI-DSSArchitecturev1.0a-preliminary.pdf

[31] T. Cerny, M. J. Donahoo, and M. Trnka, "Contextual Understanding of Microservice Architecture: Current and Future Directions," *SIGAPP Appl. Comput. Rev.*, vol. 17, no. 4, pp. 29—45, 2018. Available: https://doi.org/10.1145/3183628.3183631

[32] E. Ilkou and M. Koutraki, "Symbolic vs Sub-symbolic AI Methods: Friends or Enemies?" in *Proceeding of the CIKM 2020 Workshops*, S. Conrad and I. Tiddi, Eds., vol. 2699. CEUR Workshop Proceeding, 2020. Available: https://ceur-ws.org/Vol-2699/paper06.pdf

[33] H. Zeiner, R. Unterberger, J. Tschuden, and M. Y. Quadri, "Time-aware optimisation models for hospital logistics," in *Decision Support Systems XIII. Decision Support Systems in An Uncertain World: The Contribution of Digital Twins*, S. Liu, P. Zaraté, D. Kamissoko, I. Linden, and J. Papathanasiou, Eds. Springer, 2023, pp. 45–55. Available: https://doi.org/10.1007/978-3-031-32534-2_4

[34] K. Bogner, U. Pferschy, R. Unterberger, and H. Zeiner, "Optimised scheduling in human–robot collaboration – a use case in the assembly of printed circuit boards," *International Journal of Production Research*, vol. 56, no. 16, pp. 5522–5540, 2018. Available: https://doi.org/10.1080/00207543.2018.1470695

[35] A. Nasuta, M. Kemmerling, D. Lütticke, and R. H. Schmitt, "Reward shaping for job shop scheduling," in *Machine Learning, Optimization, and Data Science*, G. Nicosia, V. Ojha, E. La Malfa, G. La Malfa, P. M. Pardalos, and R. Umeton, Eds. Springer, 2024, pp. 197–211. Available: https://doi.org/10.1007/978-3-031-53969-5_16

[36] P. P. Khine and Z. S. Wang, "Data lake: a new ideology in big data era," *ITM Web Conf.*, vol. 17, 2018. Available: https://doi.org/10.1051/itmconf/20181703025

[37] R. Chevuri, "DAI-DSS FAIRWork Knowledge Base at Use Case Site," 2023. Available: https://fairwork-project.eu/deliverables/d5-1/D5.1_DAI-DSS%20FAIRWork%20knowledge%20base_v1.0-preliminary.pdf

[38] G. Vieira, "Initial DAI-DSS Prototype D4.2," 2023. Available: https://fairwork-project.eu/deliverables/D4.2_Initial%20DAI-DSS%20Prototype%20v1.0a-preliminary.pdf

[39] D. Falcioni and R. Woitsch, "Olive, a model-aware microservice framework," in *The Practice of Enterprise Modeling*, E. Serral, J. Stirna, J. Ralyté, and J. Grabis, Eds., vol. 432. Springer, 2021, pp. 90–99. Available: https://doi.org/10.1007/978-3-030-91279-6_7

[40] Object Management Group, "Decision Model and Notation (DMN)," 2023. Available: https://www.omg.org/dmn/

[41] D. Karagiannis, R. A. Buchmann, P. Burzynski, U. Reimer, and M. Walch, "Fundamental Conceptual Modeling Languages in OMiLAB," in *Domain-Specific Conceptual Modeling: Concepts, Methods and Tools*, D. Karagiannis, H. C. Mayr, and J. Mylopoulos, Eds. Springer, 2016, pp. 3–30. Available: https://doi.org/10.1007/978-3-319-39417-6_1

[42] R. A. Buchmann, “The Purpose-Specificity Framework for Domain-Specific Conceptual Modeling,” in *Domain-Specific Conceptual Modeling: Concepts, Methods and ADOxx Tools*, D. Karagiannis, M. Lee, K. Hinkelmann, and W. Utz, Eds. Springer, 2022, pp. 67–92. Available: https://doi.org/10.1007/978-3-030-93547-4_4

[43] R. Woitsch, W. Utz, A. Sumereder, B. Dieber, B. Breiling, L. Crompton, M. Funk, K. Bruckmüller, and S. Schumann, “Collaborative model-based process assessment for trustworthy ai in robotic platforms,” in *Society 5.0*, A. Gerber and K. Hinkelmann, Eds. Springer, 2021, pp. 163–174. Available: https://doi.org/10.1007/978-3-030-86761-4_14