Software and Systems Modeling (2021) 20:1525–1551
https://doi.org/10.1007/s10270-020-00855-w

REGULAR PAPER

Model-driven development platform selection: four industry case studies

Siamak Farshidi[1] · Slinger Jansen[1,2] · Sven Fortuin[3]

Received: 26 April 2020 / Revised: 11 November 2020 / Accepted: 14 December 2020 / Published online: 15 January 2021

Abstract
Model-driven development platforms shift the focus of software development activity from coding to modeling for enterprises. A significant number of such platforms are available in the market. Selecting the best fitting platform is challenging, as domain experts are not typically model-driven deployment platform experts and have limited time for acquiring the needed knowledge. We model the problem as a multi-criteria decision-making problem and capture knowledge systematically about the features and qualities of 30 alternative platforms. Through four industry case studies, we confirm that the model supports decision-makers with the selection problem by reducing the time and cost of the decision-making process and by providing a richer list of options than the enterprises considered initially. We show that having decision knowledge readily available supports decision-makers in making more rational, efficient, and effective decisions. The study's theoretical contribution is the observation that the decision framework provides a reliable approach for creating decision models in software production.

Keywords Model-driven development platform · Decision model · Multi-criteria decision-making · Decision support system · Industry case study

1 Introduction

Software applications are produced and maintained by software engineers, and business processes are introduced and managed by domain experts (non-developers) who mainly understand business [1]. Software development requires interactions with domain experts, necessitating a level of agreement in describing the technical phases of development. Moreover, software products are getting more complicated, so they need to be discussed at different abstraction levels depending on the technical background of the involved domain experts, the development process's phase, and the business objectives [2].

Model-driven development (MDD) is a vision of software development where models play a core role as primary development artifacts [3]. Modeling tools in software production are widespread and have reached a degree of maturity where their performance and availability are increasingly accepted, also by non-technical users. Over the last two decades, an extensive list of modeling tools [4] has been introduced [5] to support MDD, such as low-code/no-code platforms and business process management systems. Such modeling tools and platforms' primary aspiration are to boost productivity and decrease time to market by facilitating development at a higher level of abstraction and employing concepts closer to the problem domain at hand, rather than the ones given by programming languages [6].

According to Gartner, by 2024, MDD platforms will be responsible for over 65% of the application development activity, and three-quarters of large enterprises will be using at least four MDD platforms, as such platforms enable enterprises to develop applications quicker using more capabilities and fewer conventional developers [7].

Communicated by Juha-Pekka Tolvanen.

✉ Siamak Farshidi
s.farshidi@uu.nl

Slinger Jansen
slinger.jansen@uu.nl

Sven Fortuin
sfortuin@bol.com

1 Department of Information and Computer Science, Utrecht University, Utrecht, The Netherlands

2 School of Engineering Science, LUT University, Lappeenranta, Finland

3 bol.com, Utrecht, The Netherlands

A significant number of MDD platforms with a broad list of features and services are available in the market [5], so it is challenging for enterprises to select the best fitting platforms that address their requirements and priorities [8]. The selection problem can be modeled as a multi-criteria decision-making (MCDM) problem that deals with evaluating a set of alternatives and considers a set of decisions criteria [9]. MCDM poses a cost-effective solution based on its mathematical modeling method for finding the best fitting feasible alternative according to decision-makers' preferences [10].

Knowledge about MDD platforms is fragmented in a wide range of literature, documentation, and software engineers' experience. To systematically capture such knowledge and make it available in a reusable and extendable format, we have followed the framework [11] to build a decision model for the MDD platform selection problem. The framework and a decision support system (DSS) [12] were introduced in our previous studies for building decision models for MCDM problems in software production.

The DSS is a platform[1] for building MCDM decision models based on the framework. Decision models can be uploaded to the DSS's knowledge base to facilitate software-producing organizations' decision-making process according to their requirements and preferences. Furthermore, the DSS can be used over the full lifecycle and co-evolve its advice based on evolving requirements.

The rest of this study is structured as follows: Sect. 2 presents a brief description of the MDD platforms and determines the scope of the study. Section 3 formulates the MDD platform selection problem as an MCDM problem, defines the research questions of the study, and describes our research method, which is based on the design science, expert interviews, document analysis, and exploratory theory-testing case studies. This study has the following contributions:

– Section 4 explains the integration of the captured tacit knowledge of software engineers through interviews and the explicit knowledge scattered in an extensive list of websites, articles, and reports. This study's findings provide knowledge that can educate and support the decision-makers to understand: (1) which MDD platforms are available at the moment, (2) the capabilities of the MDD platforms, and (3) which features are fulfilled by which platforms.
– Section 5 describes four industry case studies in the Netherlands to evaluate the effectiveness and usefulness of the decision model to address the decision problem. Moreover, this section analyzes the DSS results and compares them with the case study participants' shortlist of feasible MDD platforms. The results show that the DSS recommended nearly the same solutions as the case study participants suggested to their companies after extensive analysis and discussions and do so more efficiently.

Section 6 highlights barriers to the knowledge acquisition and decision-making process, such as motivational and cognitive biases, and argues how we have minimized these threats to the validity of the results. Section 7 positions the proposed approach in this study among the other MDD platform selection techniques in the literature. Finally, Sect. 8 summarizes the proposed approach, defends its novelty, and offers directions for future studies.

2 Background

Hailpern and Tarr [13] give a general definition of MDD: "A software engineering approach consisting of the application of models and model technologies to raise the level of abstraction at which developers create and evolve software, with the goal of both simplifying (making easier) and formalizing (standardizing, so that automation is possible) the various activities and tasks that comprise the software life cycle."

Software production with an MDD platform is not initiated by programming (high-coding) but modeled using visual modeling or declarative development tools and pre-built templates and components understood by citizen developers. The conceptual model transforms into an application [14], such as web-based or wearable apps, by generating code or model interpretation [2]. The process of transforming a conceptual model into an application is called *conceptual modeling compilation* [15]. The development of conceptual models comprises the real world's representation using an abstraction level higher than that of source code. Likewise, source code represents a higher abstraction level than that for machine code obtained through a conventional compilation process. Therefore, it seems logical to refer to the process of transforming a conceptual model into a software product using the term *compilation*.

The simplified interface leads many to believe that building applications using MDD platforms requires little or no coding knowledge. However, sometimes these predefined components need to be customized using programming languages. For example, maybe a developer wants to place a particular widget on a web application home page, which is not a part of the MDD platform's default widget library. In this case, she needs to extend the platform capabilities by developing a unique widget and making it a new component

[1] The decision studio is available online on the DSS website: https://dss-mcdm.com.

for future projects. Additionally, some MDD platforms offer the flexibility to deploy and maintain applications on public or private clouds or even on-premises. Automated deployment, together with a cloud-native and stateless architecture, leads to high availability and fail-over to support large-scale deployments, especially in an enterprise context.

Four principles underlie the architecture of an MDD platform [16]: (1) Models expressed in a well-defined notation are a cornerstone to system understanding for enterprise-scale solutions. (2) Building systems can be organized around a set of models by imposing a series of transformations between models, organized into an architectural framework of layers and transformations. (3) A formal underpinning for describing models in a set of metamodels facilitates meaningful integration and transformation among models and is the basis for automation through tools. (4) Acceptance and broad adoption of the model-based approach require industry standards to provide consumers with openness and foster competition among vendors. Accordingly, in the literature, there are a significant number of tools and platforms that are based on the MDD paradigm, for instance, modeling notations and software process modeling languages, such as UML and Petri-nets [4].

Gartner [17] and Forrester [18] categorized MDD platforms, including low-code/no-code platforms and business process management systems into the following two sets:

Business Process Management Suite (BPMS) is a set of platforms to support business process management initiatives. BPMS is an MDD approach that aids a process improvement lifecycle from start to end—from process discovery, definition, design, implementation, monitoring, and analysis. BPMS platforms are used for automating, measuring, and optimizing business processes. Note BPMS is an extension of classical workflow management systems and approaches [19].

intelligent Business Process Management Suite (iBPMS) is a subset of low-code/no-code application development platforms. It provides the functionality needed to support more intelligent business operations, such as real-time analytics and collaborative capabilities [17]. According to Gartner's report, iBPMS is the next generation of BPMS that leverages recent technological advances to attain a degree of operational responsiveness not possible with BPMS platforms.

This study focuses on BMPS and iBPMS as two essential sets of MDD platforms to build a decision model for the decision-making process. Note that we use *MDD platforms* to refer to BMPS and iBPMS platforms for the sake of brevity.

3 Research approach

This section defines the problem, indicates the study objective, and formulates the research questions. Moreover, it elaborates on the research methods and relates them to individual research questions to which they apply. Additionally, the knowledge acquisition techniques, analysis procedures, and the tactics used to mitigate the threats to this study's validity are presented in this section.

3.1 Problem definition

In this study, we formulate the MDD platform selection problem as an MCDM problem:

Let $Platforms = \{p_1, p_2, \ldots p_{|Platforms|}\}$ be a set of MDD platforms in the market (i.e., Mendix, OutSystems, and ServiceNow). Furthermore, $Features = \{f_1, f_2, \ldots t_{|Features|}\}$ be a set of MDD features (i.e., Supporting Native modeling tool and Decision table) of the MDD platforms, and each platform p, where $p \in Platforms$, supports a subset of the set $Features$. The goal is finding the best fitting MDD platforms as solutions, where $Solutions \subset Platforms$, that support a set of MDD feature requirements, called $Requirements$, where $Requirements \subseteq Features$. An MCDM approach for the selection problem receives $Platforms$ and their $Features$ as its input, then applies a weighting method to prioritize the $Features$ based on the decision-makers' preferences to define the $Requirements$, and finally employs a method of aggregation to rank the $Platforms$ and suggests $Solutions$. Accordingly, an MCDM approach can be formulated as follows:

$$MCDM : Platforms \times Features \times Requirements \rightarrow Solutions$$

Typically, a unique optimal solution for an MCDM problem does not exist, and it is necessary to use decision-makers' preferences to differentiate between solutions [20]. Particular platforms might fit into an enterprise; however, some might be better than others. It is tough to state which platform is the best one, partially because we cannot predict the future or know how the enterprise would have evolved if a different platform was selected. Furthermore, we must note that such a technology selection process can never be completely objective because humans have to make decisions. Figure 1 visualizes the MCDM approach for the MDD platform selection problem in a 3D space. It shows that the degree of satisfaction of the decision-makers with a suggested solution is fuzzy, which means that the satisfaction degree from a decision-maker perspective may range between completely true (best fit) and completely false (worst fit) [21], which is represented by a range of colors from red to dark green.

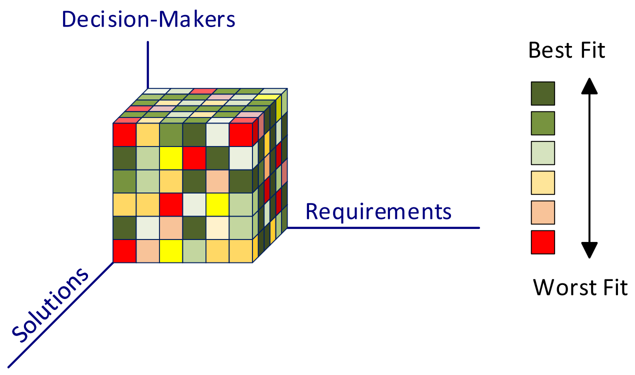

Fig. 1 This figure shows an MCDM approach for the MDD platform selection problem in a 3-dimensional space. Note the degree of the decision-makers' satisfaction with a solution according to their priorities and preferences (requirements) ranges between the best and worst fit solutions, which is represented by a range of colors from red to dark green

3.2 Research question

The main research question (MRQ) of this study is as follows: **MRQ:** How can knowledge regarding MDD platforms be captured and organized systematically to support enterprises with the decision-making process?

We formulated the following research questions to capture knowledge regarding the MDD platform systematically and to build a decision model for the decision problem based on the framework [11]:

RQ_1: Which MDD concepts should be considered as MDD features in the decision model?
RQ_2: Which MDD platforms should be considered in the decision model?
RQ_3: Which software quality attributes can be used to evaluate the MDD platforms?
RQ_4: What are the impacts of the MDD features on the quality attributes of the MDD platforms?
RQ_5: Which MDD platforms currently support the MDD features?

3.3 Research methods

Table 6 shows the research methods and data collection types of a subset of studies in the literature that address the MDD platform selection problem.

We designed a framework [11] and implemented a DSS [12] for supporting software engineers (decision-makers) with their MCDM problems in software production. Knowledge engineering theories have been employed to design and implement the DSS and the framework. The framework provides a guideline for decision-makers to build decision models for MCDM problems in software production following the six-step of the decision-making process [20]: (1) identifying the objective, (2) selection of the features, (3) selection of the alternatives, (4) selection of the weighing method, (5) applying the method of aggregation, and (6) decision-making based on the aggregation results.

In this study, we applied the framework to build a decision model for the MDD platform selection problem. Moreover, we used design science, expert interviews, and document analysis as a mixed data collection method to capture knowledge regarding MDD platforms and to answer the research questions. Then, we identified 30 MDD platforms and 94 MDD features by conducting semi-structured interviews with 26 domain experts. We also indicated the mapping between the MDD features and platforms by analyzing MDD platforms' documents and the experts' tacit knowledge. Moreover, we mapped the MDD features to the quality attributes suggested by ISO/IEC 25010 standard [22] and extended ISO/IEC 9126 standard [23] and calculate the impacts of the MDD platforms on the quality attributes based on three expert interviews. According to the acquired knowledge and guidelines of the framework [11], we modeled the decision problem as an MCDM problem and built a decision model for the MDD platform selection problem. Finally, we have evaluated the decision model by conducting four real-world case studies.

3.3.1 Design science

Design science is an iterative process [24], has its roots in engineering [25], is broadly considered a problem-solving process [26], and attempts to produce generalizable knowledge about design processes and design decisions. The design process is a set of hypotheses that can eventually be proven by creating the artifact it describes [27]. However, a design's feasibility can be supported by a scientific theory to the extent that the design comprises the theory's principles. Research investigations involve a continuous, repetitive cycle of *description*, *explanation*, and *testing* [28]. Accordingly, in most cases, theory development is a process of gradual change [29]. The research approach for creating decision models for MCDM problems is design science, which addresses research by building and evaluating artifacts to meet identified business needs [30].

Recently, we designed a theoretical framework [11] and implemented a DSS [12] for supporting software engineers (decision-makers) with their MCDM problems in software production. Knowledge engineering theories have been employed to design and implement the DSS and the framework. In this study, we applied the framework to build a decision model for the MDD platform selection problem. Additionally, we employed the DSS to facilitate the decision-making process. The research approach for creating the decision model is design science, which addresses research through the building and evaluation of artifacts to meet identified business needs [25]; accordingly, we carried out four industry case studies in the context of four real-world enterprises to evaluate the decision model.

3.3.2 Expert interviews

Twenty-six domain experts from different software-producing organizations have participated in this research to answer the research questions (see Appendix A). *Expert Interview* is an essential knowledge acquisition technique [31] in qualitative research. The primary source of knowledge to build a decision model is domain expertise. Please note that these interviews are different from the interviews we conducted during the case study interviews with the case participants.

We followed Myers' and Newman's guidelines [32] to conduct a series of qualitative semi-structured interviews with senior software engineers to explore expert knowledge regarding MDD Platforms and evaluate the outcomes of our study.

We developed a role description before contacting potential experts to ensure the right target group. Then, we contacted the experts through email using the role description and information about our research topic. The experts were pragmatically and conveniently selected according to their expertise and experience mentioned on their *LinkedIn* profile. We considered a set of expert evaluation criteria (including "Years of experience," "Expertise," "Skills," "Education," and "Level of expertise") to select the experts.

Each interview followed a semi-structured interview protocol (see Appendix B) and lasted between 45 and 60 min. We used open questions to elicit as much information as possible from the experts minimizing prior bias. All interviews were done in person and recorded with the interviewees' permission and then transcribed for further analysis.

Acquired knowledge during each interview was typically propagated to the next to validate the captured knowledge incrementally. Finally, the findings were sent to the interview participants afterward for final confirmation. Note, for the validity of the results, the research's data collection phases were not affected by the case study participants; moreover, none of the interviewees or researchers were involved in the case studies.

Please note that we did not use formal coding for the analysis of the interviews and the literature. What we did do, however, could be termed incremental concept development. During the literature study and interviews, concepts were identified that were relevant. Candidate qualities and features were identified, defined, and fine-tuned with the interviewees and later confirmed by asking the interviewees for post-analysis of the interview and literature results. While this did not constitute formal coding, we did mark concepts related to the domain, came up with the literature study, and came up with the interviews. Secondly, these concepts were incrementally fine-tuned until an agreement was reached with the interviewees.

3.3.3 Document analysis

Document analysis is a systematic procedure for reviewing or evaluating documents, including text and images that have been recorded without a researcher's intervention [33]. Document analysis is one of the analytical methods in qualitative research that requires data investigation and interpretation to elicit meaning, gain understanding, and develop empirical knowledge [34]. We reviewed webpages, whitepapers, scientific articles, fact sheets, technical reports, product wikis, product forums, product videos, and webinars to map the platforms' MDD features.

One of the principal challenges in the document analysis is the lack of standard terminology among MDD platforms [13]. Sometimes different MDD platforms refer to the same concept by different names, or even worse, the same name might stand for different concepts in different MDD platforms. Discovering conflicts is essential to prevent semantic mismatches throughout the knowledge extraction phase.

We followed the framework to capture a conceptual phrase representing a segment of data related to a particular research question. In other words, we used a conceptual mapping of MDD concepts to identify potential conflicts in knowledge sources. For instance, conceptually similar phrases and definitions regarding MDD platforms' features were collected together for more detailed analysis. Based on the framework, document analysis and conceptual mapping were employed to extract knowledge from the selected sources of knowledge and prevent semantic mismatches.

We defined an extract extraction form to obtain consistent extraction of relevant knowledge and checked whether the acquired knowledge would address the research questions. The extracted knowledge, which correspond to the research questions, has been classified into five categories: *quality attributes*, *MDD platforms*, *MDD features*, *impacts of the MDD features on the quality attributes*, and *supportability of the MDD features by the MDD platforms*. Next, the extracted knowledge was employed to build a decision model for the MDD platform selection problem. Finally, the decision model was uploaded to the knowledge base of the DSS.

3.3.4 Case study

Case study is an empirical methodology that investigates a phenomenon within a particular context in the domain of interest [35]. A case study can be employed to collect data regarding a particular phenomenon or to apply a tool and evaluate its efficiency and effectiveness using interviews. Yin [35] distinguishes four types of case study designs according to holistic versus embedded and single versus multiple. In this study, we employ holistic multiple case designs: examining multiple real-world companies' cases within their context to learn more about one specific unit of analysis and evaluating the decision model for the MDD platform selection problem. To conduct the case studies and evaluate the proposed decision model, we followed the following *case study protocol*:

1. *Step 1: Requirements elicitation* At least two managers or team leaders of the case study companies' IT departments should participate in the research, as such participants are thoroughly informed on the design decisions and the requirements of their decision context. During the interview session with each company's case study participants, we first ask them to explain the decision context and requirements. Next, we show the MDD feature list to the participants and explain the features completely. Afterward, we ask the participants to identify their MDD feature requirements and prioritize them based on the MoSCoW prioritization technique [36]. Additionally, they have to express the rationales behind the requirements elicitation. Finally, they should identify a set of MDD platforms as potential solutions for their software projects.
2. *Step 2: Results and recommendations* We define four separate cases in the DSS knowledge base according to the case studies' requirements and priorities. Next, the DSS suggests a set of feasible solutions per case individually. Then, the outcomes are discussed with the case study participants.
3. *Step 3: Analysis* We compare the feasible solutions from the DSS with the case study participants solutions. Furthermore, we analyze the outcomes and observations, report them to the case study participants, and subsequently receive their feedback on the results.

We ensured validity in the conversations with the case participants in the following ways. First, we made sure that all terms we used in the discussions were known by providing the list of features and qualities and their definitions, and discussing these terms with the participants during the interviews. Secondly, the discussions with the case participants were noted down by the researchers during the interviews, and these were processed within 24 h to ensure that none of the results would be forgotten. We did not record the meetings to avoid tension during these discussions. Finally, we confirmed the results from the decision support model with the case participants and discussed whether our inputs into the decision support tool were correct.

We have considered other study designs, such as action research studies, to support the engineers during their selection. However, by performing these case studies post hoc, we could ensure that the case participants had used their selected platforms for a more extended period of time. Another possibility would have been to survey end-users of MDD Platforms. However, as we were particularly interested in how our model and tool were used, performing multiple case studies at companies provided us with the highest level of detail for the empirical results.

4 MCDM for MDD platform selection

We follow the framework [11] as modeled in Fig. 2 to build a decision model for the MDD platform selection problem. Generally speaking, a decision model for an MCDM problem contains decision criteria, alternatives, and mappings. Figure 2 represents the main building blocks of the decision support system besides the proposed decision model.

4.1 RQ_1: MDD features

Domain experts were the primary source of knowledge to identify the right set of MDD features, even though documentation and literature studies of MDD platforms can be employed to develop an initial hypothesis about the MDD feature set. Each MDD feature has a data type, such as *Boolean* and *non-Boolean*. For example, the data types of MDD features, such as the *popularity in the market* and supportability of *real-time analytics*, can be considered as *non-Boolean* and *Boolean*, respectively.

The initial set of MDD features was extracted from the following sources: White papers, Fact sheets, Technical reports, Instruction manuals, Product wikis, Product forums, Product videos, Webinars. Additionally, twenty-three domain experts have participated in this phase of the research to identify a potential list of MDD features. Accordingly, 90 Boolean and four non-Boolean MDD features were identified and extracted from the expert interviews' results. Eventually, the validity and reliability of the final list of the features[2] were evaluated and confirmed by the domain experts.

4.2 RQ_2: MDD platforms

To answer the second research question, we identified 104 MDD platforms as our initial hypothesis based on the following three methods:

[2] The entire list of the MDD features and supportability of considered MDD platforms are available and accessible on the data repository [37].

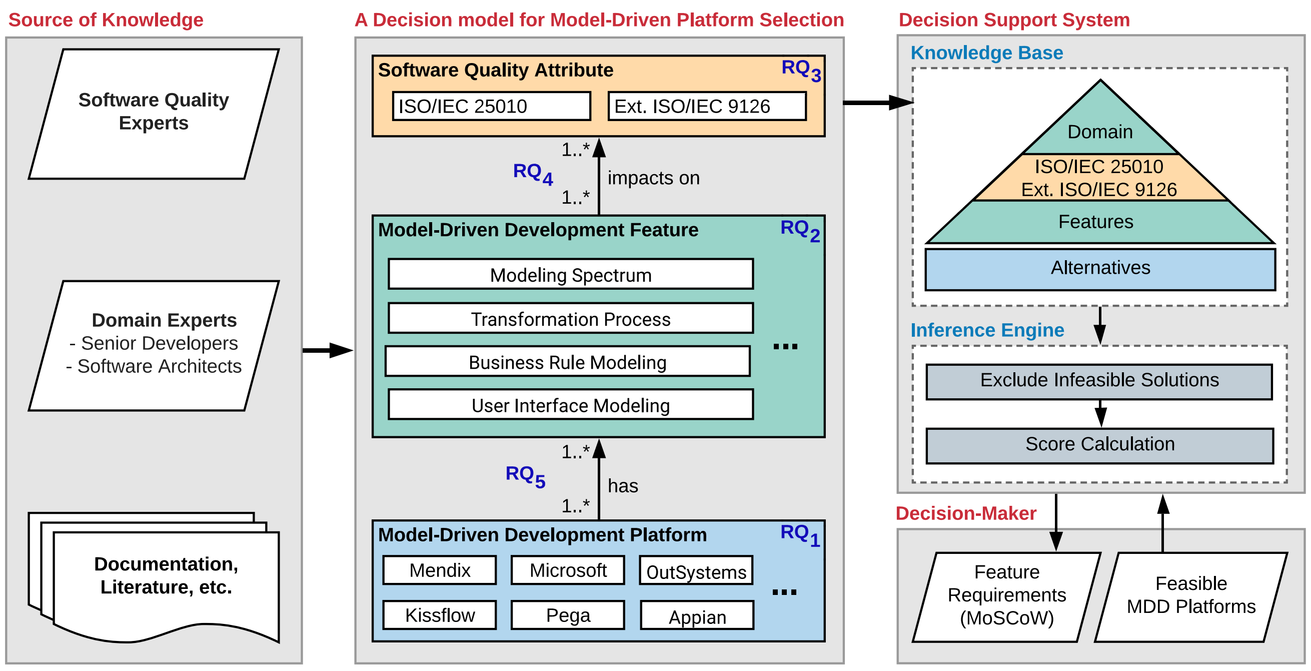

Fig. 2 This figure is adapted from our previous study [11] and shows the main building blocks of the decision support system besides the proposed decision model for the MDD Platform selection problem

– Exploring literature with the keywords "Low-Code," "No-Code," "BPMS," "iBPMS," and "Model-Driven Development" platforms.
– Exploiting our network of domain experts, including software engineers and academics. Note we conducted 26 expert interviews.
– Asking interviewed domain experts at the end of each interview whether they know of products that should be researched.

Next, we reviewed the published surveys and reports from well-known knowledge bases, such as Gartner [7,17] and Forrester [18]; eventually, we selected 30 MDD platforms that at least three sources of knowledge confirmed the necessity of their existence in the decision model. The first row of Table 1 shows the list of the selected MDD platforms.

4.3 *RQ*$_3$: software quality attributes

Based on the IEEE Standard Glossary of Software Engineering Terminology [38,39], the quality of software products is the degree to which a system, component, or process meets specified requirements (such as functionality, performance, security, and maintainability) and the extent to which a system, component, or process meets the needs or expectations of a user. It is necessary to find quality attributes widely recommended by other researchers to measure the system's characteristics.

The literature study results approved that researchers do not agree upon a set of conventional criteria, including quality attributes and features, to evaluate the MDD platforms (see Table 6). Additionally, we realized that their suggested criteria were mainly applied to specific domains to address different research questions. Consequently, a set of non-exclusive and domain-independent criteria is needed to evaluate MDD platforms.

The ISO/IEC 25010 [22] model presents best practice recommendations on the base of a quality assessment model. The quality model defines which quality characteristics should be considered when assessing the qualities of a software product.

A set of quality attributes should be defined in the decision model [11]. In this study, we used the *ISO/IEC 25010* standard [22] and *extended ISO/IEC 9126* standard [23] as two domain-independent quality models to analyze MDD features based on their impact on quality attributes of MDD platforms. The key rationale behind using these software quality models is that they are a standardized way of measuring a software product. Moreover, they describe how easily and reliably a software product can be used.

The last four columns of Table 6 show the results of our analysis regarding the common criteria and alternatives of this study with the selected publications. Let us define the coverage of the i-th selected study as follows:

$$Coverage_i = \frac{CQ_i + CF_i}{C_i} \times 100$$

where CQ_i and CF_i denote the numbers of common quality attributes (column #CQ) and features (column #CF) of the i-th selected study; furthermore, C_i signifies its number of suggested criteria. The last column (*Cov.*) of Table 6 indicates the percentage of the coverage of the considered criteria within the selected studies. On average, 75% of those criteria are already considered in this study.

4.4 *RQ*$_4$: the impacts of MDD features

The mapping between the sets *software quality attributes* and *MDD platforms* was identified based on domain experts' knowledge. Three domain experts participated in this phase of the research to map the MDD platforms ($Features$) to the quality attributes ($Qualities$) based on a Boolean adjacency matrix[3] ($Qualities \times Features \rightarrow Boolean$). For instance, *entity–attribute–relationship (EAR)* as an MDD feature influences the *functional correctness* quality attribute. The framework does not enforce an MDD feature to present in a single quality attribute; MDD features can be part of many quality attributes. For example, *native modeling tool* as an MDD feature might be linked to multiple quality attributes such as *resource utilization* and *functional appropriateness*. The experts believed that about 74% percent of the MDD features impact the following key characteristics of the MDD platforms:

– *Functional suitability* is defined in ISO/IEC 25010 as the degree to which an MDD platform functions that meet the stated or implicit requirements when used under specific conditions.
– *Usability* defines the degree to which an MDD platform can be used to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use. Moreover, it embraces quality attributes such as "Learnability," "Operability," "User error protection."
– *Maintainability* is the degree to which an MDD platform can be effectively and efficiently modified without introducing defects or degrading existing product quality. For instance, the experts believe that "Modularity," "Reusability," "Analyzability," "Modifiability," and "Testability" can be considered as key strengths of the MDD platforms that support "Model only" or "Model-centric" features.
– *Supplier* includes a set of quality attributes such as "Reputation" and "Support" of the MDD platforms.
– *Cost* denotes the amount of money that a company spends on implementing a software product using an MDD platform. It includes quality attributes such as "Implementation Cost," "Platform Cost," and "Licensing Costs."
– *Product* defines a set of quality attributes regarding the state or fact of exclusive rights and control over the property. For instance, "Stability," "Ownership," and "Guarantees" are part of this characteristic.

[3] The acquired knowledge regarding the impacts of the MDD platforms on the quality attributes was used to calculate the impact factors [11] that apply in the score calculation of the DSS. The final Boolean adjacency matrix is available online on the data repository [37].

4.5 *RQ*$_5$: supportability of the MDD features

An MDD platform contains a set of MDD features that can be either Boolean or non-Boolean. A Boolean MDD feature ($Feature^B$) is a feature that is supported by the MDD platform, for example, supporting the *Native modeling tool*. A non-Boolean MDD feature ($Feature^N$) assigns a non-Boolean value to a particular MDD platform; for example, the popularity in the market of an MDD platform can be "high," "medium," or "low." Accordingly, this study's MDD features is a collection of Boolean and non-Boolean features, where $Features = Feature^B \cup Feature^N$.

The mapping $BFP : Feature^B \times Platforms \rightarrow \{0, 1\}$ defines the supportability of the Boolean MDD features by the platforms. So that $BFP(f, p) = 0$ means that the platform p does not support the MDD feature f or we did not find any evidence for proof of this feature's supportability by the MDD platform. Moreover, $BFP(f, l) = 1$ signifies that the platform supports the feature. The mapping *BFP* is defined based on documentation of the MDD platforms and expert interviews. Tables 1 and 2 present the Boolean features that we have considered in the decision model.

The experts defined four non-Boolean MDD features, including "Popularity in the market," "active community," "Maturity level of the company," and "future roadmap." The assigned values to the non-Boolean MDD features for a specific MDD platform are based on a 3-point Likert scale (**h**igh, **m**edium, and **l**ow), where $NFP : Features^N \times Platforms \rightarrow \{H, M, L\}$, based on several predefined parameters. For instance, the "popularity in the market" of an MDD platform was defined based on the following parameters: the number of the "Google hits," "Google Trends (Means of the past 12 months)," "Twitter (follower)," and the popular forums and reports that considered the platform in their evaluation. Table 3 shows the non-Boolean programming features, their parameters, and sources of knowledge.

4.6 MDD feature requirements

The DSS [12] receives the MDD feature requirements based on the *MoSCoW* prioritization technique [36].

Table 1 This table shows the first part of the Boolean features ($Feature^B$), MDD platforms ($Platforms$), and the "BFP" mapping

Boolean features MDD platforms (BFP)	Coverage	Outsystems	Mendix	Appian	Salesforce (Lightning)	Microsoft PowerApps	Kissflow	Zoho (Creator)	ServiceNow (Now Platform)	Oracle APEX	Pega (Infinity)	Google App Maker	Quick Base	TrackVia	WaveMaker	Kony (Quantum)	GeneXus	Codeless Platforms	IBM (Digital App Builder)	Software AG (AgileApps Live)	Betty Blocks	Servoy	42windmills	Thinkwise	Usoft	Triggre	Aptean	OpenText (AppWorks 16)	WEM Modeler	GorillaIT (Karooda Platform)	AgilePoint (NX)
Platform Types																															
General-purpose platform	68.97%	1	1	0	1	1	0	1	0	1	1	1	1	0	1	1	1	1	1	0	1	1	0	1	0	1	0	0	1	1	1
Process app platform	20.69%	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	1	1	0	0	0
Mobile app platform	100.00%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Request-handling platform	13.79%	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0
Modeling spectrum																															
Model-centric (Low-code)	93.10%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	1	1	1	1	1	1	1	1
Model only (No-code)	51.72%	0	1	1	1	1	1	0	0	0	1	0	0	1	1	1	0	0	0	1	1	0	1	0	0	1	0	0	0	1	1
Developer Citizen																															
Domain Experts	68.97%	0	1	1	1	1	1	0	0	0	1	0	0	1	1	1	0	0	0	1	1	0	1	1	1	1	1	1	1	1	1
Business Analysts	100.00%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Professional Developers	65.52%	1	1	0	1	1	0	1	1	1	1	1	1	0	1	1	1	1	1	0	1	1	1	0	0	0	0	0	0	1	1
Development																															
Visual IDE	100.00%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Multi-channel/Cross-platform Application	65.52%	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	0	1	1	0	1	0	1	0	0	1	0	0	0	0
Programming mandatory	31.03%	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	1	1	0	0
Programming optional	65.52%	1	1	1	1	1	1	0	1	0	1	0	1	1	1	1	1	0	0	1	1	0	0	0	1	1	1	0	0	1	1
Integration	89.66%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	0	0
Cross-Platform Integration	79.31%	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	0	1	1	1	1	0	0	0	0	1	1	1	1	1
Integrate with an ERP system	55.17%	0	1	1	0	0	1	1	0	1	1	0	0	1	0	0	1	1	1	0	1	1	0	1	0	0	0	1	1	0	1
Data mapping	72.41%	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	0	0	1	1	1	1	1	0	0	1	0	0	0
REST	82.76%	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	0	1	1	0	1	1	1	0	0
SOAP	68.97%	1	1	1	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	0	1	1	0	1	1	0	1	0	1	0	0
OData services	41.38%	1	1	0	1	1	0	1	1	1	1	0	1	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	1	0	0
Importing data	68.97%	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	1	0	0	1	1	1	1	1	0	0	1	0	0	0
Deployment	100.00%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Public Cloud platform	100.00%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Private Cloud platform	37.93%	1	1	1	0	1	0	0	0	1	0	0	1	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
On-premise	72.41%	1	1	1	1	1	1	1	1	1	1	0	1	0	1	1	1	1	1	1	1	1	0	0	1	0	0	1	0	0	1
Access control	100.00%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Identity and permissions management	100.00%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Multi-factor authentication (MFA)	51.72%	1	1	1	1	1	1	1	0	1	1	0	1	0	0	1	1	0	1	0	1	1	0	1	0	0	0	0	0	0	0
OAuth	75.86%	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	0	1	1	0	0	0	0	1	1	1	0	1
Single Sign-On (SSO)	75.86%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	0	0	0	1	0	0	1	1	1	0	1
Data Management	93.10%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	0	1
SQL or NoSQL databases	65.52%	1	1	1	1	1	0	1	1	1	1	1	1	0	1	1	1	0	1	0	1	1	0	1	1	0	1	0	0	0	0
Web API	86.21%	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	0	1
Service Calls	72.41%	1	1	1	1	1	0	0	1	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	0	0
Local application-specific databases	44.83%	1	1	1	1	1	0	1	1	1	1	0	0	0	1	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0
Connectors to various back-ends or services	55.17%	1	1	1	1	1	0	1	1	1	1	0	0	0	1	1	1	1	1	0	0	1	0	1	0	0	0	0	1	0	0
Real-time Analytics	68.97%	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	1	0	0	1
Report and Analytics	68.97%	1	1	1	1	1	1	1	1	1	1	0	1	1	0	0	1	1	1	1	1	0	0	1	0	0	1	1	0	0	1
Application lifecycle manager	82.76%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	0	1	1	0	0
user stories	41.38%	1	1	1	1	0	1	1	1	1	1	0	0	0	0	0	1	0	0	0	1	1	0	0	1	0	0	0	0	0	0
Built-in team collaboration	82.76%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	0	1	1	0	0
sandbox-to-production phases	34.48%	1	1	0	0	1	0	0	1	1	1	0	1	0	0	0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0
component catalogue	37.93%	1	1	0	0	1	0	0	1	1	0	0	1	0	1	0	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0
Application and portfolio management	51.72%	1	1	1	1	1	1	1	1	1	1	0	1	0	1	0	1	0	1	1	0	1	0	0	0	0	0	0	0	0	0
Version control	48.28%	1	1	1	1	1	1	1	1	1	1	0	1	0	1	0	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0
Administrative controls	44.83%	1	1	1	1	1	0	1	1	1	1	0	1	0	1	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0
User Interface																															
Tool set	62.07%	1	1	1	1	1	0	0	1	1	0	1	1	0	1	1	1	1	1	1	1	1	0	1	0	0	1	0	0	0	0
Form & View	100.00%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Predefined components	100.00%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Multilingual Apps	62.07%	1	1	1	1	1	1	1	1	1	1	1	0	0	0	1	1	0	1	1	1	1	0	0	1	0	0	0	1	0	0
Company-Branded Templates & Styling	72.41%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	1	0	0	0
Create extensions and widget libraries	58.62%	1	1	0	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0
Data Modeling																															
Entity-Attribute-Relationship (EAR)	86.21%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	1	1	1	1	0	1
Object-Role Modeling (ORM)	20.69%	0	1	0	1	1	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0
Ontology modeling	13.79%	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
Business rule modeling																															
Data rules	96.55%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
Process rules	93.10%	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
Decision table	41.38%	1	1	1	1	1	1	1	1	1	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0
Decision tree	44.83%	1	1	1	1	1	1	1	1	1	1	0	1	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0
Expression modeling																															
Expression editor	79.31%	1	1	1	1	1	1	1	1	1	1	0	1	1	1	0	1	0	1	1	1	1	1	1	1	0	0	1	1	0	1
Natural language rules	27.59%	0	0	0	1	0	0	1	1	0	1	0	0	0	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
Close to programming language	37.93%	1	1	0	0	1	0	0	1	1	0	0	1	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
Programmed	27.59%	1	0	0	0	1	0	1	0	0	0	0	1	0	0	0	1	0	0	0	0	1	1	1	0	1	0	0	0	0	0

Note 1s on each row indicate that the corresponding platforms support the MDD feature of that row, and 0s signify the corresponding platforms do not support that feature, or we did not find any strong evidence of their supports based on the documentation analysis. Moreover, the rows in black indicate the categories of the features, and the rows in blue show the features, and the rows below them are their subfeatures. The definitions of the features are available on the data repository [37]

Table 2 This table shows the second part of the Boolean features ($Feature^B$), MDD platforms ($Platforms$), and the “BFP” mapping

Boolean features MDD platforms (BFP)	Coverage	Outsystems	Mendix	Appian	Salesforce (Lightning)	Microsoft PowerApps	Kissflow	Zoho (Creator)	ServiceNow (Now Platform)	Oracle APEX	Pega (Infinity)	Google App Maker	Quick Base	TrackVia	WaveMaker	Kony (Quantum)	GeneXus	Codeless Platforms	IBM (Digital App Builder)	Software AG (AgileApps Live)	Betty Blocks	Servoy	42windmills	Thinkwise	Usoft	Triggre	Aptean	OpenText (AppWorks 16)	WEM Modeler	GorillaIT (Karooda Platform)	AgilePoint (NX)
Model Transformation																															
Modeling Tool	100.00%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Native modeling tool	48.28%	1	1	1	1	1	0	0	1	1	0	0	0	0	0	0	1	1	0	1	1	1	0	0	1	0	0	1	0	0	1
Web modeling tool	79.31%	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0	1	0	1	1	0	1	1	0	1	1	1
Engine	100.00%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Code Generation	55.17%	0	1	0	1	1	0	1	1	0	0	0	1	0	1	1	1	1	0	0	0	1	1	0	0	1	1	0	0	1	1
Model interpretation	44.83%	1	0	1	0	0	1	0	0	1	1	1	0	1	0	0	0	0	1	1	1	0	0	1	1	0	0	1	1	0	0
Storage	82.76%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	0	1	1	1	1	1	1	0	0	1	0	1
XML/JSON as data storage	68.97%	1	1	1	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	0	0	1	0	1	1	0	0	0	1	0	1
Store model locally	51.72%	1	1	1	1	1	0	1	1	0	0	0	0	1	1	0	1	1	1	0	1	0	1	0	0	1	0	0	0	0	1
Portability	65.52%	1	1	1	1	1	0	1	1	1	1	1	0	0	1	0	1	1	1	0	1	1	1	0	0	1	0	1	1	0	0
Convert model to text	31.03%	1	1	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	1	0	1	0	0	1	0	1	1	0	0
Support different stacks	34.48%	1	1	0	0	1	0	1	1	1	0	1	0	0	1	0	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0
Plug and play architecture	37.93%	1	1	1	1	0	0	1	1	1	1	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0
Business Process Automation																															
Workflow	100.00%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Process flow	93.10%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	1	1	1	1	1
Case flow	44.83%	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	1	1	0	0	0
Application Types																															
Mobile Apps	96.55%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
Web portals	100.00%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Web applications	100.00%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Smartwatch (wearable) Apps	13.79%	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Security and Protection (Compliance)																															
HIPAA compliant	51.72%	0	1	1	1	1	1	1	0	0	1	1	1	1	1	1	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0
ISO 27001-2013 certification	65.52%	1	1	1	1	1	1	1	1	0	1	1	0	1	0	1	0	0	1	1	1	0	0	0	1	0	1	1	1	0	1
SOC 1	26.67%	0	1	1	0	1	0	0	1	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0
SOC 2	56.67%	1	1	1	0	1	1	1	1	0	1	1	1	1	0	1	0	0	1	1	0	0	0	0	0	0	1	1	0	0	1
SOC 3	33.33%	0	1	1	0	1	1	0	0	0	1	0	0	1	0	0	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0
PCI DSS	34.48%	1	1	1	1	1	0	1	0	0	1	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0
GDPR	86.21%	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	0	1	1	0	1	1	1	0	1

Note 1s on each row indicate that the corresponding platforms support the MDD feature of that row, and 0s signify the corresponding platforms do not support that feature, or we did not find any strong evidence of their supports based on the documentation analysis. Moreover, the rows in black indicate the categories of the features, and the rows in blue show the features, and the rows below them are their subfeatures. The definitions of the features are available on the data repository [37]

Decision-makers should prioritize their MDD feature requirements using a set of weights ($W_{MoSCoW} = \{w_{Must}, w_{Should}, w_{Could}, w_{Won't}\}$) according to the definition of the *MoSCoW* prioritization technique. MDD feature requirements with *Must-Have* or *Won't-Have* priorities act as hard constraints and MDD feature requirements with *Should-Have* and *Could-Have* priorities act as soft constraints. So that, the DSS excludes all infeasible MDD platforms which do not support MDD features with *Must-Have* and support MDD features with *Won't-Have* priorities. Then, it assigns nonnegative scores to feasible MDD platforms according to the number of MDD features with *Should-Have* and *Could-Have* prioritizes [11].

Decision-makers specify desirable values, according to their preferences, for non-Boolean MDD feature requirements. For example, a decision-maker could be interested in prioritizing MDD platforms with the *Maturity level* above average. Therefore, the *maturity level of the company* above average is a *Should-Have* feature.

5 Empirical evidence: the case studies

Four industry case studies in the context of four real-world enterprises have conducted to evaluate and signify the decision model's usefulness and effectiveness to address the MDD platform selection problem. To increase diversity in our evaluation, we selected the case studies from different domains, such as workflow management, project management, and enterprise resource planning (ERP) systems. Furthermore, one of the case studies was a software consultancy company working in close collaboration with Mendix and was interested in evaluating other potential alternatives. We had a session with the decision-makers at each case study company to capture their functional and quality requirements, constraints and assumptions, and technology acceptance criteria. They explained their concerns and barriers, such as their budget or lack of time and technical knowledge in the development team to build the desired soft-

Table 3 This table shows the NFP mapping between the non-Boolean MDD features and platforms

Non-Boolean features- MDD platforms (NFP)	Outsystems	Mendix	Appian	Salesforce (Lightning)	Microsoft PowerApps	Kissflow	Zoho (Creator)	ServiceNow (Now Platform)	Oracle APEX	Pega (Infinity)	Google App Maker	Quick Base	TrackVia	WaveMaker	Kony (Quantum)	GeneXus	Codeless Platforms	IBM (Digital App Builder)	Software AG (AgileApps Live)	Betty Blocks	Servoy	42windmills	Thinkwise	Usoft	Triggre	Aptean	OpenText (AppWorks 16)	WEM Modeler	GorillaIT (Karooda Platform)	AgilePoint (NX)	Source of knowledge
Popularity in the market	H	H	H	H	H	H	H	H	H	H	M	M	M	M	M	M	M	M	L	L	L	L	L	L	L	L	L	L	L	L	Domain experts
Google hits	13.500	12.800	8.960	39.700	2.780	18.300	45.700	9.030	36.300	8.840	2.240	3.910	8.010	2.420	3.700	710	8.520	2.600	5	3.470	802	3	301	6	11	50	15	33	2	1.620	Query string: "Product Name"+ "Low code"+ "No code"
Google Trends (Means of the past 12 months)	80,84	82,78	67,67	82,61	65,14	66,96	85,49	76,14	89,20	67,55	72,78	16,63	58,69	47,51	53,35	77,69	66,61	81,00	65,41	36,71	53,88	0,00	0,00	34,76	0,00	55,53	29,53	31,92	0,00	0,00	www.trends.google.com
Twitter (follower)	23.9K	10.8K	15,9K	483.8K	23.3K	3,5K	55,3K	32,2K	6,2K	47.7K	23	5,4K	3,1K	18,3K	11,6K	6,7K	4,5K	4,5K	17	3,9K	1K	849	290	220	86	1,3K	2K	138	9	15K	www.twitter.com
Gartner	1	1	1	1	1	0	1	1	1	1	0	1	1	0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	www.gartner.com
The Forrester Wave	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	www.forrester.com
softwaretestinghelp (2020)	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	www.softwaretestinghelp.com
pcmag	1	1	1	1	1	0	1	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	www.pcmag.com
TrustRadius	1	1	1	0	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	www.trustradius.com
G2	1	1	1	1	1	1	1	1	1	1	0	0	1	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	www.g2.com
predictiveanalyticstoday	1	0	0	0	0	0	0	0	1	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	www.predictiveanalyticstoday.com
featuredcustomers	1	1	1	0	1	1	1	0	0	0	0	1	1	1	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	www.featuredcustomers.com
apriorit	1	1	1	1	1	1	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	www.apriorit.com
objectivity	1	1	0	1	1	0	0	1	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	www.objectivity.co.uk
altitudemarketing	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	www.altitudemarketing.com
Active Community	H	H	H	H	H	L	M	H	H	H	H	M	L	L	M	L	L	H	L	L	L	L	L	L	L	H	L	L	L	L	Domain experts
LinkedIn (job openings)	149	132	170	3K	8,7K	1	6	1,4K	614	913	4K	10	14	3	17	N/A	N/A	8,9K	N/A	3	20	N/A	64	74	N/A	80	N/A	N/A	N/A	N/A	www.linkedin.com
LinkedIn (Followers)	56K	20,4K	29,1K	2M	10,15M	5,3K	255,7K	277K	5M	191,65K	16,63M	7,9K	2,5K	2,1K	40,2K	9,2K	2,9K	7,97M	N/A	5,4K	1K	48	2,1K	497	211	20,20K	3,1K	N/A	157	1,2K	www.linkedin.com
Glassdoor (jobs)	132	157	280	1.9K	10K	N/A	15	4,2K	11K	646	6,8K	44	15	N/A	9	N/A	7	17K	N/A	23	2	N/A	27	N/A	N/A	116	N/A	N/A	N/A	6	www.glassdoor.com
Maturity level of the company	H	M	H	H	H	L	H	H	H	H	H	M	M	M	H	M	M	H	L	M	M	M	M	M	L	H	M	L	M	M	Domain experts
Founded	2001	2005	1999	1999	1975	2012	1996	2004	1977	1983	1998	1999	2006	2014	2007	1988	1988	1911	2013	2012	2001	2009	2002	1987	2013	2012	2001	N/A	2001	2003	www.glassdoor.com
Number of Employees	1001 to 5000	501 to 1000	1001 to 5000	10000+	10000+	101 to 250	201 to 500	5001 to 10000	10000+	1001 to 5000	10000+	201 to 500	51 to 200	51 to 200	1001 to 5000	201 to 500	51 to 200	10000+	N/A	51 to 200	51 to 200	11 to 50	51 to 200	11 to 50	11 to 50	1001 to 5000	501 to 1000	N/A	11 to 50	51 to 200	www.glassdoor.com www.crunchbase.com www.linkedin.com
Type	Private	Private	Public	Public	Public	Private	Private	Public	Public	Public	Public	Private	Private	Subsidiary	Private	Private	Private	Public	Private	Private	Private	Private	Private	Private	Private	Private	Private	Private	Private	Private	www.glassdoor.com www.crunchbase.com www.linkedin.com
Revenue per year	$100 to $500 M	$100 to $500 M	$100 to $500 M	$10+ B	$10+ B	N/A	$100 to $500 M	$2 to $5 B	$10+ B	$500 M to $1 B	$10+ B	N/A	18,6 M	15,9 M	$100 to $500 M	$5 to $10 M	N/A	$10+ B	N/A	$5 to $10 M	N/A	N/A	N/A	N/A	N/A	$100 to $500 M	N/A	N/A	N/A	$10 to $25 M	www.glassdoor.com www.crunchbase.com www.linkedin.com
Future Roadmap	H	H	M	H	H	L	L	H	H	L	H	L	M	H	L	M	L	H	L	M	L	L	L	M	L	M	H	L	L	L	Domain experts
Research and Development department	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	0	1	1	1	1	1	0	0	0	Product website
science-industry collaboration	1	1	1	1	1	0	0	1	1	0	1	0	1	1	0	1	0	1	0	1	0	1	0	1	0	1	1	0	0	0	Product website
Co-publications in peer reviewed journals	1	1	0	1	1	0	0	1	1	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	Web of Science

Note the *popularity in the market*, *active community*, *maturity level of the company*, and *future roadmap* are the non-Boolean MDD features that were considered in this study. The parameters of these features are listed below each feature, for example, *founded*, *number of Employees*, *type*, and *revenue per year* are the parameters of the *maturity level of the company*

ware product. Then, we showed the feature list[4] besides their explanations [37] and asked them to identify their feature requirements based on the MoSCoW prioritization technique (Table 4).

Note the case study participants have identified several potentially feasible MDD platforms for their projects through multiple internal expert meetings and investigation into MDD platforms before participating in this research (see the CSP row in Table 5). Four industry cases were defined based on the MoSCoW prioritization technique and stored in the DSS knowledge base. Next, the inference engine of the DSS generated feasible solutions for each case. The rest of the section describes the case study companies' shortlists and analyzes the DSS outcomes.

5.1 Case study 1: Nederlandse Spoorwegen (NS)

Nederlandse Spoorwegen (Dutch: NS; English: Dutch Railways) is a Dutch state-owned company, the principal passen-

[4] The Boolean and non-Boolean features are presented in Tables 1, 2, and 3.

ger railway operator in the Netherlands. Founded in 1938 and with more than 4500 employees, NS provides rail services on the Dutch leading rail network. The Information and Communications Technology (ICT) department of NS has more than 400 employees and is responsible for monitoring the IT projects' short-term and long-term progress.

The experts at the ICT department implemented a workflow management system called NOVA, based on the Mendix platform, to enable users to define different workflows for different types of jobs or processes in the context of IT projects. The case study participants stated that NOVA mappings out the IT projects' workflows in an ideal state; it finds redundant tasks, automates the processes, and identifies bottlenecks or areas for improvement.

The case study participants stated that they want to evaluate a shortlist of feasible MDD platforms that they came up with through an extensive investigation into potential alternatives. They considered Mendix and Microsoft PowerApps as two MDD platforms before participating in this research. The case study participants mentioned that the third alternative for them could be "High-Coding" or implementing a workflow management system by hiring a team of senior software engineers and architects.

5.1.1 NS requirements

The case study participants at NS defined a subset of the key functionality of NOVA as follows:

- It gets a clear overview of work in progress in a particular workflow (R17).
- It allows creating, prioritizing, assigning tasks, divides tasks into workflow stages, decides who works on what part, and allows monitoring work moving through multiple stages (R1, R16, R17).
- It enables centralized governance, so creating task lists, adding tasks/subtasks, subscribing to the entire tasklist, assigning tasks/subtasks to one or multiple people, logging activities, and scheduling events can be managed and monitored in one place (R5, R6).
- It creates custom roles and grants access to people based on their responsibilities (R1).
- It provides a dynamic structure for executing non-routine unpredictable business processes that require coordination of multiple tasks and complex decision-making (R8, R16).
- It has a policy access-control mechanism that restricts access to authorized users and defines users' roles and privileges (R1).

The participants expressed that the MDD platform should be popular enough in the market (R24), as popularity indicates that an MDD platform is widely purchased by other businesses and has an active community (R58). Furthermore, they believed that the maturity level of the MDD platform is an essential criterion that assesses the efficiency and effectiveness of an MDD platform (R9). Additionally, the potential MDD platform should deploy the workflow management system on a private cloud due to higher security, flexibility, and availability (R6).

The case study participants identified their feature requirements based on the MoSCoW prioritization technique (see Table 4). Then, we defined a case according to their requirements and priorities in the DSS knowledge base.

5.1.2 Results and analysis

The case study participants at NS identified 40 MDD feature requirements that more than half of them prioritized as "Must-Have" features (see Table 5). The DSS excluded 26 MDD platforms from the 30 platforms in its knowledge base and offered four potential MDD platforms to NS. Table 5 shows that Mendix was the first feasible platform for NS. Additionally, Oracle APEX, Microsoft PowerApps, and OutSystems were scored as the second to fourth potential solutions.

The case study participants were looking for a low-code or no-code platform to prioritize "Model-only" (R20) and "Model-centric" (R10) as two Should-Have features. Based on our assessment, Mendix and Microsoft PowerApps support both of these features. "Ontology modeling" (R27) as a Should-Have feature is only supported by Oracle APEX. Moreover, "Plug and play architecture" as another Should-Have feature does not support by Microsoft PowerApps.

The experts who participated in this case study expressed that NOVA is currently based on Mendix as they had some legal limitations to select a suitable platform that meets their requirements. NS is a semi-government organization that should follow some government bureaucracy; for instance, they have to deploy NOVA on a national platform that uses a local data center inside the Netherlands borders. They mentioned that the DSS results showed that they made the right decision in their selection process; additionally, they can consider more potential solutions in their future evaluation.

5.2 Case study 2: innovation-kite customer

Innovation-Kite is a software development company in the Netherlands and Germany with more than 500 employees. They have a "Solution Center" with an international network of experienced ICT-specialists and developers. The experts at Innovation-Kite stated that the agile development methodology requires closer interaction between end-users and developers. The local Agile/Scrum Business Engineering should make general use of specialized developers who can make specific integrations and adjustments (customizations)

Table 4 This table represents the entire list of the feature requirements that were defined by the case study participants

RID	MDSD feature requirements	Nederlandse Spoorwegen (NS)	Innoviation-Kite Customer	Bizzomate	Royal IHC
R1	Identity and permissions management	Must-Have	Must-Have	Must-Have	Must-Have
R2	SQL or NoSQL databases	Must-Have	Must-Have	Must-Have	Must-Have
R3	Tool set	Must-Have	Must-Have	Must-Have	Must-Have
R4	Predefined components	Must-Have	Must-Have	Must-Have	Must-Have
R5	Web portals	Must-Have	Must-Have	Must-Have	Must-Have
R6	Private Cloud platform	Must-Have	Should-Have	Must-Have	Must-Have
R7	Entity-Attribute-Relationship (EAR)	Must-Have	Must-Have	Should-Have	Must-Have
R8	Data rules	Must-Have	Must-Have	Should-Have	Must-Have
R9	Maturity level of the company	Must-Have	Must-Have	Should-Have	Must-Have
R10	Model-centric (Low-code)	Should-Have	Must-Have	Must-Have	Should-Have
R11	Business Analysts	Should-Have	Should-Have	Must-Have	Must-Have
R12	Form & View	Must-Have	Should-Have	Should-Have	Must-Have
R13	Plug and play architecture	Should-Have	Must-Have	Should-Have	Must-Have
R14	Programming optional		Must-Have	Must-Have	Must-Have
R15	Workflow		Must-Have	Must-Have	Must-Have
R16	Process rules	Must-Have	Must-Have	Should-Have	Could-Have
R17	Case flow	Must-Have	Must-Have	Could-Have	Should-Have
R18	Decision table	Should-Have	Must-Have	Could-Have	Must-Have
R19	Process flow		Should-Have	Must-Have	Must-Have
R20	Model only (No-code)	Should-Have	Should-Have	Must-Have	Could-Have
R21	Decision tree	Should-Have	Must-Have	Could-Have	Should-Have
R22	Visual IDE	Should-Have	Should-Have	Could-Have	Must-Have
R23	Professional Developers		Must-Have	Could-Have	Must-Have
R24	Popularity in the market	Should-Have	Should-Have	Should-Have	Should-Have
R25	Expression editor		Should-Have	Should-Have	Must-Have
R26	Web modeling tool		Should-Have	Could-Have	Must-Have
R27	Ontology modelling	Should-Have	Could-Have	Could-Have	Should-Have
R28	Native modeling tool		Could-Have	Could-Have	Must-Have
R29	Object-Role Modeling (ORM)		Should-Have	Could-Have	Should-Have
R30	Natural language rules		Should-Have	Could-Have	Should-Have
R31	Support different stacks		Could-Have	Should-Have	Should-Have
R32	XML/JSON as data storage	Must-Have		Could-Have	
R33	Integrate with an ERP system			Must-Have	Could-Have
R34	Code Generation			Could-Have	Must-Have
R35	Two step generation			Could-Have	Must-Have
R36	Store model locally			Could-Have	Must-Have
R37	General-purpose platform	Must-Have			
R38	REST	Must-Have			
R39	SOAP	Must-Have			
R40	OData services	Must-Have			
R41	Web API	Must-Have			
R42	Service Calls	Must-Have			
R43	Connectors to various back-ends or services	Must-Have			
R44	Company-Branded Templates & Styling	Must-Have			
R45	Domain Experts		Could-Have	Should-Have	Could-Have
R46	Convert model to text		Should-Have	Could-Have	Could-Have
R47	Real-time Analytics	Could-Have	Could-Have	Could-Have	Could-Have
R48	Report and Analytics	Could-Have	Could-Have	Could-Have	Could-Have
R49	Version control	Could-Have	Could-Have	Could-Have	Could-Have
R50	Close to programming language			Could-Have	Should-Have
R51	Model interpretation			Could-Have	Should-Have
R52	Public Cloud platform		Should-Have		
R53	Create extensions and widget libraries	Should-Have			
R54	Multi-channel/Cross-platform Application	Could-Have			
R55	data mapping	Could-Have			
R56	importing data	Could-Have			
R57	Mobile Apps	Could-Have			
R58	Active Community	Could-Have			
R59	Future Roadmap	Could-Have			
R60	Programmed				Could-Have

Note, the Boolean and non-Boolean features that are presented in Tables 1, 2, and 3. The first column signifies the requirement id (RID). We used this column to determine the link between the feature requirements and the actual needs of the case study participants within an existing customer infrastructure and environment, so that test work and specific optimizations can also be performed cost-effectively.

One of their customers was a small startup company with around ten employees. The startup company requested a software application to help them estimate activities, scheduling, cost control, and budget management. The experts at Innovation-Kite wanted to design and implement a customized project management system for this customer by employing one of their strategic technology partners, Betty Blocks, and Mendix. Additionally, the case study participants stated that without the time and budget limitation of their customers, they have enough in-house expertise and knowledge to build an entirely new software product so that high-coding can be considered the third potential solution.

The case study participants joined this research to evaluate the shortlist of potential solutions (Betty Blocks, Mendix, and High-Coding) for this startup company. Moreover, they desired to know about other possible MDD platforms that they have to take into account.

5.2.1 Innovation-kite requirements

The case study participants at Innovation-Kite defined a subset of the customized project management system's key functionality:

- Viewing progress across all ongoing projects, identifying projects at risk, monitoring timelines, and sharing project status in real-time (R15, R47, R48)
- Keeping workflow tools in one place, having a centralized management unit for details and updates, storing projects' files in a secured data storage, and keeping templates always consistent (R2, R5, R6, R52)
- Reporting a clear picture of how the resources are being used (R48)
- Offering account management and provisioning system to define new end-users, roles, and privileges (R1)
- Managing all activities and tasks required to maintain a desired level of excellence (R19)

The Innovation-Kite experts expressed that they want to be independent of particular programming languages and development processes. However, they required a level of flexibility to add new functionality or customize an existing component. Therefore, they prioritized the "Model-centric (Low-code)" feature as Must-Have and the "Model only (No-code)" as a Should-Have feature (R10, R20). Moreover, they wanted to employ their technical knowledge so that a potential MDD platform has to support professional developers (R23). The case study participants mentioned that their customers always want to deploy their software products on the cloud; however, according to their budgets and infrastructures, it can be on private or public clouds; thus, they considered both of these options as Should-Have features (R6, R52).

5.2.2 Results and analysis

The case study participants identified 37 MDD feature requirements, including 43.24% hard-constraint features (Must-Have) and 56.76% soft-constraint features (Should-Have and Could-Have). The DSS suggested five possible solutions, namely Mendix, Salesforce (Lightning), Betty Blocks, OutSystems, and ServiceNow (Now Platform). Table 5 shows that Mendix was the first, and Betty Blocks was the third feasible platform for this case study. They did not consider Salesforce (Lightning) as a solution, as they had no experience with employing this platform.

The DSS scored Betty Blocks as the third solution as it does not support "Decision Table" (R18), "Natural language rules" (R30), and "Object-Role Modeling" (R29). According to our assessment, Mendix does not support "Natural language rules" and "Ontology modeling." Please note that the Should-Have features have higher priorities than Could-Have features, so MDD platforms that support more Should-Have features score higher.

The experts who participated in this case study stated that "OutSystems" could be a potential solution as it supports all the feature requirements that they required. However, they needed to perform a cost-benefit analysis to evaluate its usefulness. The experts mentioned that the DSS could support them for their future evaluation, reducing decision-making time. However, we need to keep the knowledge base of the DSS besides the decision model regularly up to date.

5.3 Case study 3: Bizzomate

Bizzomate is a software consultancy company in the Netherlands to support organizations with technical and technological problems, such as MDD platform selection. Organizations hire external consultancy companies, such as Bizzomate, when their internal resources and expertise are insufficient. External consultants analyze an organization's existing setup and make suggestions. Additionally, the experts at Bizzomate advises its customers on how to configure their software applications, write code, fix bugs, or customize their software systems for specific tasks or businesses.

The experts at Bizzomate stated that they use MDD platforms to increase agility in software development. Such platforms assist them with working closely together in one environment, and various stakeholders can collaborate, create, iterate, and release solutions in a fraction of the time compared to traditional development methods.

Mendix is the leading partner of Bizzomate in the software development process. The experts at Bizzomate typically employ the Mendix platform to implement customized software solutions for their customers. They have recently investigated a little bit regarding other potential platforms and considered Betty Blocks an alternative solution for their customers. The case study participants joined this research to evaluate the selected shortlist of MDD platforms based on requirements that typically take into account to build a customized software solution.

5.3.1 Bizzomate requirements

The case study participants at Bizzomate defined a subset of their essential MDD feature requirements that they generally consider to select an MDD platform for their customers:

- The MDD platform has to enable developers to use models to develop software (R10) and generate code automatically (R35).
- Developers have to use models to build software and solely communicate with each other about the system in terms of models (R20). Note coding terminology is absent.
- A full application could be created without any programming, but developers can use one programming language (R14).
- The platform has to be able to govern users by enforcing both authentication and authorization. Authentication verifies a user's identity. Once authenticated, the verified user may use any of the resources their account is authorized to access (R1).
- Toolset gives modelers access to a what-you-see-is-what-you-get editor, in which access is provided to different user interface components. Within such an editor, the user is free to edit those elements' height and width (R3).
- The platform has to allow users to model user interface predefined components without altering the user interface components' location, width, and height (R4).
- The platform has to support developers with designing and implementing web portals (R5). A web portal is a specially designed website that brings information from diverse sources, like emails, online forums, and search engines, together in a uniform way. Usually, each information source gets its dedicated area on the page for displaying information; often, the user can configure which ones to display.
- The platform should support EAR[5] (R7), which is used to represent attributes as well as entities and relationships.

[5] Entity-Attribute-Relationship.

The participants expressed that non-Boolean features such as popularity in the market (R24) and the company's maturity level (R9) play an essential role in the MDD platform selection process.

5.3.2 Results and analysis

The case study participants identified 42 MDD feature requirements, including 30.95% hard-constraint features (Must-Have) and 69,05% soft-constraint features (Should-Have and Could-Have). They defined a limited number of Must-Have features as they did not focus on a particular IT project. The case study participants put their emphasis on the platform-specific features that they were already familiar with. In other words, their feature requirements were biased to the features that their shortlist of MDD platforms was supported them. Thus, the DSS results did not surprise them. Table 5 shows that Mendix was the first, and Betty Blocks was the third feasible platform for this case study.

The DSS scored Betty Blocks as the third solution as it does not support a set of the soft constraint requirements, such as "Code Generation" (R34), "Decision Table" (R18), "Natural language rules" (R30), and "Object-Role Modeling" (R29).

The case study participants stated that they have never considered Appian as an alternative solution for their project because they do not have any experience with this platform; however, they will investigate its functionality and possibilities for future projects.

5.4 Case study 4: royal IHC

Royal IHC is an international supplier of innovative equipment, ships, and services for offshore, dredging, and wet mining. Royal IHC enables customers to execute complex projects from the water surface to the ocean floor in the most challenging maritime environments. The head office is located in the Netherlands, but more than 3000 employees work from offices worldwide. Thus, customer support is provided on every continent. The company faces ever-changing customer needs and healthy global competition.

Digitization is an inevitable factor for every organization, including Royal IHC, to provide customers with innovative solutions. The case participants asserted that Royal IHC would gain increased business agility, transparency, and uniformity of information by consolidating the IT environment, a vital part of its national and international value chain. Recently, the ICT department at Royal IHC introduced an ERP application called "One IHC," based on the low-code platform from Mendix, to support their key strategic goals of collaboration, globalization, and growth.

Royal IHC participated in this research to assess their current platform (Mendix) and an alternative to it (Betty Blocks).

Table 5 DSS results

	Nederlandse Spoorwegen (NS)		Innovation-Kite customer		Bizzomate		Royal IHC	
Context	Workflow management system		Project management system		Software consulting company		ERP system	
#Employees	5–10		5–10		20–50		50–100	
Requirements								
Must-have		52.50%		43.24%		30.95%		50.00%
Should-Have		25.00%		37.84%		23.81%		28.57%
Could-Have		22.50%		18.92%		45.24%		21.43%
Won't-Have		0.00%		0.00%		0.00%		0.00%
#Feature Req.	40		37		42		42	
CP Shortlist								
1	Mendix		Betty Blocks		Mendix		Mendix	
2	Microsoft PowerApps		Mendix		Betty Blocks		Betty Blocks	
3	High-Code		High-Code					
DSS Solutions								
1	Mendix	91%	Mendix	93%	Mendix	99%	Mendix	77%
2	Oracle APEX	86%	Salesforce (Lightning)	92%	Appian	90%	OutSystems	67%
3	Microsoft PowerApps	84%	Betty Blocks	72%	Betty Blocks	87%	Betty Blocks	48%
4	OutSystems	77%	OutSystems	71%				
5			ServiceNow (Now Platform)	70%				

Moreover, they wanted to know about any potential MDD platforms that could be considered in the near future.

5.4.1 Royal IHC requirements

The case study participants defined the following subset of requirements of One IHC to select the best fitting MDD platform.

- The platform has to support Royal IHC's demand for applications with a native mobile experience on multiple devices. The platform needs to be available both online and offline (R22, R26, R29).
- The platform has to enhance the collaboration between business stakeholders and IT (R11). It leads to an increase in the organization's development capacity to meet the growing demand for applications, dashboards, and portals (R23, R47, R48).
- The platform has to support component-based architectures (R13).
- The platform has to enable users to save models as byte code on their local machine (R36).
- The platform could be integrated with an ERP system (R33).
- A set of data rules must be defined to ensure that only values compliant with the data rules are allowed within a data object. Data rules will form the foundation for correcting or removing data (R8).

5.4.2 Results and analysis

The case study participants identified 42 MDD feature requirements that half of them were Must-Have features. Their hard-constrained feature requirements were mainly biased toward the features the shortlist of MDD platforms already supported, and they were completely aware of them. They wanted to know about alternative solutions, so the other half of their feature requirements were mainly about nice-to-have MDD features (soft-constrains). Table 5 shows that Mendix was the first, and Betty Blocks was the third feasible platform for this case study.

The DSS scored Betty Blocks as the third solution as it does not support a set of the soft constraint requirements, such as "Code Generation" (R34), "Decision Table" (R18), "Ontology modeling (R27)" (R30), and "Object-Role Modeling" (R29).

The case study participants asserted the OutSystems platform could be an interesting alternative to them, and they have not considered it up to now because of a lack of knowledge and expertise regarding this platform; however, they will consider it an option in their future evaluations. Note Royal IHC hired some experts from Bizzomate to support them with their decision-making process, so it was not surprising that their shortlists were the same.

6 Discussion

6.1 Case study participants

Software products may be more successful in some regions. Not every MDD platform is equally represented in different regions of the world. We observed specific MDD platforms that are primarily active in the Netherlands and focus their support efforts on the Netherlands because there is the most business for them. As aforementioned, the case study companies were located in the Netherlands. Almost all of them preferred to select one of the locally produced MDD platforms, specifically Mendix and Betty Blocks, because of their concerns regarding legal issues and safety. While we did not collect each platform's sales data, we have to remain cognizant that some MDD platforms may be over- or under-represented in particular geographic regions.

The total cost of ownership of MDD platforms plays an inevitable factor in the decision-making process. However, none of the case study participants considered it a "Must-Have" feature, as they believed that functional suitability, maturity, and popularity of potential solutions should be prioritized higher. One of the case study participants expressed that "we have to examine total costs besides total benefits to make a rational decision about the potential solution that will provide the highest positive impact on the future of our business." The DSS assigns higher scores to the general-purpose platforms, such as Mendix and Appian, as they offer a vast set of services and functions.

Biases, such as motivational and cognitive [40], arise because of shortcuts or heuristics that decision-makers use to solve problems and perform tasks. The Hawthorne effect [41], which is the tendency for decision-makers to change their behavior when they are being observed, is a form of cognitive bias. The case study participants might have been more careful in the observational setting than in the real setting because they are being observed by scientists judging their selected MDD feature requirements and priorities. Moreover, the Bandwagon effect [42], which is the tendency to do or believe things because many other decision-makers do or believe the same, is another form of cognitive bias. The Bandwagon effect typically shows up in group decisions. To mitigate the Hawthorne and Bandwagon effects, individual *and* group interviews have been conducted.

Please note that sometimes the case study participants are biased toward a specific alternative solution. For instance, in this study, all case study participants asserted that Mendix was one of their alternative solutions. Regional limitation, popularity in the market, financial plus political issues, and tacit knowledge of the case study participants can be considered potential factors which were limiting alternative solutions. Accordingly, conducting case studies in different regions could lead to different MDD feature requirements; consequently, the DSS can suggest different rankings or even entirely different alternative solutions.

6.2 Domain experts

The experts expressed that the decision-making process is a lot different for small organizations than large ones. Looking at the IT landscape, we notice a difference in the selection process because the requirements of small and large organizations are remarkably different. Small enterprises typically start purchasing a unique MDD platform to solve multiple problems; they cannot invest in multiple platforms to perform different tasks because of financial constraints. Larger enterprises can invest more money to employ multiple MDD platforms for different tasks. However, using multiple platforms requires more training costs and knowledge sharing possibilities. The best-fitting MDD platform for a company should add values instead of just solving quick issues.

The experts asserted that MDD platforms should not be employed in three use cases: (1) Complex applications with rich functionality, such as software products, require continuous development and maintenance to integrate a significant number of services and components from third parties. Thus, an MDD platform is not the best way to develop complex applications, and it is better to hire a development team that can address the functional requirements and quality concerns. (2) The MDD platforms should not be used to build applications for enterprises that employ the generated applications to perform their core businesses. The key rationale to avoid using MDD platforms for such scenarios is that MDD platforms are mainly designed for simple reoccurring problems, and they limit creativity by simplifying the complexity of the real world. Thus, such enterprises will be limited to a set of predefined functions and cannot make new wild decisions. (3) The MDD platforms would not be a cost-effective solution for businesses that rely principally on freemium end-users, as MDD platforms may charge their customers based on the number of their end-users.

Software development is an iterative and incremental process, based on a collection of invaluable concepts and principles [43,44]. As extensively discussed in [45,46], it should be kept in mind that MDD is not a concrete methodology, but a generic approach that can be applied to software development processes to take advantage of its promises. The main issue with using models to drive software engineering directly is that they are far from flexible. First, end-users are limited by the type of the MDD platforms they use. Second, they are only flexible in the parts of the solution covered by the used domain-specific languages. The higher the level of abstraction, the more the commonalities will be "hard-coded" in the MDD platforms. Third, sometimes models are made flexible at only a limited set of predefined components by lower-level languages.

6.3 The decision model

The case study participants confirm that the updated and validated version of the DSS is useful in finding the shortlist of feasible solutions. Finally, it reduces the time and cost of the decision-making process. Our website[6] is up and running to keep the knowledge base of the decision support system up to date and valid. We aim to create a community around the platform that regularly updates the curated knowledge base with new MDD platform features. We consider it as future work to enable third parties to add new features and products to the database in a wiki-style manner. These additions need to be approved by us, as it may be tempting for product marketers to overstate the features present in the platform.

The study of heuristics-and-biases has investigated various decision-making shortcuts and has documented their inferior performance [47,48]. However, these uncomplicated heuristics can be viewed as smart approaches to save time so that a decision-maker can respond immediately [49]. Applying simple rules is sometimes an answer to complexity [50]. When faced with a problem that is highly complex to solve optimally, the decision-maker falls back on a simple rule that makes sense based on what is understood. Fast-and-frugal heuristics can perform well in certain domains [51], such as MDD selection, to find the best fitting alternatives based on a limited set of criteria, for instance, background knowledge and experience of the decision-maker. Thus, the decision model can be considered a method to evaluate the shortlist of decision-makers' alternative solutions and assist them with decision-making under uncertainty.

We believe that the theoretical contribution and the answer of the main research question (see Sect. 3.2) of this study is a decision model that can be used to make informed decisions in software production, and models from software engineering, such as the ISO standard quality model and the MoSCoW prioritization technique, are fundamental building blocks in such decisions. Researchers can replace the ISO standard quality model with more specific quality attributes to customize the decision model. Although we employ the MoSCoW prioritization technique to simplify the understanding and manage priorities, other researchers can employ other types of prioritization techniques to define the feature requirements.

With the knowledge available through the decision model, researchers can more rapidly evaluate MDD platforms in the market, add more platforms or features to the decision

[6] https://dss-mcdm.com.

model systematically according to the presented guideline, and employ the reusable knowledge (presented in Tables 1, 2, 3, and 4) to develop new concepts and solutions for future challenges.

6.4 Limitations and threats to validity

The validity assessment is an essential part of any empirical study. Validity discussions typically involve construct validity, internal validity, external validity, and conclusion validity.
Construct validity refers to whether an accurate operational measure or test has been used for the concepts being studied. In the literature, decision-making is typically defined as a process or a set of ordered activities concerning stages of problem identifying, data collection, defining alternatives, selecting a shortlist of alternatives as feasible solutions with the ranked preferences [52,53]. To mitigate the threats to the construct validity, we followed the MCDM theory and the six-step of a decision-making process [20] to build the decision model for the MDD platform selection problem. Moreover, we employed document analysis and expert interviews as two different knowledge acquisition techniques to capture knowledge regarding MDD platforms. Additionally, the DSS and the decision model have been evaluated through four real-world case studies at four different real-world enterprises in the Netherlands.
Internal validity attempts to verify claims about the cause-effect relationships within the context of a study. In other words, it determines whether the study is sound or not. To mitigate the threats to the internal validity of the decision model, we define DSS success when it, in part, aligns with the case-study participants shortlist and when it provides new suggestions that are identified as being of interest to the case study participants. Emphasis on the case study participants' opinion as a measurement instrument is risky, as the case study participants may not have sufficient knowledge to make a valid judgment. We counter this risk by conducting more than one case study, assuming that the case study participants are handling their interest and applying the DSS to other problem domains, where we find similar results [11,54–56].
External validity concerns the domain to which the research findings can be generalized. External validity is sometimes used interchangeably with generalizability (feasibility of applying the results to other research settings). We evaluated the decision model in the context of Dutch enterprises. To mitigate threats to the research's external validity, we captured knowledge from different sources of knowledge without any regional limitations to define the constructs and build the decision model. Accordingly, we hypothesize that the decision model can be generalized to all enterprises worldwide who face uncertainty in the MDD platform selection problem. Another question is whether the framework and the DSS can be applied to other problem domains as well. The problem domains [11,54–56] were selected opportunistically and pragmatically, but we are convinced that there are still many decision problems to which the framework and the DSS can be applied. The categories of problems to which the framework and the DSS can be applied successfully can be summed up as follows: (1) The problem regards a technology decision in system design with long-lasting consequences, (2) there is copious scientific, industry, and informal knowledge publicly available to software engineers, and (3) the (team of) software engineer(s) is not knowledgeable in the field but very knowledgeable about the system requirements.
Conclusion validity verifies whether the methods of a study such as the data collection method can be reproduced, with similar results. We captured knowledge systematically from the sources of knowledge following the MCDM framework [11]. The accuracy of the extracted knowledge was guaranteed through the protocols that were developed to define the knowledge extraction strategy and format (see Appendix B). A review protocol was proposed and applied by multiple research assistants, including bachelor and master students, to mitigate the threats to the research's conclusion validity. By following the framework and the protocols, we keep consistency in the knowledge extraction process and check whether the acquired knowledge addresses the research questions. Moreover, we crosschecked the captured knowledge to assess the quality of the results, and we had at least two assistants extracting data independently.

7 Related work

In this study, snowballing was applied as the primary method to investigate the existing literature regarding techniques that address the MDD platform selection problem. Table 6 summarizes a subset of selected studies that discuss the problem. As aforementioned, the last column (*Cov.*) of Table 6 indicates the percentage of the coverage of the considered criteria within the selected studies. On average, 75% of those criteria are already considered in this study. In other words, the decision model contains a significant number of criteria, including features and quality attributes, that have been mentioned in the literature.

7.1 Intelligent business process management suite selection

In the literature, a wide range of publications has assessed different iBPMS platforms and compared them against a set of criteria. Gartner [17] reported considered 26 criteria, such as sales execution/pricing and marketing strategy, to evaluate nineteen iBPMS platforms as leaders, challengers, niche players, and visionaries.

Sanchis et al. [57] introduced a framework to manage the overall network of a collaborative manufacturing and logistics environment that enables humans, applications, and Internet of things devices to seamlessly communicate and interoperate in the interconnected environment, promoting resilient digital transformation. Then, the authors conducted a literature study regarding MDD platforms to identify sixteen features that they support. Finally, the authors compared their framework with six low-code platforms against the features.

Forrester [18] researched a list of low-code platforms, including 13 vendors, to consider for the evaluation. From that initial pool of vendors, they narrowed the final list based on several inclusion criteria, such as low-cost-of-entry commercial models, building many business use cases, and primarily targeting large enterprises. Then, they collected data from products and strategies through a detailed questionnaire, demos and briefings, and a reference-customer survey. They used those inputs, along with the analyst's experience and expertise in the marketplace, to score the platforms, applying a relative rating system that compares each platform against the others in the evaluation.

Vugec et al. [58] identified the following four social business process management dimensions based on literature study: egalitarianism, collective intelligence, self-organization, and social production. Next, the authors selected nine iBPMSs reported by Gartner [17] to compare their functionality against the social dimensions.

Sattar [59] assessed 13 low-code platforms based on supporting ten criteria, such as cloud platform attributes, and then introduced a decision tree for the low-code platform selection problem.

Zolotas et al. [60] presented a low-code platform based on the REST architecture that enables developers to model attribute-based access control policies without requiring any code writing. Then, the authors compared their approach with eleven low-code platforms against four security features. Melo et al. [61] considered the ISO/IEC 25010 standard besides a set of quality aspects, such as vendor and cost, to evaluate Oracle Apex and OutSystems.

Hendriks et al. [62] researched for essential characteristics of low-code platforms and how they should be matched to each other. They performed interviews with industry experts to determine how the industry looks upon matching situations and platforms. Finally, they introduced a framework based on the characteristics to support organizations with the iBPMS platform selection.

Forrester [5] carried out an online vendor survey to assess 42 low-code platforms. The authors of the report then divided low-code platforms into the following five categories using their background and functionality: general purpose platforms, process application platforms, database application platforms, request handling platforms, and mobile application platforms.

Wasilewski [63] compared the Gartner's reports about BPMS and iBPMS markets from 2009 to 2015 to analyze the behavior of the leaders in the Magic Quadrant.

7.2 Business process management suite selection

A vast range of BPMS platforms is currently available on the market to cater to a wide variety of modeling objectives. A subset of publications that reported on the evaluation of BPMS platforms is presented as follows.

Şen et al. [64] applied the analytic hierarchy process (AHP)[7] and the Fuzzy Technique for Order Preference by Similarity to Ideal Solution (FTOPSIS)[8] to address the problem of choosing BPMS for a retailer operating in the textile sector. In the first step, the AHP implementation and the pairwise comparisons were taken from the seven decision-makers determined the decision criteria' weights. In the second step, the FTOPSIS method was performed to select the best fitting BPMS with the decision-makers' quantitative and qualitative evaluations.

Meidan et al. [65] performed a formal survey based on a systematic literature review method and quality models to classify and compare BPMSs according to a set of characteristics of open source BPMS. Additionally, they observed that every BPMS provider used its terminology to explain business process concepts (e.g., join and fork elements, exception handling, events). Unifying the terminology could help to improve interoperability and portability among BPMSs. Furthermore, the evaluation showed that BPMSs could be classified into two families: The first one is oriented to normal users (e.g., Bonita and ProcessMaker), and the second one is platform-oriented to developers and expert users (e.g., Activiti, Camunda, and jBPM).

Vukšić et al. [66] presented a guideline and a set of selection criteria such as maturity, reporting and analytics, business rules, user interface and user experience, and modeling notation to evaluate three BPMS platforms (IBM, K2, and Software AG).

Kapteijns et al. [72] performed research regarding BMPS platforms in small-scale development projects to investigate the case study participants' level of satisfaction. Moreover, the authors collected a set of BMPS features from the literature to compare four BMPS platforms against each other.

[7] AHP is an MCDM technique for making decisions between alternatives. AHP allows decision-makers to capture their strategic goals as a set of weighted criteria that they then use to rank alternatives.

[8] The *TOPSIS* is an MCDM approach that employs information entropy to assess alternatives. Fuzzy logic is an approach to computing based on "degrees of truth" rather than the usual Boolean logic. Sometimes combinations of fuzzy logic with other MCDM approaches, such as FTOPSIS, are employed to solve MCDM problems.

Table 6 Comparison of a subset of selected studies from the literature that addresses the MDD platform selection problem

Study	Type	R. method	Data Col.	Platforms	Year	Approach	MCDM	PC	QA	#C	#A	#CQ	#CF	#CA	Cov. (%)
This study	RP	Design Sc. Experts Doc Analysis Case Study	Mixed	iBPMS BPMS	2020	DSS	Yes	No	ISO/IEC 25010 EX. ISO/IEC 9216	151	30	57	94	30	100
[57]	RP	SLR	Quantitative	iBPMS	2020	Benchmarking	No	N/A	Domain Specific	16	7	0	11	2	68
[7]	Report	Survey Doc Analysis	Quantitative	iBPMS	2019	SA	No	N/A	Domain Specific	15	18	7	5	12	80
[18]	Report	Survey Doc Analysis	Quantitative	iBPMS	2019	SA	No	N/A	Domain Specific	20	13	2	14	9	80
[17]	Report	Survey Doc Analysis	Quantitative	iBPMS	2019	SA	No	N/A	Domain Specific	26	19	9	4	6	50
[58]	RP	Doc Analysis	Quantitative	iBPMS	2019	Benchmarking	No	N/A	Domain Specific	4	9	1	3	5	100
[59]	MT	Case Study Experts Doc Analysis	Mixed	iBPMS	2018	Benchmarking	No	N/A	Domain Specific	10	13	2	8	3	100
[60]	RP	Doc Analysis	Quantitative	iBPMS	2018	Benchmarking	No	N/A	N/A	4	11	0	2	5	50
[64]	RP	Experts	Qualitative	BPMS	2018	AHP FTOPSIS	Yes	Yes	Domain Specific	4	5	2	2	0	100
[62]	MT	Experts	Qualitative	iBPMS	2017	WSM	Yes	No	Domain Specific	16	5	5	7	3	75
[61]	RP	Doc Analysis	Quantitative	iBPMS	2017	Benchmarking	No	N/A	ISO/IEC 25010 Domain Spe-cific	38	2	20	12	2	84

Table 6 continued

Study	Type	R. method	Data Col.	Platforms	Year	Approach	MCDM	PC	QA	#C	#A	#CQ	#CF	#CA	Cov. (%)
[65]	RP	SLR	Quantitative	BPMS	2017	SA	No	N/A	Domain Specific	41	7	10	21	0	76
[5]	Report	Survey Doc Analysis	Quantitative	iBPMS	2016	SA	No	N/A	Domain Specific	5	42	0	4	12	80
[63]	RP	Doc Analysis	Quantitative	BPMS	2016	SA	No	N/A	N/A	35	27	0	23	6	66
[66]	RP	SLR	Quantitative	BPMS	2016	Benchmarking	No	N/A	Domain Specific	11	3	1	8	2	82
[67]	RP	Case Study Doc Analysis	Qualitative	BPMS	2015	WSM	Yes	No	Domain Specific	11	13	3	8	1	100
[68]	RP	Survey	Quantitative	BPMS	2015	WSM	Yes	No	Domain Specific	4	8	0	3	0	75
[69]	RP	Survey	Qualitative	MDA	2014	Benchmarking	No	N/A	Domain Specific	9	8	4	3	0	78
[70]	RP	Experts Doc Anal-ysis	Qualitative	BPMS	2014	FTOPSIS	Yes	Yes	Domain Specific	48	5	19	16	0	73
[71]	RP	Experts Doc Anal-ysis	Qualitative	BPMS	2010	WSM	Yes	No	Domain Specific	53	10	10	40	0	94
[72]	RP	Case Study Doc Analysis	Qualitative	BPMS	2009	Benchmarking	No	N/A	Domain Specific	12	4	1	6	1	58
[73]	RP	Case Study Doc Analysis	Qualitative	BPMS	2009	AHP	Yes	Yes	Domain Specific	10	5	5	3	0	80

The first six columns indicate the selected study (Study), the publication type (Type) (including research paper (RP), master thesis (MT), and report (R)), the research methods (R. Method) (including expert interview (Experts), document analysis (Doc Analysis), design science (Design Sc.), systematic literature review (SLR), survey, and case study), the data collection type (Data Col.), and MDD platforms (Platforms), and the publication year (Year) of the corresponding selected studies, respectively. The seventh column (Approach) indicates the decision-making approach that the studies have employed to address the MDD platform selection problem. The eighth column (MCDM) denotes whether the corresponding decision-making technique is an MCDM approach. The ninth column indicates whether the MCDM approach applied pairwise comparison (PC) as a weight calculation method or not. The tenth column (QA) determines the type of quality attributes. The eleventh and twelfth columns (#C and #A) signify the number of criteria and alternatives considered in the selected studies. The next three columns indicate the numbers of common quality attributes (#CQ), features (#CF), and alternatives (#CA) of this study (the first row) with the selected studies. The last column (Cov.) shows the percentage of the coverage of the considered criteria (quality attributes and features)

Delgado et al. [67] presented a systematic approach based on the weighted sum model (WSM)[9] for assessing BPMS tools, both open-source and proprietary. The authors suggested a list of relevant vital characteristics for BPMS tools and a way of evaluating the provided support using test cases and a case study to provide an overall view of the tool support.

Mejri et al. [68] used a questionnaire to capture a set of BPMSs' strengths and weaknesses in terms of flexibility from their researchers and developers. Then, they used the weighted sum method to calculate the scores of the BPMSs and rank them accordingly. Ravasan et al. [70] introduced a set of functional and non-functional criteria for selecting the right BPMS for an organization. The authors then applied the FTOPSIS approach to calculate the weight of the criteria based on decision-makers' requirements and priorities.

Davies and Reeves [71] reported on the Australian government department's experiences in selecting a BPM tool to support its process modeling, analysis, and design activities. Candidate solutions were identified for evaluation by researching case studies and market overviews.

Štemberger et al. [73] presented a method for BPMS selection to support decision-makers (managers and IT experts) with the BPMS selection process. Their approach was based on the AHP method and developed BPM tools features from project goals and critical success factors. Their research reported two points: (1) to choose the best fitting software tool for a particular application and business, an organization requires a method for the evaluation of a BPMS, and (2) the selection of a BPMS is a multi-criteria decision process. Accordingly, a suitable method for making multi-objective decisions should be employed.

7.3 Strengths and liabilities

Studies based on benchmarking [57–61,66,69,72] and statistical analysis (SA) [5,7,17,18,63,65] are typically time-consuming approaches and mainly applicable to a limited set of alternatives and criteria, as they require in-depth knowledge of programming languages and concepts. Such analysis is subject to increased error, particularly when a relational analysis is used to attain a higher interpretation level. One of the critical issues regarding statistical analysis is the tendency to skip unjustified conclusions concerning causal relationships. Researchers usually obtain evidence that two variables are highly correlated; however, that does not prove that one variable causes another. Finding relationships among correlations and causation needs in-depth expertise and experience regarding MDD platforms and their concepts, as such links are mainly qualitative. Additionally, benchmarking and statistical analysis are likely to become outdated soon and continuously kept up to date, which is a high-cost process.

As aforementioned, finding the best MDD platform for an organization is a decision-making process that deals with evaluating several alternatives and criteria. Accordingly, the selected platform should address the concerns and priorities of the decision-makers. Conversely to MCDM approaches, studies based on "Benchmarking" and "Statistical Analysis" principally offer generic results and comparisons and do not consider individual decision-maker needs and preferences. A variety of MCDM approaches have been introduced by researchers recently.

The majority of the MCDM techniques [62,64,64,67,68, 71,73] define domain-specific quality attributes to evaluate the alternatives. Such studies are mainly appropriate for specific case studies. Furthermore, MCDM approaches are valid for a specified period; therefore, the results of such studies will be outdated by MDD platforms' advances. Note that, in our proposal, this is also a challenge, and we propose a solution for keeping the knowledge base up to date in Sect. 6. Some of the methods, such as FTOPSIS and AHP, are not scalable, so in modifying the list of alternatives or criteria, the evaluation process should be redone. Therefore, these methods are costly and applicable to only a small number of criteria and alternatives. This study has considered 151 criteria and 30 alternatives to building a decision model for the MDD platform selection problem.

In contrast to the named approaches, the cost of creating, evaluating, and applying the proposed decision model in this study is not penalized exponentially by the number of criteria and alternatives. It is an evolvable and expandable approach that splits down the decision-making process into four maintainable phases [54]. Moreover, we introduce several parameters to measure the values of non-Boolean criteria, such as the maturity level and market popularity of the MDD platforms. The proposed decision model addresses main knowledge management issues, including capturing, sharing, and maintaining knowledge. Furthermore, it uses the ISO/IEC 25010 [22] as a standard set of quality attributes. This quality standard is a domain-independent software quality model and provides reference points by defining a top-down standard quality model for software systems.

Recently, we built five decision models based on the framework to model the selection of database management systems [54], cloud service providers [11], blockchain platforms [55], software architecture patterns [56], and programming languages. In all five studies, case studies were conducted to evaluate the DSS's effectiveness and usefulness in addressing MCDM problems. The results confirmed that the DSS performed well to solve the mentioned problems in software production. We believe that the framework can be employed as a guideline to build decision models for MCDM problems in software production.

[9] *weighted sum model* is an aggregation function that transforms multiple criteria into a single value by multiplying each criterion by a weighting factor and summing up all weighted criteria.

8 Conclusion and future work

In this study, the selection process of the model-driven development platforms is modeled as a multi-criteria decision-making problem that deals with evaluating a set of alternatives and taking into account a set of decision criteria [9]. Moreover, we presented a decision model for the selection problem based on the technology selection framework [11]. The novelty of the approach provides knowledge about model-driven development platforms to support uninformed decision-makers while contributing a sound decision model to knowledgeable decision-makers. Furthermore, it incorporates foundational software engineering concepts, such as the ISO software quality standards and the MoSCoW prioritization technique, besides knowledge engineering theories. We conducted four industry case studies to evaluate the decision model's usefulness and effectiveness to address the decision problem. We find that while organizations are typically tied to particular ecosystems by extraneous factors, they can benefit significantly from our DSS by evaluating their decisions, exploring more potential alternative solutions, and analyzing an extensive list of features.

The case studies show that this article's decision model also provides a foundation for future work on MCDM problems. We intend to build trustworthy decision models to address the *programming language* selection problem as our (near) future work.

A Expert profiles

See Table 7.

Table 7 The company, position, and number of years of experience (Exp.) of the domain experts who participated in this research

#	Company	Position	Exp.
1	42 Windmills	Lead Architect	8
2	42 Windmills	CTO	11
3	AFAS	CIO	24
4	AFAS	Architecture Team Lead	7
5	Betty Blocks	CEO	12
6	Blueriq	Product Owner	8
7	OpenText Cordys	Product manager	11
8	OpenText Cordys	Founder	19
9	GeneXus	CEO	20
11	GeneXus	Business developer	12
12	IBM	Software Architect	13
13	Mendix	CTO	15
14	Mendix	Team Lead Mobile team	6
15	Outsystems	Pre-sales architect	15
16	Outsystems	Regional sales executive	15
17	Servoy	CEO	19
18	Thinkwise	CTO	16
19	Triggre	CTO	7
20	Usoft	CTO	19
21	WEM.io	Lead developer	11
22	GorillaIT	CEO	9
23	ForMetis	CEO	23
24	ForMetis	Business developer	9
25	Freelancer	Software architect	7
26	Freelancer	Software architect	36

B Expert interview protocols

[Evaluation of the MDD features and platforms]

Step 1 A brief description of the project, the decision model, the DSS, and the main goal of the interview.
Step 2 Estimation of the expert background knowledge regarding MDD platforms and concepts:
What do you understand under MDD?
What do you understand under MDD Platforms?
What do you understand under code generator and generation?
What do you understand under the model interpreter and interpretation?
Step 3 Introductory questions:
How long have you worked on MDD?
Could you describe what your current position encompasses?
How is your position related to MDD?
Step 4 Decision-making questions:
When do enterprises need MDD platforms?
How do enterprises typically select MDD platforms?
What are the essential features from your perspective for selecting the best fitting MDD platforms?
Which MDD platforms are typically considered as alternative solutions by enterprises?
Step 5 Evaluation of the sets of MDD features/platforms:
What do you think about these MDD features/ platforms?
Which MDD features/platforms should be excluded from the list?
Which MDD features/platforms should be added to the list?
Step 6 Closing
What do you think about our work?
May we contact you if we have any further questions?
Can we use the name of your company in the scientific paper, or do you prefer an anonymous name?
Can we use your name in the scientific paper, or do you prefer an anonymous name?
Do you have any questions?

[Mapping between the MDD features and the quality attributes]

Step 1 A brief description of the project, the decision model, the DSS, and the main goal of the interview.
Step 2 Estimation of the expert background knowledge regarding MDD platforms and concepts:
What do you understand under MDD?
What do you understand under MDD Platforms?
Are you familiar with the ISO/IEC quality models?
Step 3 Mapping between the MDD features and the quality attributes: (Note: this step will be repeated for all of the features and quality attributes).
Does the MDD feature [X] have a positive impact on the quality attribute [Y]? For instance, if an MDD platform supports "SOAP protocol" means that it has positive impacts on "Availability" and "Interoperability."
Step 4 Closing
What do you think about our work?
May we contact you if we have any further questions?
Can we use the name of your company in the scientific paper, or do you prefer an anonymous name?
Can we use your name in the scientific paper, or do you prefer an anonymous name?
Do you have any questions?

References

1. Olariu, C., Gogan, M., Rennung, F.: Switching the center of software development from it to business experts using intelligent business process management suites. In: Soft Computing Applications, pp. 993–1001. Springer (2016)
2. Brambilla, M., Cabot, J., Wimmer, M.: Model-driven software engineering in practice. Synth. Lect. Softw. Eng. **3**(1), 1–207 (2017)
3. Staron, M.: Adopting model driven software development in industry: a case study at two companies. In: International Conference on Model Driven Engineering Languages and Systems, pp. 57–72. Springer (2006)
4. García-Borgoñon, L., Barcelona, M.A., García-García, J.A., Alba, M., Escalona, M.J.: Software process modeling languages: a systematic literature review. Inf. Softw. Technol. **56**(2), 103–116 (2014)
5. Richardson, C., Rymer, J.R.: Vendor Landscape: The Fractured, Fertile Terrain of Low-Code Application Platforms. FORRESTER, Janeiro (2016)
6. Sendall, S., Kozaczynski, W.: Model transformation: the heart and soul of model-driven software development. IEEE Softw. **20**(5), 42–45 (2003)
7. Vincent, P., Iijima, K., Driver, M., Wong, J., Natis, Y.: Magic Quadrant for Enterprise Low-Code Application Platforms. Gartner Inc, Stamford (2019)
8. Hutchinson, J., Whittle, J., Rouncefield, M.: Model-driven engineering practices in industry: social, organizational and managerial factors that lead to success or failure. Sci. Comput. Program. **89**, 144–161 (2014)
9. Triantaphyllou, E., Shu, B., Sanchez, S.N., Ray, T.: Multi-criteria decision making: an operations research approach. Encycl. Electr. Electron. Eng. **15**(1998), 175–186 (1998)
10. Dhiman, H.S., Deb, D.: Decision and Control in Hybrid Wind Farms. Springer, Berlin (2020)
11. Farshidi, S., Jansen, S., De Jong, R., Brinkkemper, S.: A decision support system for cloud service provider selection problems in software producing organizations. In: 2018 IEEE 20th Conference on Business Informatics (CBI), vol. 1, pp. 139–148. IEEE (2018)
12. Farshidi, S., Jansen, S., De Jong, R., Brinkkemper, S.: Multiple criteria decision support in requirements negotiation. In: the 23rd International Conference on Requirements Engineering: Foundation for Software Quality (REFSQ 2018), vol. 2075, pp. 100–107 (2018)
13. Hailpern, B., Tarr, P.: Model-driven development: the good, the bad, and the ugly. IBM Syst. J. **45**(3), 451–461 (2006)
14. Ceri, S., Brambilla, M., Fraternali, P.: The history of webml lessons learned from 10 years of model-driven development of web applications. In: Conceptual modeling: Foundations and applications, pp. 273–292. Springer (2009)
15. Pastor, O., Molina, J.C.: Model-Driven Architecture in Practice: A Software Production environment based on conceptual modeling. Springer, New York (2007)
16. Brown, A.W.: Model driven architecture: principles and practice. Softw. Syst. Model. **3**(4), 314–327 (2004)
17. Dunie, R., Schulte, W.R., Cantara, M., Kerremans, M.: Magic Quadrant for Intelligent Business Process Management Suites. Gartner Inc, Stamford (2019)
18. Rymer, J.R., Koplowitz, R., Leaders, S.A., Mendix, K., are Leaders, S., ServiceNow, G., Performers, S., MatsSoft, W., are Contenders, T.: The forrester wave^{TM}: Low-code development platforms for ad&d professionals
19. Van Der Aalst, W.M.: Business process management demystified: a tutorial on models, systems and standards for workflow man-

agement. In: Advanced Course on Petri Nets, pp. 1–65. Springer (2003)
20. Majumder, M.: Multi criteria decision making. In: Impact of Urbanization on Water Shortage in Face of Climatic Aberrations, pp. 35–47. Springer (2015)
21. Dvořák, O., Pergl, R., Kroha, P.: Affordance-driven software assembling. In: Enterprise Engineering Working Conference, pp. 39–54. Springer (2018)
22. ISO: Iec 25010: 2011 systems and software engineering: systems and software quality requirements and evaluation (square): system and software quality models. Int. Organ. Stand. **34**, 2910 (2011)
23. Carvallo, J.P., Franch, X.: Extending the ISO/IEC 9126-1 quality model with non-technical factors for cots components selection. In: Proceedings of the 2006 International Workshop on Software Quality, pp. 9–14. ACM (2006)
24. Simon, H.A.: The Sciences of the Artificial, 3rd edn. MIT Press, Cambridge (1996)
25. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design science in information systems research. MIS Q. **28**, 75–105 (2004)
26. Fortus, D., Krajcik, J., Dershimer, R.C., Marx, R.W., Mamlok-Naaman, R.: Design-based science and real-world problem-solving. Int. J. Sci. Educ. **27**(7), 855–879 (2005)
27. Walls, J.G., Widmeyer, G.R., El Sawy, O.A.: Building an information system design theory for vigilant EIS. Inf. Syst. Res. **3**(1), 36–59 (1992)
28. Meredith, J.R., Raturi, A., Amoako-Gyampah, K., Kaplan, B.: Alternative research paradigms in operations. J. Oper. Manag. **8**(4), 297–326 (1989)
29. Baxter, L.A.: A tale of two voices: relational dialectics theory. J. Fam. Commun. **4**(3–4), 181–192 (2004)
30. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design science in information systems research. Manag. Inf. Syst. Q. **28**(1), 6 (2008)
31. Chen, W.K.: The Electrical Engineering Handbook. Elsevier, New York (2004)
32. Myers, M.D., Newman, M.: The qualitative interview in is research: examining the craft. Inf. Organ. **17**(1), 2–26 (2007)
33. Bowen, G.A., et al.: Document analysis as a qualitative research method. Qual. Res. J. **9**(2), 27 (2009)
34. Corbin, J., Strauss, A.: Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory. Sage publications, London (2014)
35. Yin, R.: Case Study Research and Applications: Design and Methods. Sage publications, London (2017)
36. DSDM consortium and others: The DSDM Agile Project Framework Handbook. DSDM Consortium, Ashford (2014)
37. Farshidi, S., Jansen, S., Fortuin, S.: Model-driven development platform selection: four industry case studies. https://doi.org/10.17632/fbg29x5vkk.1 (2020)
38. Software Engineering Standards Committee et al.: IEEE standard for software maintenance. IEEE Std, pp. 1219–1998 (1998)
39. Samadhiya, D., Wang, S.-H., Chen, D.: Quality models: role and value in software engineering. In: 2010 2nd International Conference on Software Technology and Engineering, vol. 1, pp. V1-320. IEEE (2010)
40. Montibeller, G., Winterfeldt, D.: Cognitive and motivational biases in decision and risk analysis. Risk Anal. **35**(7), 1230–1251 (2015)
41. Jones, S.R.G.: Was there a hawthorne effect? Am. J. Sociol. **98**(3), 451–468 (1992)
42. Nadeau, R., Cloutier, E., Guay, J.-H.: New evidence about the existence of a bandwagon effect in the opinion formation process. Int. Polit. Sci. Rev. **14**(2), 203–213 (1993)
43. Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J.: Principles behind the agile manifesto. Agile Alliance, pp. 1–2 (2001)
44. Pressman, R.S.: Software Engineering: A Practitioner's Approach. Palgrave macmillan, New York (2005)
45. Asadi, M., Ramsin, R.: MDA-based methodologies: an analytical survey. In: European Conference on Model Driven Architecture-Foundations and Applications, pp. 419–431. Springer (2008)
46. Embley, D.W., Liddle, S.W., Pastor, O.: Conceptual-model programming: a manifesto. In: Handbook of Conceptual Modeling, pp. 3–16. Springer (2011)
47. Kahneman, D., Slovic, S.P., Slovic, P., Tversky, A.: Judgment Under Uncertainty: Heuristics and Biases. Cambridge University Press, Cambridge (1982)
48. Tversky, A., Kahneman, D.: Judgment under uncertainty: heuristics and biases. Science **185**(4157), 1124–1131 (1974)
49. Gigerenzer, G., Selten, R.: Bounded Rationality: The Adaptive Toolbox. MIT Press, Cambridge (2002)
50. Simon, H.A.: A behavioral model of rational choice. Q. J. Econ. **69**(1), 99–118 (1955)
51. Gigerenzer, G., Todd, P.M.: Simple Heuristics that Make Us Smart. Oxford University Press, Oxford (1999)
52. Fitzgerald, D.R., Mohammed, S., Kremer, G.O.: Differences in the way we decide: the effect of decision style diversity on process conflict in design teams. Personal. Individ. Differ. **104**, 339–344 (2017)
53. Kaufmann, L., Kreft, S., Ehrgott, M., Reimann, F.: Rationality in supplier selection decisions: the effect of the buyer's national task environment. J. Purch. Supply Manag. **18**(2), 76–91 (2012)
54. Farshidi, S., Jansen, S., de Jong, R., Brinkkemper, S.: A decision support system for software technology selection. Journal of Decision Systems, 27(sup1), pp. 98–110 (2018)
55. Farshidi, S., Jansen, S., España, S., Verkleij, J.: Decision support for blockchain platform selection: Three industry case studies. IEEE Transactions on Engineering Management (2020)
56. Farshidi, S., Jansen, S., van der Werf, J.M.: Capturing software architecture knowledge for pattern-driven design. J. Syst. Softw. (2020)
57. Sanchis, R., García-Perales, Ó., Fraile, F., Poler, R.: Low-code as enabler of digital transformation in manufacturing industry. Applied Sciences **10**(1), 12 (2020)
58. Vugec, D.S., Stjepić, A.-M., Sušac, L.: Business process management software functionality analysis: supporting social computing and digital transformation. ISSN 2671-132X Vol. 1 No. 1 pp. 1-876 June 2019, Zagreb, page 547 (2019)
59. Sattar, N.A.: Selection of low-code platforms based on organization and application type. Master's thesis, Business and Management, Lappeenranta University of Technology, Finland (2018)
60. Zolotas, C., Chatzidimitriou, K.C., Symeonidis, A.L.: Restsec: a low-code platform for generating secure by design enterprise services. Enterp. Inf. Syst. **12**(8–9), 1007–1033 (2018)
61. de Oliveira Melo, C., Moraes, J., Ferreira, M., da Costa Figueiredo, R.M.: A method for evaluating end-user development technologies. In: Organizational Transformation and Information Systems (SIGORSA) (2017)
62. Hendriks, D., Hoppenbrouwers, S.J.B.A., van Bommel, P.: The selection process of model based platforms. Master's thesis, Radboud University Nijmegen, the Netherlands (2017)
63. Wasilewski, A.: Business process management suite (BPMS) market changes 2009–2015. Inf. Syst. Manag. **5**, 585–592 (2016)
64. Şen, A.Y., Semiz, N., Güneş, B., Algül, D., Gergin, Z., Dönmez, N.D.: The selection of a process management software with fuzzy topsis multiple criteria decision making method. In: The International Symposium for Production Research, pp. 150–167. Springer (2018)
65. Meidan, A., García-García, J.A., Escalona, M.J., Ramos, I.: A survey on business processes management suites. Comput. Stand. Interfaces **51**, 71–86 (2017)

66. Vukšić, V.B., Brkić, L., Baranović, M.: Business process management systems selection guidelines: theory and practice. In: 2016 39th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), pp. 1476–1481. IEEE (2016)
67. Delgado, A., Calegari, D., Milanese, P., Falcon, R., García, E.: A systematic approach for evaluating BPM systems: case studies on open source and proprietary tools. In: IFIP International Conference on Open Source Systems, pp. 81–90. Springer (2015)
68. Mejri, A., Ghanouchi, S.A., Martinho, R.: Evaluation of process modeling paradigms enabling flexibility. Procedia Comput. Sci. **64**, 1043–1050 (2015)
69. Marín, B., Salinas, A., Morandé, J., Giachetti, G., de la Vara, J.L.: Main features for MDD tools: an exploratory study. In: International Conference on Model-Driven Engineering and Software Development, pp. 183–196. Springer (2014)
70. Ravasan, A.Z., Rouhani, S., Hamidi, H.: A practical framework for business process management suites selection using fuzzy topsis approach. ICEIS **3**, 295–302 (2014)
71. Davies, I., Reeves, M.: BPM tool selection: the case of the Queensland court of justice. In: Handbook on Business Process Management, vol. 1, pp. 339–360. Springer (2010)
72. Kapteijns, T., Jansen, S., Brinkkemper, S., Houët, H., Barendse, R.: A comparative case study of model driven development vs. traditional development: the tortoise or the hare. From code centric to model centric software engineering: Practices, Implications and ROI, 22 (2009)
73. Štemberger, M.I., Bosilj-Vukšić, V., Jaklić, M.I.: Business process management software selection: two case studies. Econ. Res. **22**(4), 84–99 (2009)

Siamak Farshidi is a postdoctoral researcher at the University of Amsterdam (UvA). His research interests lie primarily in the area of knowledge engineering, conceptual modeling, and software architecture. Farshidi received his Ph.D. in computer science from Utrecht University, based on the work entitled "Multi-Criteria Decision-Making in Software Production".

Slinger Jansen is an assistant professor at the Department of Information and Computer Science at Utrecht University. His research focuses on software product management and software ecosystems, with a strong entrepreneurial component. Jansen received his Ph.D. in computer science from Utrecht University, based on the work entitled "Customer Configuration Updating in a Software Supply Network".

Sven Fortuin is a strategy and business developer at bol.com (located in the Netherlands), the leading b2c e-commerce retailer in the Benelux. He graduated from Utrecht University with a B.Sc. In Information Sciences and an M.Sc. in Business Informatics, and an M.Phil. in Technology Policy from Cambridge Judge Business School. During his M.Sc. in Business Informatics, he extensively researched model-driven software development for a year while writing his master thesis at AFAS Software.