UTRECHT UNIVERSITY

MASTER'S THESIS

CONCEPTUAL FRAMEWORK ASSESSING A BPM INITIATIVE FOR LOW-CODE DEVELOPMENT PLATFORM SUITABILITY

by Kirill Sadovnikov

DEPARTMENT OF INFORMATION AND COMPUTING SCIENCES

UTRECHT, 4TH OF DECEMBER 2022

FIRST SUPERVISOR: Dr. G.C. (Inge) van de Weerd

SECOND SUPERVISOR: Dr. ir. J.M.E.M. (Jan Martijn) van der Werf

EXTERNAL SUPERVISOR: R. (Robbert) Sweijen

Utrecht University

KPMG

Abstract

The digital transformation has forced organizations to digitalize their business processes in order to stay innovative and competitive. However, organizations also face many uncertainties when deciding what technology to employ in the abundance of choice. Low-code development platforms (LCDPs) promise time and cost reduction through rapid and easy-to-use application assembly, and are seen as major facilitator during a digital transformation. Therefore, we propose a conceptual framework for organizations to assess whether their business process management (BPM) initiative to digitalize business processes is suitable for LCDP support. The framework is developed through a study of literature, a focus group, and expert interviews, resulting in 18 factors to be considered by organizations. An evaluation using fictitious use case analyses showed that the model was well-received, especially with regard to its completeness and operationality. To the best of our knowledge, this is the first work studying organizational adoption of low-code for the sake of BPM.

Acknowledgements

This thesis marks the final part to achieve my degree of Master of Science in Business Informatics. I can say that I am very proud of the final result and the whole process that went into it. For this, I would like to thank many people.

First of all, I would like to thank Dr. Inge van de Weerd for guiding me throughout this thesis. Her excellent feedback and advice, openness and flexibility to new ideas, and availability at any moment have given me all the tools and motivation during the course of this project. I have learned a lot and I could not be happier with her as my supervisor, I am very thankful for that.

I would also like to thank Robbert Sweijen for the opportunity to do this research project at KPMG under his supervision. Many elements of this research would not have been possible without his comments and ideas. Thereby, I would also like to thank my colleagues at the Digital Process Excellence team for making my time there as enjoyable as it was.

Furthermore, I want to thank Dr. Jan Martijn van der Werf for being my second supervisor and co-author of our scientific paper. His feedback and advice have taught me a lot and have greatly benefited my work.

Of course, I wish to thank all the participants in this research, some of which I rudely interrupted during their summer holidays with a message on LinkedIn. It was a great experience speaking to many experts in various fields, sharing opinions and experiences, and I hope the final result will also be useful to them in their lines of work.

Lastly, I would like to thank my friends, girlfriend, and family for their support, feedback, and ideas, but also for their company during many, many coffee breaks in the library.

Kirill Sadovnikov

Utrecht, 4th of December 2022

Table of contents

1 Introduction

The widespread adoption of digital technology by society has transformed businesses in all industries (van Veldhoven & Vanthienen, 2022). The technological demands of customers and employees have risen and corporations that fail to adapt could be surpassed by innovative, digital start-ups (Konopik, Jahn, Schuster, Hossbach, Pflaum, 2022). This phenomenon was introduced already in 2000 as the 'Digital Transformation' by Patel and McCarthy (2000). Lately, with disruptive events such as the COVID-19 pandemic further accelerating the shift to digital technologies, organizations have no choice but to adapt and implement digital transformation strategies (Kraus, Durst, Ferreira, Veiga, Kailer & Weinmann, 2022).

Digital transformation goes beyond the sole adoption and implementation of digital technologies in an organization (Konopik et al., 2022). In order to respond to continuously changing demands and remain competitive, organizations also have to digitalize their business processes (Denner, Püschel & Röglinger, 2018). Digitalizing business processes involves aligning the business processes and the organization's information technology (IT) infrastructure through the use of digital technology to automate business operations and improve flexibility and responsiveness. (Lin & Hsia, 2011; Imgrund, Fischer, Janiesch & Winkelmann, 2018). It is, however, no easy feat and organizations face many uncertainties when deciding what technology to employ in the abundance of choice (Ackx, 2014). Moreover, research fails to provide guidelines on how organizations should approach process digitalization in times of digital transformation (Imgrund et al., 2018).

Organizations can refer to Business Process Management (BPM) for methods, techniques, and tools to support such initiatives (Dumas, La Rosa, Mendling & Reijers, 2018). BPM has matured as a research discipline and has proven to successfully help organizations in improving and innovating business processes that add value to an organization and its customers (vom Brocke, Zelt & Schmiedel, 2016). For an organization wanting to digitalize large, end-to-end processes in a flexible way, Business Process Management Systems (BPMSs) are proposed as an effective IT solution during digital transformation (Xu, Xu & Li, 2018; Brkić, Tomičić Pupek & Bosilj Vukšić, 2020). Dumas et al. (2018) define BPMSs as "*systems that support the design, analysis, execution, and monitoring of business processes on the basis of explicit process models*". These functionalities enable organizations to orchestrate a process, reducing the employees' workload, and increasing process performance, compliance, and adaptability (Štemberger, Bosilj Vukšić & Jaklić, 2009; Ravasan, Rouhani & Hamidi, 2014; Dumas et al., 2018).

Organizations undertaking a BPM initiative to implement a BPMS face a decision of whether to build it in-house or to acquire a packaged solution from a vendor. This is known as the traditional 'build vs. buy' decision (McManus, 2003; Ravesteyn & Batenburg, 2010). The general consensus is that strategy-specific systems, that increase an organization's competitive advantage, should be built in-house, and

packaged solutions are efficient for generic applications and processes (McManus, 2003). Recently, a new option has emerged. These platforms, called low-code development platforms (LCDPs), are characterized by the newly coined 'low-code' method where traditional coding is replaced with drag-and-drop assembly and modeling of information systems (Richardson & Rymer, 2014; Sahay, Indamutsa, Di Ruscio & Pierantonio, 2020; Sanchis, García-Perales, Fraile & Poler, 2020; Vincent et al., 2020). These platforms are becoming increasingly popular as they promise to combine the flexibility of building a solution, and the efficiency of buying one (Cicman, Bratincevic & Rymer, 2021).

1.1 Problem statement

LCDPs are seen as major facilitators for organizations going through a digital transformation due to their rapid and flexible development capabilities (Sanchis et al., 2020). Moreover, a lot of LCDPs offer process and workflow automation functionalities (Luo, Liang, Wang, Shahin & Zhan, 2020) allowing the construction of a BPMS. Therefore, using low-code to digitalize business processes would seem like an obvious solution for organizations.

Nevertheless, low-code development is not suitable for every problem, or any organizational setting (Frank, Maier & Bock, 2021). Moreover, many decision-makers struggle in evaluating and selecting the most effective digital solutions that can help their organization to advance its business processes (Denner et al., 2018; Imgrund et al., 2018). A lack of understanding and distrust in LCDP capabilities has also already hampered its adoption (Alsaadi, Radain, Alzahrani & Alshammari, 2021). Therefore, in order to stay innovative and competitive, decision-makers need guidance in assessing how effective low-code will be for their BPM initiative to digitalize business processes. However, research to date has not yet found the conditions when an LCDP can be a suitable solution for organizations (Bock and Frank, 2021a).

Therefore, the objective of this thesis is to close this gap for LCDPs aimed at BPMS development, hereafter referred to as 'low-code BPM'. We propose a conceptual framework for organizations to assess the suitability of their BPM initiative to be supported by low-code BPM. This should facilitate organizations to make an assured decision on low-code BPM adoption.

1.2 Research context

This study was performed in cooperation with the Digital Process Excellence team at KPMG Advisory N.V. (hereafter referred to as 'KPMG'). This team provides management consultancy services for a wide range of customers with the goal of improving business processes and achieving operational excellence through technologies such as robotic process automation (RPA), process mining, and chatbots. Recently, the number of cases concerning low-code BPM had begun to rise and this posed the question: when do we develop using low-code BPM and when do we opt for another solution? This

question motivated this study to come up with a scientifically-based framework for organizations considering a low-code solution for their BPM initiative.

1.3 Outline

This thesis is structured as follows. Chapter 2 begins by laying out the theoretical background of the research regarding low-code, model-driven engineering, and BPMS. In Chapter 3, we present our research method. In Chapters 4 to 6, we illustrate our findings and how our proposed framework has been constructed. The evaluation results follow in Chapter 7. We discuss all our results, their implications, and limitations in Chapter 8. Lastly, we conclude our study in Chapter 9.

2 Theoretical background

LCDPs used to build a BPMS, which we call 'low-code BPM', are the main object of study in this thesis. Nevertheless, current literature lacks a single, clear definition of 'low-code' or the differentiation of LCDPs from other technologies (Frank et al., 2021). In order to scope low-code, this chapter presents our conceptualization. Thereafter, we elaborate on how and why organizations have opted to adopt low-code, or what limitations have deterred them. Then, we explain why we can draw a comparison to Model-Driven Engineering (MDE) in order to analyze low-code's characteristics. Lastly, we explain how contextual considerations surrounding BPM projects affect our proposed framework.

2.1 Conceptualization of low-code BPM

With a clear definition for 'low-code' lacking, terms such as low-code platform (LCP), low-code application platform (LCAP), and low-code development platform (LCDP) are used interchangeably throughout studies (Bock & Frank, 2021a). Nowadays, a broad variety of solutions position themselves on the market for being low-code, while offering diverse products. Noticeably, many of these solutions already existed before the term low-code became prominent. In an analysis of low-code vendors, Bock and Frank (2021a) found that previous products were offered as, among others, 'platform as a service' (PaaS), as a BPM tool, or as a 'model-driven application platform'.

Therefore, we have analyzed literature to provide our own conceptualization. Generally, we found that LCDPs consist of three main functionalities: (1) they allow the modeling of (data) system structures and business processes, (2) they provide capabilities to design custom graphical user interfaces (GUIs), and (3) they offer flexible integration with external systems (Sahay et al., 2020; Vincent et al., 2020; Bock & Frank, 2021a). LCDPs facilitate this through low-code, an approach where standardized, high-level functional components can be assembled easily in a visual designer, instead of textual coding (Metrôlho, Araújo, Ribeiro & Castela, 2019; Sahay et al., 2020; Sanchis et al., 2020).

Furthermore, Frank et al. (2021) present a classification of LCDPs into four groups: 'basic data management platforms', 'workflow management systems', 'extended GUI- and data-centric IDEs', and an all-encompassing fourth 'multi-use platforms' group. The first group offers features found mostly in data management systems and the third focuses on developing web- or mobile applications, both outside of our scope. As BPMSs were originally known as workflow management systems (Dumas et al., 2018), we consider low-code BPM to fall under the 'workflow management systems' group. 'Multi-use platforms' with an extensive BPM focus are also included as these incorporate workflow management system characteristics.

Frank et al. (2021) propose that these platforms specialize in workflow automation, through conceptual modeling language like Business Process Model and Notation (BPMN) or other structures. Moreover,

we found that these platforms support end-to-end case management, through a workflow engine that governs the users' interaction, and provide integrations with internal and external systems (Alsaadi et al., 2021; Bock & Frank, 2021b). The latter can be in the organization's IT landscape or AI (Frank et al., 2021), machine learning (Koplowitz, 2017), or other automating services that the platform provides. Lastly, low-code BPM provides functionality to build analytics dashboards to monitor process performance (Waszkowski, 2019; Alsaadi et al., 2021).

We combine overall LCDP features with 'workflow management system' specializations to define low-code BPM in this study as: *Low-code BPM supports rapid application development for end-to-end case management, with the workflow model and engine at its core, enabling process automation, integration, monitoring and enhancement with intelligent automation technology through easy-to-use component assembly and modeling*.

2.2 Low-code adoption incentives and deterrents

LCDPs have seen a fast rise in popularity in the recent couple of years (Frank et al., 2021). Gartner, a renowned technological research firm, expects the LCDP market to reach $29 billion in revenue in 2025 as enterprises will use low-code technologies for 70% of their new applications (Wong, Iijima, Leow, Jain & Vincent, 2021). A survey by Luo et al. (2021) demonstrated that LCDPs are frequently used to develop frontend applications, to integrate systems in an IT landscape, but also for business process automation. Few studies have yet investigated the latter use. Cai, Huang, Kessler, and Fottner (2022) have developed a low-code implementation method and demonstrated it by automating a part of a business process. It has shown that automating business processes is possible through low-code and it reduced manual workload, and improved IT flexibility. Waszkowski (2019) describes the design of a "BPM low-code platform" that could support a business process as a BPMS does. However, literature on low-code is still scarce (Sanchis et al., 2020) and no research to date has yet determined when an organization's initiative to construct a BPMS is suited for low-code BPM (Frank et al., 2021).

Several studies have provided evidence of low-code's benefits and disadvantages in general. The main advantage of LCDPs is that it allows users to build applications by dragging and dropping pre-built functional modules and minimizes the need for coding (Adrian, Hinrichsen & Nikolenko, 2020; Sanchis et al., 2020). This enables a range of opportunities.

Firstly, it enables end users without a programming background (commonly called *citizen developers*) to be involved with application development (Sahay et al., 2020). This allows citizen developers, with in-depth subject matter knowledge, to build and tailor the applications exactly to the required system functionality (Adrian et al., 2020; Iho, Krejci & Missonier, 2021). Moreover, this suits a growing need for application developers to tackle a rising demand for business applications (Metrôlho et al., 2019). It would allow professional developers to focus on other application management tasks such as installation and deployment (Alsaadi et al., 2021).

Secondly, relying on prebuilt functional modules increases the speed and adaptability of software development (Metrôlho et al., 2019; Sahay et al., 2020; Sanchis et al., 2020; Frank et al., 2021; Luo et al., 2021). For example, a portal tracking the COVID-19 disease in New York City was set up in three days using low-code (Woo, 2000). Vendors estimate that application delivery through LCDPs can be up to 10 times faster than usual application development (Iho et al., 2021).

Furthermore, cost reduction (Alsaadi et al., 2021; Frank et al., 2021; Luo et al., 2021), increased maintainability (Sahay et al., 2020; Bock & Frank, 2021a; Alsaadi et al., 2021), and improved system quality (Luo et al., 2021) have been cited as other potential benefits. Sanchis et al. (2020) have found that the top three reasons for organizations to adopt low-code are the acceleration of digital transformation, an increase in responsiveness to the business, and the reduction of dependency on hard-to-hire technical skills.

Literature also discusses the shortcomings of low-code. From a technical perspective, LCDPs have been seen struggling with scalability issues, lack of customization on design and layout, and lack of interoperability between LCDPs (Alsaadi et al., 2021; Luo et al., 2021). Woo (2000) argues that applications developed without software development professionals may be difficult to maintain, inefficient performance-wise, and insecure. For citizen developers, a steep learning curve has also been a common challenge (Sahay et al., 2020; Luo et al., 2021). Organizations have disregarded LCDPs most commonly due to a lack of low-code knowledge, distrust in LCDP's capabilities to construct the required solution, and concerns about 'lock-in' with a low-code vendor (Sanchis et al., 2020; Alsaadi et al., 2021).

2.3 Comparison to MDE

As previously stated, current LCDPs allow to model data- and system structures and have previously been seen as 'model-driven application platforms' (Sahay et al., 2020; Bock & Frank, 2021a). Studies have argued that MDE[1] forms the origins of low-code development. MDE is described as a software development methodology that uses models at the core of the whole development process (Kent, 2002; Staron, 2006). In MDE, models are used to define the workings of an application and, in practice, the whole system development process is based on models. The concept was seen as a solution for the increasing industry-wide problem of software complexity – the abstract models eliminate complexity and the automation eases the complexity of software development (Mohagheghi & Dehlen, 2008). As low-code development also has a high dependency on models, Cabot (2020) strongly argues that low-code does not have any technical contribution as opposed to MDE and is merely 'sold' differently.

[1] Model-Driven Engineering (MDE) is only a part of the whole Model Driven Architecture (MDA) proposed by the Object Management Group (OMG), as explained in Kent (2002). Other terminology, such as Model-Driven Development (MDD) or Model-Based Engineering/Development (MBE/MBD), exists referring to corresponding concepts (Cabot, 2020). However, as this methodology is not the main subject in this study, we use MDE for the remainder of this study.

Similarly, recent studies have described LCDPs as being based on model-driven design (Alsaadi et al., 2021) and following model-driven engineering principles (Sahay et al., 2020). Assuming low-code originates from MDE, we use this as a historical lens to analyze MDE's successes and failures.

Great benefits in productivity, software quality, maintainability, and interoperability were expected if MDE were to be applied in organizations (Mohagheghi & Dehlen, 2008; Hutchinson, Whittle & Rouncefield, 2014). In the early years of MDE, incremental productivity, standardization, and quality gains were found in case studies. However, the results were far from convincing and the gains did not live up to expectations (Mohagheghi & Dehlen, 2008).

Organizational and managerial factors were found to play a big role in MDE adoption. Top management frequently refused the transition to a model-driven approach as it would imply investing in the redefinition of already functional systems without adding new functionality (Hutchinson et al., 2014). Moreover, the new approach required developers to significantly change their institutionalized software development approach, which was met with resistance (Aranda, Damian & Borici, 2012). At the same time, the lack of appropriate tooling affected MDE's adoption in the industry throughout its lifetime. Early indications found that MDE tools were not able to support the whole software development process, minimizing its effectiveness (Mohagheghi & Dehlen, 2008). Although tooling developed over the years, the maturity and capability of the tools still did not convince organizations to adopt (Aranda et al., 2012; Bucchiarone, Cabot, Paige & Pierantonio, 2020). Whittle, Hutchinson, Rouncefield, Burden, and Heldal (2017) have summarized all the above MDE tool-related issues into a taxonomy, where the factors span four themes (technical, internal organizational, external organizational, social). To sum up, the overall benefits of MDE have never outweighed its costs (Cabot, Clarisó, Brambilla & Gérard, 2017).

Seeing the technical and organizational difficulties endured in MDE adoption, we wonder whether similar issues will affect low-code. With the continuous development of various LCDPs by acclaimed technical companies such as Microsoft and Salesforce (Vincent et al., 2020), the technical factors hampering MDE could, possibly, be solved for low-code. Nonetheless, decision-makers need to carefully assess whether low-code's benefits outweigh the disadvantages in their organization.

2.4 Context awareness for low-code BPM

Thus far, we have found that LCDPs have their benefits and disadvantages, and are not suitable for all use cases. According to MDE literature, there are various technical, organizational, external, and social factors affecting successful adoption (Whittle et al., 2017). In regards to BPM, vom Brocke, Zelt, and Schmiedel (2016) argue that the context of the BPM initiative is essential to consider as well. For BPM projects to be efficient and productive, situational conditions of the organization and its surroundings need to be considered. In BPMS adoption literature, the importance of alignment between the BPMS and the organization's use case has also been cited (Ravasan et al., 2014; Bosilj Vukšić, Brkić &

Tomičić Pupek, 2018). Without context awareness, BPM projects are prone to fail (vom Brocke & Schmiedel, 2014). Therefore, vom Brocke et al. (2016) propose a framework to elicit the context in which BPM is applied through four dimensions (goal, process, organizational, and environmental). Factors falling under these dimensions allow organizations to decide how to approach their BPM initiative.

We use the taxonomy of Whittle et al. (2017) and the framework of vom Brocke et al. (2016) as a foundation to search for factors that are vital to assess when considering low-code BPM. Research has been performed to rate the organization's suitability for other BPM-related technologies, such as RPA (Plattfaut, Borghoff, Godefroid, Koch, Trampler & Coners, 2022) or process mining (Mamudu, Bandara, Wynn & Leemans, 2022), but never for low-code BPM. We aim to close that gap with our conceptual framework.

3 Research methodology

Our conceptual assessment framework has been constructed following the Design Science approach, as it "*creates and evaluates IT artifacts intended to solve identified organizational problems*" (Hevner, March, Park & Ram, 2004, p. 77). The 'IT artifact' in this study is the proposed assessment framework and the 'organizational problem' has been illustrated in the previous chapters and by the questions posed by KPMG. Additionally, we chose design science because it allows for iterative build-and-validate loops (Hevner et al., 2004). Hereby, the IT artifact is validated and subsequently rebuilt through multiple cycles. This is to improve the artifact quality, but also increase knowledge about the problem. With the scarcity of scientific literature concerning low-code (Sanchis et al., 2020), we want to extract both artifact quality and theoretical knowledge.

In the next subchapter, we describe the various phases in our methodology. Thereafter, we discuss the approach to analyze the data resulting from our research. Lastly, we describe the measures taken to mitigate possible threats to validity.

3.1 Design science research methodology

We have used the Design Science Research Methodology (DSRM) by Peffers, Tuunanen, Rothenberger, and Chatterjee (2007) to structure our methodology. Figure 1 shows how the DSRM is applied in our methodology. In the following sub-chapters, we present the research questions and the evaluation criteria, encompassing the first two phases in the DSRM. Thereafter, we show how our framework is initially constructed based on literature, and then empirically validated through build-and-validate loops as the third phase in the DSRM. Finally, we explain our approach to demonstrate and evaluate the framework. The 'Communication'-phase is not explicitly discussed. This thesis and the final framework present our communication approach. Moreover, a scientific paper has been submitted to the 31st European Conference on Information Systems (ECIS). This paper can be found in Appendix I.

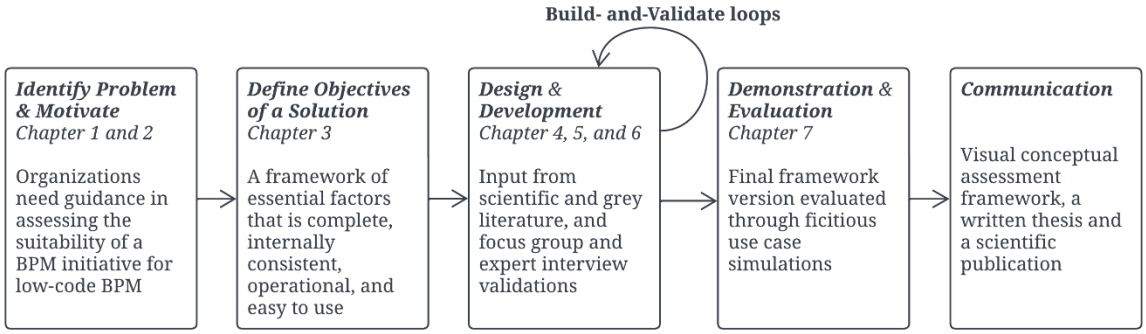


Figure 1. The Design Science Research Methodology (Peffers et al., 2007) adapted to our study

3.1.1 Research questions

Every design science study begins with identifying the problem and what value a new design solution would pose (Peffers et al., 2007). The introduction in Chapter 1 already presented the organizational problem, thus, only a brief summary is presented here.

Organizations across all industries are forced to cater to changing and increasing customers' and employees' demands for modern and innovative technologies (Konopik et al., 2022). This is called the digital transformation, which does not only entail implementing modern technologies, as organizations also have to digitalize their business processes (Denner et al., 2018). BPMSs could be a potential solution to sustain large, end-to-end processes (Xu et al., 2018; Brkić et al., 2020) and, usually, these are either built in-house or acquired as a packaged solution (McManus, 2003). However, the current emergence of LCDPs, which offer flexibility, cost-effectiveness, and quick time-to-market, serves as an interesting alternative (Frank et al., 2021). To help organizations understand and identify the digital solutions that can advance their business processes (Denner et al., 2018), we study under what conditions low-code BPM platforms are an effective solution. Therefore, the following research question is stated:

> ***RQ**: "How can organizations assess whether a low-code development platform is suitable to support their business process management initiative?"*

The intended BPM initiative in our research question implies the digitalization of business processes in an organization through the implementation of a BPMS.

We answer the main research question through two sub-questions. Firstly, we need to identify what aspects of a BPM initiative organizations should assess. Initially, we consulted existing literature for this. Due to the novelty of low-code BPM, both literature from comparable research areas and grey literature were consulted. Thereafter, our initial findings from literature were empirically validated through two build-and-validate loops. The sub-question posed is:

***SQ1**: "What characteristics of a business process management initiative are essential when assessing the suitability of low-code BPM?"*

Secondly, we want to create a practical framework that is useful and usable by organizations to do an assessment. Therefore, we constructed a framework that integrates all the findings above. Thereafter, we validated the framework's design iteratively to come to a final version. To see how our framework performs in use, we employed an empirical evaluation with intended end users to measure predefined evaluation criteria and ask:

***SQ2**: "How does the created framework help organizations to assess the suitability of low-code BPM?"*

3.1.2 Framework objectives

In the second phase of the DSRM, Peffers et al. (2007) emphasize the importance of stating the objectives of the artifact before the evaluation.

The evaluation objectives for our conceptual assessment framework are derived from criteria specified by March and Smith (1995). They present possible evaluation criteria for four possible outputs of design science: constructs, model, method, and instantiation. As our framework is both a model to represent BPM initiative characteristics and a method to be used in an assessment, we use evaluation criteria for these outputs. The chosen criteria are centered around its purpose: Our framework should provide a complete and consistent picture of all BPM initiative characteristics that differentiate low-code BPM from other solutions, and should enable organizations to assess the effectiveness of low-code BPM for their BPM initiative through an easy-to-use framework. As March and Smith (1995) do not provide clear definitions, we refer to the definitions found in Prat, Comyn-Wattiau, and Akoka (2015). The evaluation criteria for our conceptual assessment framework are presented in Table 1.

Criteria (March & Smith, 1995)	**Definition** (Prat et al., 2015)
Completeness	The degree to which the activity of the artifact contains all necessary elements and relationships between elements.
Internal consistency	The degree of uniformity, standardization, and freedom from contradiction among the elements of the activity of the artifact.
Operationality	The ability to perform the intended task or the ability of humans to effectively use the method.
Ease of use	The degree to which the use of the artifact by individuals is free of effort.

Table 1. The evaluation criteria for the low-code BPM assessment framework

3.1.3 Framework construction

In the third phase of the DSRM, we design and develop our framework in several iterations. This sub-chapter describes how the initial framework version had been constructed. In the following sub-chapter, we elaborate upon the iterative validations.

The first version of the conceptual assessment framework is based on a structured literature research. As literature on low-code (BPM) is still scarce (Sanchis et al., 2020), we have included relevant studies on BPM and MDE to further increase the reliability of our literature review on low-code BPM. Google Scholar and Semantic Scholar were used as search engines. The primary search terms revolved around the term ‘low-code’, combined with ‘development’, ‘platform’, ‘adoption’, ‘factors’, ‘process automation’, ‘BPM’, ‘BPMS’, and ‘MDE’. For MDE and BPMS literature we combined ‘MDE’ or ‘BPMS’ with ‘adoption’, ‘development’, ‘organization’, and ‘effects’. This resulted in the first set of literature. While analyzing, relevant references were used to find other papers through backward snowballing. We also used forward snowballing on the literature from our dataset to find papers of interest.

Certain selection criteria were considered when constructing our literary dataset. Firstly, the work had to be scientifically peer-reviewed to be considered. Due to its scarcity, most literature on low-code was taken into account when it was deemed relevant based on the abstract. For MDE and BPM literature, the criteria were more strict, and preferable widely-cited papers were chosen. In the case of MDE, the literature had to 'relate to the model-driven methodology and explain its (organizational) implementation effects'. In the case of BPMS, the literature had to 'relate to the concept of BPMS implementation and its (organizational) effects'. All in all, the resulting set of scientific literature consisted of 29 studies.

We decided to also include grey literature in the study. Many acclaimed market research firms have been analyzing the advancement of low-code in recent years. With their focus on the industry adoption of low-code, and a lack of scientific literature, these reports provide rich data that can serve as input to answer the sub-questions. Only reports from renowned market research firms, large vendors, consultancy firms, or other implementation partners were selected. This resulted in 12 grey literature documents.

Lastly, an exploratory interview was conducted with an Intelligent Automation consultant at KPMG acquainted with low-code development. This interview served as a confirmation and exploration of the identified problem and helps the triangulation of this study. Gathering data from multiple sources provides the credibility of results (Bowen, 2009). As the objective of the interview is exploratory, based on existing findings, a semi-structured interview was chosen (Wohlin, Runeson, Höst, Ohlsson, Regnell & Wesslén, 2012). The exploratory interview protocol is presented in Appendix A.

The resulting dataset for the initial framework version resulted in 41 scientific and grey literature documents and the exploratory interview. These were analyzed and integrated into an initial framework version.

3.1.4 Framework validation

To further validate and strengthen the found concepts, we triangulate our findings through two empirical build-and-validate loops (Kaplan & Maxwell, 2005). Our chosen approach applies concepts from the 'Framework for Evaluation in Design Science' (FEDS) by Venable, Pries-Heje, and Baskerville (2016). Although the framework's evaluation is presented in Chapter 3.1.5, it also applied the FEDS and is briefly mentioned in this sub-chapter as well.

The FEDS is specifically tailored for design science where it defines two axes of an evaluation strategy. Firstly, the functional purpose of the evaluation is divided between formative and summative evaluations. In short, formative evaluations focus on how the artifact is experienced during use while summative evaluations focus on the outcomes of using the artifact. Secondly, the paradigm of the evaluation explains in what environment the evaluation is taking place. Naturalistic evaluations explore

the artifact in its actual environment while artificial evaluations are often laboratory experiments or simulations to test hypotheses.

Venable et al. (2016) sketch various possible strategies for navigating the dimensions based on the artifact's goal. Following the circumstance selection criteria found in Venable et al. (2016, Table 1), a social- and user-oriented artifact, whose benefits in the real world need to remain for a long time, tends towards the 'Human Risk & Effectiveness' evaluation strategy. We chose this strategy as our framework serves an organizational problem and it should be suitable for decision-makers in organizations. A 'Human Risk & Effectiveness' evaluation strategy dictates that the first validation should be artificial and formative in nature. Promptly, the artifact should be moved to its natural scenario with its intended users and be thoroughly tested through summative evaluations. We decided the validation methods, and the evaluation of our final framework, following this approach. Our strategy, and the possible other strategies, are illustrated in Figure 2. In the next two sub-chapters, we discuss the details of our validation methods.

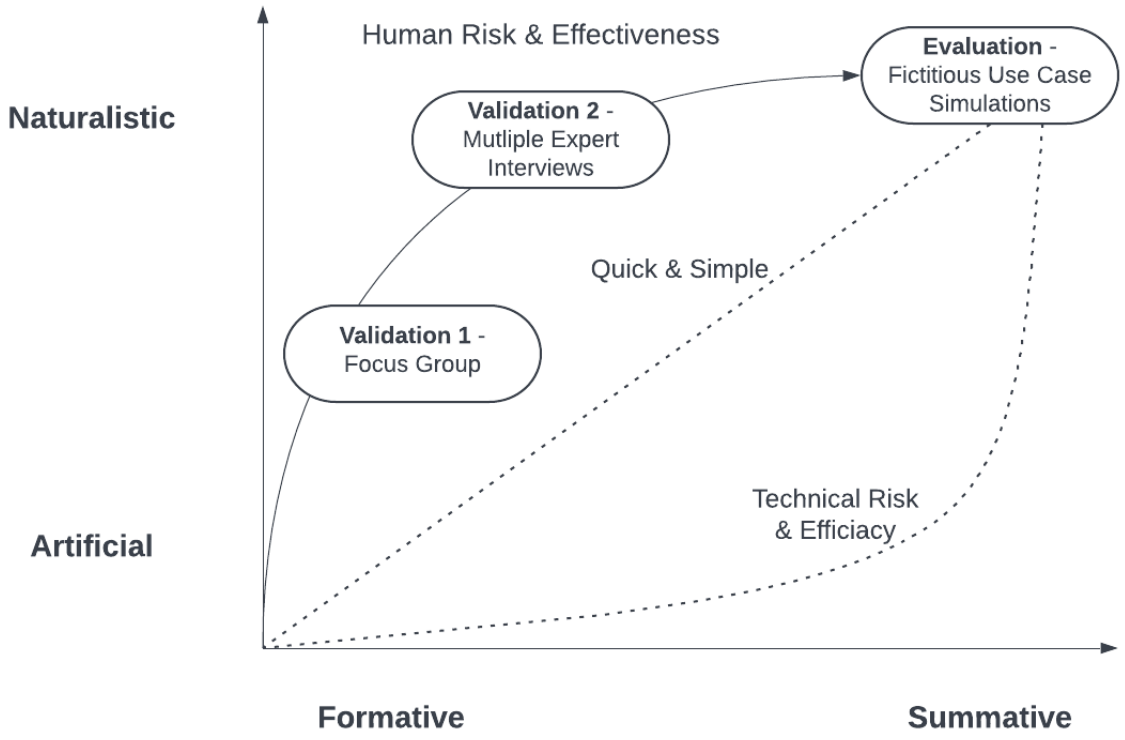


Figure 2. The validation and evaluation approach represented through the FEDS, adapted from Venable et al. (2016)

3.1.4.1 Focus group

The first validation consisted of a focus group with consultants and managers from KPMG. The session's goal was to hear from experts in the field about what aspects organizations ought to consider regarding low-code BPM, and how to framework represented these in a formative way. As the session

was conducted in KPMG, the organization collaborating on this study, this represents an artificial evaluation method. We opted for a focus group since it can explore a range of different ideas and represent various perspectives among a group of people (Krueger & Casey, 2014). This broadness is deemed useful as a first empirical validation. Moreover, an initial pilot test of the framework at KPMG allows the artifact and the organizational problem to align. The protocol for the focus group can be found in Appendix B.

The focus group was conducted according to the methodology by Krueger and Casey (2014). The core question that the focus group had to answer was this study's research question: *How can organizations assess whether a low-code development platform is suitable to support their business process management initiative?* Based on the input from the focus group, we developed a refined framework version.

3.1.4.2 Expert interviews

The second validation consisted of interviews with experts on low-code BPM. Expert interviews enable more in-depth analyses of topics in which the participant has expertise (Meuser & Nagel, 2009). This evaluation method is more naturalistic as the variety of chosen experts stood for the array of possible end users that might use the framework. Moreover, the artifact is becoming more advanced and user-oriented in this phase, allowing for a more summative evaluation according to the FEDS. Criterion sampling is also applied to make sure that the participants have enough experience in the field of low-code BPM. The following criteria were framed:

- An interview participant should have cooperated on at least one implementation of a low-code solution.
- An interview participant should have completed at least a low-code vendor's basic development training.

The protocol for the expert interviews can be found in Appendix C. Based on the input from the expert interviews, we constructed the final version of our framework.

3.1.5 Framework evaluation

The evaluation of the framework is conducted in the fourth phase of the DSRM. We employ a naturalistic and summative approach, as prescribed by the FEDS, through fictitious use case simulations. The goal of these simulations is to evaluate the framework on the evaluation criteria from Table 1. The simulations took place in the Digital Process Excellence team where consultants and managers evaluated the framework by doing an assessment on whether low-code BPM would be a suitable solution for a fictitious organization. This sub-chapter describes the evaluation.

Three real-life, historical KPMG use cases were employed for these simulations. In each use case, an organization had a BPM-related problem for which they sought an IT solution. Two use cases, eventually, resulted in low-code BPM adoption, the other in acquiring a packaged solution.

As part of the preparation, the manager involved in a use case was questioned about the business problem that the organization had faced. Moreover, a use case description explaining this business problem was formulated together with the researchers. As the evaluation participants work in the same team where these use cases were carried out, there is a chance that participants have prior knowledge of the use case. Therefore, various measures were imposed to mitigate this risk. Firstly, the use case descriptions were anonymized together with the use case experts to make them unrecognizable. Secondly, fake logos, organization names, and employee titles were given to further cover the use case. The three use case descriptions can be found in Appendix G. The use case description served as an introduction to the participants during the simulation.

Each simulation consisted of one participant assessing one of the three use cases. We started each simulation by introducing the session, explaining our conceptualization of low-code BPM as the object of study, and then providing the use case description and low-code BPM assessment framework. In total, the participant was given around 10 minutes to familiarize themselves with the use case description and the framework, and think about an initial advice to the fictitious organization. Hereafter, as the use case descriptions were concise, the participant had the opportunity to interview the manager for more in-depth knowledge surrounding the use case. Using the framework, the participants assessed whether low-code BPM would be an effective solution for the fictitious organization. After 15 minutes, the participant had to verbally present their advice on whether the use case is suitable for low-code BPM. The idea of this simulation was to imitate a real-life scenario as closely as possible, where the framework is used by decision-makers. The researchers observed how the participants used the framework and what questions were asked throughout the simulation. Following our evaluation criteria, the final decision taken by the participant on low-code BPM suitability was less important than how they used the artifact.

After each simulation, we interviewed the participant and asked them to fill in a survey. The survey consisted of two questions per evaluation criteria, resulting in eight questions that the participants had to rate on a 5-point Likert scale. The survey questions were based on Aier and Fischer (2011). The fictitious use case simulation protocol and survey can be found in Appendix D. The complete simulation and the interview thereafter were recorded for analysis. The survey also included open questions for any comments or remarks that the participants had regarding the framework. Moreover, the researchers took notes on how the participant used the framework throughout the simulation.

3.2 Data analysis

The exploratory interview, focus group, each expert interview, and each fictitious use case simulation were transcribed in order to be analyzed. Apart from the evaluation survey, this makes our dataset entirely qualitative. Throughout each step of our method, we employed thematic analysis, described by

Braun and Clarke (2006), to iteratively analyze our data and derive our results. This sub-chapter describes our approach.

We used a combination of deductive and inductive coding to derive the BPM initiative characteristics that should be assessed. A deductive coding approach uses a pre-existing framework or theory as a reference to find elements of interest in the dataset (Crabtree and Miller, 1999). For this, we combine the MDE tooling issues themes of Whittle et al. (2017) and the BPM context dimensions of vom Brocke et al. (2016), as presented in the following chapter. Thereafter, we applied an inductive approach to directly code elements in our data with the framework in mind. Subsequently, we searched and reviewed our codebook for themes, categorized them under the dimensions, and gave the themes a definition and a description, according to the methodology by Braun and Clarke (2006). These themes become the content of our conceptual assessment framework.

The analyses of the validations followed a similar procedure. However, we revisited the existing themes from the previous framework version and their substantiation. As we conducted and analyzed each method sequentially, contradictions and discrepancies could occur when integrating results (Moran-Ellis et al., 2006). Therefore, throughout our results, we present our rationale on how we reconciled such discrepancies.

In order to provide transparency in data analysis, the resulting codebooks are illustrated in Appendix E for all our thematic analyses.

3.3 Threats to validity

Various measures have been implemented in our methodology to provide valuable and reliable results. We discuss possible threats to validity, and our approach to mitigate them, following the classification scheme by Yin (2009).

Construct validity relates to the reliable understanding of the studied constructs in the study. In general, low-code's lack of a clear definition poses a threat to construct validity as the scope of this study could be misunderstood by participants. Moreover, LCDPs specifically for BPM could have supplementary characteristics that have never been conceptualized in scientific research. Therefore, we conceptualized 'low-code BPM' for this study in Chapter 2.1 based on the LCDP classification of Frank et al. (2021). We introduced this conceptualization to the participants of each empirical validation and evaluation. A threat to construct validity could also occur when inferring the evaluation results. Using thematic analysis, we measured how well the framework performed on the evaluation criteria. Valuing the evaluation criteria could be highly interpretative. To strengthen this analysis, we also employed a survey where the evaluation criteria were directly measured using validated questions from Aier and Fischer (2011).

Internal validity evaluates whether the causal relationships are valid, and not affected by other variables. The main embodiment of this threat in this study is whether the framework's evaluation accurately measured the evaluation criteria. Naturally, an extensive real-life evaluation of the framework would have been ideal to evaluate the framework. However, this was not achievable during the timeframe of this study, so an alternative was found where employees from the Digital Process Excellence team were asked to evaluate the framework using historical use cases performed in the team. This poses an obvious threat to internal validity whereas a participant could have prior knowledge of the use case. Therefore, the use cases were anonymized in cooperation with the use case managers in order to be accurate but unrecognizable by the participants.

External validity concerns itself with the generalizability of the results. An obvious risk is that the evaluations were performed in the same team, where end users with similar backgrounds evaluated the framework. This could minimize the generalizability of the framework to other types of end users. Therefore, we attempt to mitigate this by requiring participants in the focus group and expert interviews to also evaluate the framework's usability, besides validating its contents. As these experts have different backgrounds, we make sure the framework is accessible to different types of end users.

Reliability refers to the reproducibility of this study if performed by others. This study employs a fully qualitative approach as it is better suited for exploratory research. The reliability of qualitative research in evaluating information systems could decrease due to the subjectivity of researchers' interpretations (Kaplan & Maxwell, 2005). To combat this threat to reliability, first of all, all materials relating to the empirical evaluations are found in Appendices E, F, and G for reproducibility. Moreover, we believe we have reliable findings as multiple types of literary resources were used and various empirical methods were employed to triangulate the data throughout this study. Lastly, the MDE (Whittle et al., 2017) and BPM (vom Brocke et al., 2016) frameworks provide theoretical substantiation to study low-code BPM.

4 Initial framework – Literature review

This chapter describes the first input for our framework. It also represents the first instance of phase three in our DSRM, the 'design and development' phase, leading to an initial framework version. This version was developed on the basis of a literature review and an exploratory interview. Before going into the results, we present the theoretical foundation underlying our framework and how the framework should be interpreted. Thereafter, we present the results. Lastly, we discuss how the results are incorporated into the initial framework version.

4.1 Framework structure

The aim of our framework is to present a complete and internally consistent set of factors that are vital to consider by organizations in an easy-to-use and operational manner. For the purpose of completeness and internal consistency, our framework is theoretically substantiated by two aforementioned models for MDE (Whittle et al., 2017) and BPM (vom Brocke et al., 2016). Apart from providing a theoretical foundation, it allows us to categorize the factors under these dimensions increasing the framework's ease of use.

The taxonomy of MDE tooling-related issues describes four themes (technical, internal organizational, external organizational, social) and the framework of contextual factors in BPM is drawn through four dimensions (goal, process, organization, environment) to consider. We combine these themes and dimensions together to form our framework's core structure. Two pairs, one from each model, have been merged due to overlapping concepts. The internal organizational factors (Whittle et al., 2017) and the organization-dimension (vom Brocke et al., 2016) both assess the structure, characteristics, and culture intra-organizationally. Moreover, both frameworks incorporate inter-organizational characteristics through the assessment of external (Whittle et al., 2017) or environmental (vom Brocke et al., 2016) factors. Therefore, we end up with six dimensions defining the core structure of our framework, visualized in Table 2.

Dimension	**Definition questions**
Technical	What technical (solution) characteristics could be a differentiator?
Organizational	What organizational (internal) characteristics could be a differentiator?
Environmental	What environmental (external) characteristics could be a differentiator?
Social	What social (engagement) characteristics could be a differentiator?
Process	What differences in business process characteristics can be a differentiator?
Goal	What differences in BPM project characteristics can be a differentiator?

Table 2. The core structure of the assessment framework.

All elicited characteristics of a BPM initiative that are important to consider when deciding on low-code BPM adoption fall under one of these dimensions. These characteristics are hereafter referred to as 'factors' throughout this study.

Each factor was found using thematic analysis, as described in Chapter 3.2. While reviewing the factors, in line with the methodology by Braun and Clarke (2006), we found that there are three distinct ways to interpret the factors: some factors are absolutely essential, some might be important depending on the organization's context, while other factors are, instead, incentives to adopt low-code BPM. In order to produce a complete framework, we decided not to limit ourselves to a certain perspective. Hence, we inductively analyzed our previously-elicited factors, established three 'factor types', and defined how these should be interpreted by the users of the framework:

- **Key factors**: These factors describe key BPM initiative characteristics that should be present for the success of a low-code BPM implementation in an organization. These factors can be objectively gauged to be present or not and a lack of them can, rarely, be mitigated before adoption. We do not define these as the widely-known critical success factors (Leidecker & Bruno, 1984) as we cannot fully assure successful performance. However, these factors highlight vital aspects which should carefully be considered by decision-makers.
- **Reflection factors**: These are important factors that organizations should be aware of, and carefully consider to be present in an organization's BPM initiative. If not, these could possibly be mitigated before adopting low-code BPM. Contrary to key factors, these factors are self-reflective in nature and highly subjective to the context of an organization. What the effect of such a factor is on the low-code BPM adoption decision is, also, highly dependent on each BPM initiative's context.
- **Beneficial factors**: These factors are proven to be especially supportive of low-code BPM. Organizations that have successfully adopted low-code are often cited to have these aspects. A lack of this factor should not drive decision-makers away. Contrary, if the BPM initiative falls under a beneficial factor, it can be seen as an additional incentive to choose low-code BPM.

As the reader may conclude, some factors can be subjective and are up to the interpretation of the user. In the end, these factors are meant to guide the decision-making process. By incorporating different dimensions of low-code BPM and visualizing them as categories, the conceptual framework aims to provide a complete picture in an easy-to-use manner. Using various factor types, the framework strives for internal consistency and operationality.

4.2 Results

Here, we present the results following our literature review and exploratory interview. Each following sub-chapter describes a dimension, the potential benefit of low-code BPM to that dimension, and its underlying factors to consider. Thereby, the type of factor is also mentioned. At the end of Chapter 4, we include an overview of all factors, with their title and definitions.

4.2.1 Process dimension

When digitalizing a business process, low-code BPM can support any business process model and tailor the BPMS to the needs of any organization (Koplowitz, 2017). However, the characteristics of such business process have been found to play a crucial role in BPM implementations (vom Brocke et al., 2016) and should, therefore, be considered. The following two factors describe beneficial process characteristics for low-code BPM.

In abstract terms, organizations consist of either core processes, which are generally standardized processes supporting internal operations, or strategic processes, which are processes defining the business model and differentiating the organization from its competition (McManus, 2003). In make-or-buy literature, the general consensus is that core processes are more effectively and efficiently supported by acquired packaged solutions while strategic systems should be developed in-house (McManus, 2003). In line herewith, it has been found that LCDPs are most effectively used for strategic processes to improve cost efficiency (OutSystems, 2019; Bratincevic, 2020). The customizability offered through low-code becomes cheaper than adapting packaged solutions (Bratincevic, 2020). Some LCDP implementation examples do support entire core processes, sometimes in combination with existing systems (OutSystems, 2019). However, the value and cost-effectiveness of these implementation is yet unknown as these solutions are based on long term results (Bratincevic, 2020). Therefore, a **strategic process** is a beneficial factor where low-code BPM is most effective.

On the other hand, a survey by OutSystems (2019) had shown that the most popular apps developed by LCDPs were specifically customer- or partner-oriented and this has two reasons. Firstly, the emphasis of BPMSs is shifting from internal, standardized systems towards customer serving applications where application and process requirements change continuously. This requires a modelling approach that allows constant change (Koplowitz, 2017), which low-code BPM allows. Secondly, customer satisfaction is key for value-adding processes (vom Brocke et al., 2016) and low-code BPM provides GUI designers to support the customer experience. Therefore, we elicit from literature that a **front-office process** is especially suitable for low-code BPM, thus a beneficial factor. To be clear, front-office processes are business processes that interact with stakeholders outside of the organization and, thus, have a high probability of being modified due to external effects, such as changing customer demands or other triggers.

4.2.2 Goal dimension

While the process dimension focused on the to-be digitalized business process, factors under the goal dimension reflect the overall aims and goals of the whole BPM initiative. Apart from the obvious goal of digitalizing a business process, low-code enables quick and flexible development in organizations (Luo et al, 2020; Sahay et al., 2020; Sanchis et al., 2020; Alsaadi et al., 2021; Cicman et al., 2021; Frank et al., 2021). However, to reap all the benefits, various factors should be considered.

As we recall, different types of LCDPs have different end-goals. The core utilization of the 'workflow management systems' group is to develop business applications, governed by a workflow engine (Frank et al., 2021). Moreover, low-code BPM allows to integrate various business applications in the business process into a single application where the process participant works and interacts with the integrated business process (KPMG, 2021). The goal to digitalize and integrate the whole end-to-end process is known as process orchestration (Dumas et al., 2018; Xu et al., 2018). We, therefore, define the goal of an **application for process orchestration** as a key factor. It may seem obvious to include this as a factor. However, an initiative solely focused on application development, data management, or a headless workflow automating system that could also be described as a BPM initiative, would be more suited to other types of LCDPs (Bock & Frank, 2021b).

As low-code is a rising technology, many organizations are trialing it for the first time. Previous MDE research has shown that promised benefits do not occur immediately after the first development project (Hutchinson et al., 2014). Organizations that worked on a project-by-project basis often disregarded the technology unjustly. Moreover, first implementations often incurred difficulties as organizations faced the technology for the first time. Instead, organizations should focus on the long-term, with a progressive and iterative project approach (Hutchinson et al., 2014). BPM best-practices also state how BPM initiatives should be a continuous initiative, not an ad-hoc project (vom Brocke & Schmiedel, 2014). Taking lessons from previous research, **iterative and long-term project orientation** is included as a key factor. This way, the development and maintenance of business applications through low-code can become cost- and time-efficient in the long-term (Olariu, Gogan & Rennung, 2016).

Previous research states that time and flexibility benefits come at a cost of limited size and scalability of applications developed using LCDPs (OutSystems, 2019; Bucchiarone et al., 2020; Sahay et al., 2020; Alsaadi et al., 2021). Also, the exploratory interview showed that the larger and more complex an application may become, the less citizen development can take place and the less time-efficient development becomes. This would mean that only experienced software developers could build applications, while being limited to the tools offered by the LCDP, possibly making LCDPs unsustainable in the long-term (Koplowitz, 2017). The example presented by Woo (2020) where an application was built in days during COVID-19 times highlight this trade-off. On the one hand, the organization employed low-code to be able to build and adapt their application to changing external effects. On the other hand, this applications focused on one specific process and was limited in size. Therefore, organizations should carefully consider the widely cited benefits such as short delivery time and high adaptability as these could be based on smaller applications than the BPM initiative intends. The **application size** is thus an important reflection factor. Larger applications can be developed through low-code BPM, but these will be project that should be conducted in a continuous and iterative manner.

4.2.3 Technical dimension

The previous dimension clarified the project aspects to consider when considering low-code BPM adoption. But technological aspects of low-code should also be studied. Generally, LCDPs benefit application development through low-code's simplicity, leading to higher maintainability of applications, and through reducing the effort of repetitive tasks in development (Alsaadi et al., 2021; Bock & Frank, 2021a). However, there are various technical considerations in a BPM initiative that could diminish these benefits.

First of all, LCDPs offer predefined development capabilities and tools. This implies that LCDP benefits can only be accomplished when all software development requirements are satisfied by the LCDP's framework (Bock & Frank, 2021a; Cai et al., 2022). Note that the 'Application size' factor assesses the requirements for the application to be developed, not the platform on which the application is developed. Here, the possibilities for software testing, deployment, version control and other software development practices are considered. Currently, LCDPs do not have open standards and it is not possible to add new functionalities, hampering the scalability of LCDPs (Sahay et al., 2020). Moreover, research in MDE had shown that platforms lacked debugging tools, no support for parallelization, concurrent execution and distribution, or an overall lack of performance (Bucchiarone et al., 2020). Therefore, organizations should analyze **LCDP's scalability and capability** according to their software development standards before adopting low-code BPM. This is a reflection factor as it highly depends on the development practices incorporated in the organization.

Secondly, a key issue faced by developers has been the compatibility of LCDPs with the organization's IT infrastructure (Koplowitz, 2017). On the one hand, low-code BPM offers process integration features through connections with external systems. This enables low-code BPM to act as a layer between the IT infrastructure and the user. On the other hand, large organizations have strict requirements for system integration which could be in conflict with low-code technology (Koplowitz, 2017). In general, BPMS compatibility with the surrounding IT infrastructure is an absolutely essential requirement (Bosilj Vukšić et al., 2018). Therefore, **compatibility with IT infrastructure** is a key factor that should be assessed for the current, and future, BPM initiatives. If full compatibility is not possible, another solution should be considered.

Thirdly, a widely discussed technical issue of low-code is its security. Various publications discuss how security of LCDPs has hampered its adoption (Olariu et al., 2016, Koplowitz, 2017; OutSystems, 2019; Bratincevic, 2020; Alsaadi et al., 2021; Hoogsteen & Borgman, 2022). On the one hand, Alsaadi et al. (2021) found that very few LCDPs have security certificates. Company information is stored on such platforms therefore running a risk of being compromised. This could, partially, be mitigated by choosing for either an on-premises or hybrid solution (Koplowitz, 2017). Apart from the technical aspect, software development performed by non-IT experts could pose security risks as IT governance

(discussed later in this chapter) is undermined (Olariu et al., 2016). Studies have found an increased chance of data security, compliance issues, and cyberattacks (Sanchis et al., 2020; Hoogsteen & Borgman, 2022). Therefore, assessing the **security according to business standards** is vital and a key factor when developing using low-code BPM.

4.2.4 Social dimension

The social dimension concerns itself with the employees and middle-managers responsible for implementing a low-code solution. The people actually developing applications using LCDPs are vital to consider, as shown in research into low-code (Alsaadi et al., 2021), MDE (Hutchinson et al., 2014), and BPMS (Trkman, 2010). One of the main benefits of LCDPs is the lower dependency on software development skills (OutSystems, 2019; Sanchis et al., 2020; Vincent et al., 2020; Frank et al., 2021). This allows employees to apply their creativity and innovativeness to develop useful applications, as mentioned in an expert interview. However, a couple of important social factors should be considered before adopting low-code BPM in an organization.

The promise of low-code's simplicity and citizen development potential needs to be nuanced. Although application development or process automation is becoming more accessible, it remains a complex process that requires certain technical skills (Bucchiarone et al., 2020; Frank et al., 2021; Hoogsteen & Borgman, 2022). Starting with LCDPs involves a steep learning curve for novices where a certain level of software development knowledge is still required (Sahay et al., 2020; Luo et al., 2021). For now, as low-code is a 'new technology', there is an overall lack of low-code skilled people in the community (OutSystems, 2019; Sanchis et al., 2020; Alsaadi et al., 2021). Following from the points above, we can imply that an organization should have technically savvy employees with, at least some, software development experience for low-code BPM. **Existing software development knowledge** is seen as a reflection factor as it could be mitigated through training and external partnerships (mentioned later in this chapter).

Besides low-code skills, studies have pointed out a lack of trust being a possible cause of resistance. On the one hand, distrust regarding technical capabilities of LCDPs, such as the type of systems it could develop (Sanchis et al., 2020), could result in employees reverting to traditional development (Aranda et al., 2012). MDE studies have shown that developers mistrusting the technological capabilities employed workarounds, hereby defeating the purpose of the technology (Hutchinson et al., 2014). Moreover, developers distrust low-code technologies due to aforementioned limitations in scalability, security risks, and integration problems (Alsaadi et al., 2021). The exploratory interview found that IT departments were adverse towards low-code development due to the burden of additional system maintenance and security work regarding applications developed by non-technical people. Adopting low-code development is then seen as only adding towards the workload. Thirdly, overall misconception of the technology has been considered previously as one of the main obstacles for MDE

adoption (Hutchinson et al., 2014). Also, in BPMS implementation studies, mistaken beliefs in capabilities caused fear of job loss, dissatisfaction and conflicts among employees (Bach, Bosilj Vukšić & Suša Vugec, 2017). Practical recommendations by Bach et al. (2017) state that educational programs should be included before beginning an implementation project to resolving this misunderstanding. Therefore, **employee trust** should be assured, and subsequently evolved, under employees before adopting any low-code technology in an organization. As this factor can be mitigated by an organization, we see this as a reflection factor.

Contrarily, an expert highlighted that the presence of young and eager-to-learn employees in an organization is beneficial for low-code BPM adoption. From their eagerness, such employees are more involved in experimenting with the technology. "*Low-code stimulates innovative potential in employees*", stated the expert, as users could create an application based on their own ideas. Moreover, younger generations have more affinity and experience with technology, and an urge to make their work more efficient through technology. On the other hand, employees that lack such characteristics are reluctant to change. This has been seen with the adoption of MDE (Aranda et al., 2012) but also with the first adoptions of low-code technology (OutSystems, 2019; Sanchis et al., 2020). Institutional theory can explain this hesitance as people have repetitive and habit-forming tendencies. These cause workers to prefer the known approach instead of switching to the unknown (Aranda et al., 2012). Therefore, we see **innovation-driven culture** as a beneficial factor.

4.2.5 Organizational dimension

The organizational dimension differentiates itself by focusing on the organization as a structured business, with a management leading it and deciding its strategy. It might seem to overlap with the social and goal dimensions. However, the social dimension focuses on the developers and middle managers implementing low-code instead of the higher management. The goal dimension looks at explicit project goals and characteristics instead of organizational structures and company-wide strategic decisions. For an organization, low-code BPM primarily offers increased responsiveness to changing digital requirements and market conditions, strengthening a company's competitiveness (Olariu et al., 2016; Sanchis et al., 2020; Frank et al., 2021). Still, the organization should analyze various factors before committing to low-code for their BPM initiative.

The most prominent success factor cited in scientific literature is active top management support. It refers to the involvement by top managers through guiding processes, providing resources and steering the organization (Trkman, 2010; Hoogsteen & Borgman, 2022). Especially in the case of technology-adoption, top management support is vital (Hsu, Liu, Tsou & Chen, 2018). Active top management support includes, for example, communication of opportunities or proactive guidance through technology trainings, opposed to pure decision-making with passive support (Hoogsteen & Borgman, 2022). Active support was crucial with citizen development practices as top management could free

needed resources for technology implementation (Hoogsteen & Borgman, 2022). Human and financial resources are not only vital for the implementation of innovative technologies, but also in any BPM initiative (vom Brocke et al., 2016). Altogether, the support from top management is key to kickstarting a BPM initiative through the first courses of action (Bach et al., 2017). Therefore, **active top management support** is a key factor in the organizational dimension.

Apart from the involvement and support of top managers, the technological initiative should be supported by business goals. This connection between IT and business strategy is often called the 'business-IT alignment' and is seen as a lasting obstacle for any implementation of a new technology in an organization (Luftman & Brier, 1999). Various studies in relatable fields to low-code development have found business-IT alignment as an essential factor. Trkman (2010) found that, apart from top management involvement, the link between the BPM initiative and the organizational strategy was a critical success factor. The BPM initiative should add to strategic value creation (vom Brocke & Schmiedel, 2014). Moreover, Whittle et al. (2017) state that adopting MDE technology should also be seen as a business decision and the IT and business goals are to be aligned. Lastly, citizen development generally originates from business and the IT department must support the business through guidance and supervision (Bratincevic, 2020; Hoogsteen & Borgman, 2022). When business and IT is too separated and there is a lack of communication and cooperation, resistance in the organization will be insurmountable, the exploratory interview revealed. Therefore, **business-IT alignment** should be present in an organization and is seen as a key factor.

Closely related to business-IT alignment is the necessity of IT governance. IT governance entails the guidance and structures in place to sustain the IT infrastructure in enabling an organization's strategy and objectives (de Haes & van Grembergen, 2009). As LCDPs lower the barriers for employees to develop IT, there is a need for strict guidance (Hoogsteen & Borgman, 2022). With it, previously mentioned security threats and breaches of regulatory requirements may occur (Olariu et al., 2016; OutSystems, 2019). Moreover, besides IT development, BPM initiatives also require proper governance (vom Brocke & Schmiedel, 2014). Hoogsteen and Borgman (2022) argue that a centralized IT governance is the most suitable form as it organizes the governance from one point in the organization, instead of multiple decentralized governance structures (Brown, 1997). This enhances clarity and control when a myriad of workers engages with low-code BPM (Hoogsteen & Borgman, 2022). The exploratory interview also pointed out the requirement of having a centralized IT governance structure before adopting low-code technology. Although it is possible to develop governance while implementing low-code, setting up governance is costly (Olariu et al., 2016). Therefore, we see a **centralized IT governance** structure as a key factor that should be present if low-code BPM were to be implemented.

As previously stated, one of the main reasons for adopting low-code has been to lower the dependability on highly skilled developers in an organization. However, this does not come at a cost. When an organization already has an established software development process, introducing low-code development will inevitably cause great changes (Hutchinson et al., 2014; Frank et al., 2021). Hutchinson et al. (2014) describe an MDE implementation case where an organization did not integrate the new approach and, in the end, the MDE integration only caused harm and inevitable failure. Further studies into MDE implementations have shown that organizations should account for team composition, roles, and job description changes after implementing MDE tools (Aranda et al., 2012). Such changes in an organizational structure could cause the balance in companies to change and subsequently also affect the organizational culture (Bach et al., 2017). Therefore, a company should carefully reflect and prepare for the effect that **integrating a new software development approach**, namely the low-code development approach, would pose. We, thus, define this as a reflection factor.

4.2.6 Environmental dimension

The organizational dimension describes factors regarding internal matters and how relationships are within an organization. The environmental dimension, on the other hand, refers to the relationships between an organization adopting low-code and its external environment. Two types of external relationships are found to be beneficial for low-code BPM.

First of all, LCDPs have been implemented by organizations in many different industries. Although above average utilization is reported in the utility sector and below average in the public sector (OutSystems, 2019), it is impossible to draw an exact distinction between sectors supportive and unsupportive of low-code. Studies do show that LCDPs could be especially supporting for organizations in high-paced and changing industries (OutSystems, 2019). This could be due to new innovations, transforming the industry, or due to sensitivity to regulatory changes. BPMSs enable change and risk management, strengthen analytical capabilities and allow open innovation which is more suitable for such environments (vom Brocke et al., 2016; Bosilj Vukšić et al., 2018). The previous conceptualization of low-code BPM emphasizes exactly these qualities. The same applies to MDE tools, as they have also been seen used in changing and unpredictable environments (Whittle et al., 2017) and LCDPs, by design, encourage business responsiveness to changing demands (Alsaadi et al., 2021). As long as processes remain structured, and do not conflict with the previously mentioned process-factors, low-code BPM is suitable for **high-paced and changing environments** and, therefore, we see this as a beneficial factor.

Apart from the relationship between an organization and its industry competitors, an organization also has relationships with business partners. These interorganizational partnerships can be vital for the success of low-code adoption. Hoogsteen & Borgman (2022) highlight how organizations may struggle to implement citizen development practices without vendor support. Additionally, external parties

specializing in low-code development could contribute to an easier implementation. MDE literature had shown that a development 'middle man' can be needed for a successful adoption (Aranda et al., 2012). Such partners can provide training sessions, knowledgeable implementation managers and assist with problem-solving. BPMS literature similarly highlights the partner's reputation, knowledge, and experience in serving clients as an important contributing factor (Bosilj Vukšić et al., 2018). Therefore, if an organization already has partnerships with LCDP implementation specialists, this could be a contributing factor in adopting technology. Therefore, **existing external implementation partners** are seen as a beneficial factor aiding low-code BPM adoption.

4.3 Towards the initial framework

All the factors described above are consequently integrated into an initial conceptual assessment framework, presented in Figure 3. We follow the guidelines by Miles, Huberman, and Saldaña (2014) for constructing conceptual frameworks as a framework should both provide an integrated view to the research problem and represent the concepts in a logical and graphical manner.

The six dimensions and their underlying factors are separated by colored- and dashed-lines respectively. Moreover, the different types of factors are color-coded as shown in the legend. Three other features are integrated in the assessment framework to aid users in assessing their own BPM initiative.

Firstly, the dimensions that are likely to 'interact' more frequently with each other are positioned in closer proximity. This allows different end users to easier assess the side of the framework in relation to their own background. For example, an organization's environment (*environmental dimension*) naturally has an influence on the way the organization is governed. The top management (*organizational dimension*) determining this, further, decide on the BPM project's goals (*goal dimension*). Moreover, the top management steers the middle managers and workers (*social dimension*) performing the BPM initiative. The latter then actually develop using low-code BPM (*technical dimension*). The business process (*process dimension*), which interacts with its environment, is then automated using low-code BPM, through the prescribed project goals, by the company's employees or external partnerships.

Secondly, the order of the underlying factors is not at random with the same usefulness in mind. This can be seen in how the 'Employees with low-code affinity' factor, describing the technical savviness of the team, is close to the technical dimension. These two are likely to be assessed by someone with a technical background and it is, thus, sensible to put these close to each other. Moreover, a person assessing the 'Business-IT alignment' will likely also be able to assess the 'Iterative and long-term project orientation' of the organization.

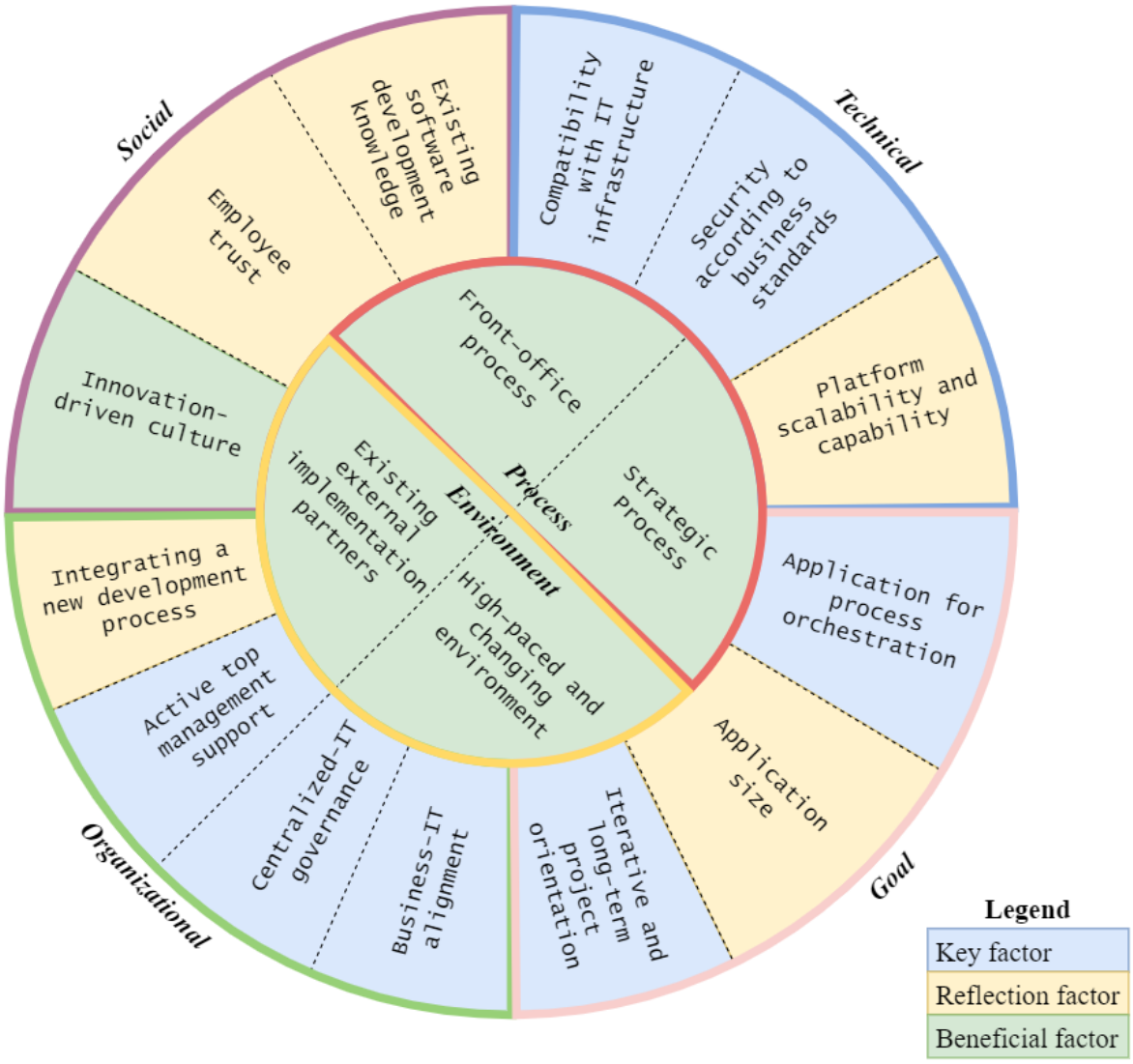


Figure 3. The initial conceptual assessment framework

In the end, the conceptual framework is accompanied by a summary table providing definitions to all factors. We follow a similar template to Plattfaut et al. (2022), who conducted a study on RPA's success factors, resulting in Table 3.

Dimension	Factor	Description	Applicability to low-code BPM	Explanatory quote	Sources
Process	Strategic process [Beneficial]	Whether a process defines an organization and can be seen as of strategic value	Due to the capabilities of low-code BPM, strategic processes are especially well-suited for low-code support	"*(…) strategic, mission-critical systems should be developed in-house whenever possible*" (McManus, 2003)	McManus (2003); OutSystems (2019); Bratincevic and Rymer (2020)
	Front-office process [Beneficial]	Whether a process has a high probability of being modified by external effects, such as changing customer demands, or other external triggers	Due to the capabilities of low-code BPM, regularly changing processes are especially well-suited for low-code support	"*(…) in a new world focused on serving customers, processes change rapidly, and speed to market is king*" (Koplowitz, 2017)	vom Brocke et al. (2016); Koplowitz (2017); OutSystems (2019)
Goal	Application for process orchestration [Key]	Whether the BPM initiative's goal is to have an application integrating an end-to-end process as a one-stop shop for the user	If the BPM initiative does not fall under this definition, likely another solution would be more suitable	"*(…) [LCDPs] integrate, in one environment, multiple well-known and traditional system design components so as to reduce the efforts of routine tasks in implementing business applications (…)*" (Bock & Frank, 2021a)	Koplowitz (2017); Sahay et al. (2020); Bock and Frank (2021b); Frank et al. (2021)
	Iterative and long-term project orientation [Key]	Whether this BPM initiative is part of a long-term commitment to low-code BPM and consists of multiple project iterations	If the BPM initiative does not fall under this definition, likely another solution would be more suitable	"*Organizations which structure their work on a strict project-by-project basis may find it much more difficult to successfully trial MDE*" (Hutchinson et al., 2014)	Hutchinson et al. (2014); vom Brocke and Schmiedel (2014); Olariu et al. (2016)
	Application size [Reflection]	An organization needs to assess whether the loss in application scalability is less than the benefit of rapid and flexible development	If the outcome is that an organization needs an application with high complexity, and other existing solutions support this need, the other solution might be a better option	"*(…) [LCDPs] can promote software development productivity if all the requirements of a given project can be satisfied within the predefined, immutable framework of a certain LCP (…)*" (Bock & Frank, 2021a)	Koplowitz (2017); OutSystems (2019); Bucchiarone et al. (2020); Sahay et al. (2020); Alsaadi et al. (2021); Bock & Frank (2021a); Cicman et al. (2021)
Technical	LCDP's scalability and capability [Reflection]	Whether building applications through a certain LCDP, with its own limitations on scalability and capability, is workable according to organization standards	If low-code BPM does not suffice to the organizational standards, likely another solution would be more suitable	"*(…) lack of debugging methods and tools; lack of performance, scalability, and inability to deal even with mid-sized models; little or no support for parallelization, concurrent execution or distribution; and poor interoperability*" (Bucchiarone et al., 2020)	Bucchiarone et al. (2020); Sahay et al. (2020); Vincent et al. (2020); Bock & Frank (2021a); Cai et al. (2022); Frank et al. (2021)
	Compatibility with IT	Whether the LCDP solution offers satisfactory	If low-code BPM does not offer the necessary	"*Problems integrating the apps developed by LCDP*	Koplowitz (2017); Bosilj Vukšić et al.

	infrastructure [Key]	compatibility features in an organization's IT infrastructure	compatibility features, likely another solution would be more suitable	*with other systems*" (Alsaadi et al., 2021)	(2018); Alsaadi et al. (2021)
	Security according to business standards [Key]	Whether the LCDP solution offers satisfactory security standards according to an organization's IT governance	If low-code BPM does not offer the necessary security features, likely another solution would be more suitable	"*Research shows that software development outside of the IT domain, such as citizen development, is a real threat to data security and compliance issues and means that businesses are more likely to encounter cyberattacks*." (Hoogsteen & Borgman, 2022)	Olariu et al. (2016), Koplowitz (2017); OutSystems (2019); Sanchis et al. (2020); Alsaadi et al. (2021); Saghafian et al. (2021); Hoogsteen and Borgman (2022)
Social	Existing software development knowledge [Reflection]	Whether the employees in an organization have low-code skills, or at least software development knowledge to develop a low-code BPM application	Organizations should assess whether this applies to the organization or take applicable measures	"*(…) building business applications with LCDPs is still a complex process that demands specific abilities, which in turn, hamper organizational adoption decisions*" (Hoogsteen & Borgman, 2022)	OutSystems (2019); Bucchiarone et al. (2020); Sanchis et al. (2020); Alsaadi et al. (2021); Frank et al. (2021); Hoogsteen & Borgman (2022)
	Employee trust [Reflection]	Whether developers or middle managers have trust in the potential, capabilities and future of low-code BPM before the adoption-decision is made	Organizations should assess whether this applies to the organization or take applicable measures	"*(…) fear of job loss because of redesigned processes, dissatisfaction with the logic of decision-making, conflicts and lack of communication and interaction within the new working environment*." (Bach et al., 2017)	Aranda et al. (2012); Hutchinson et al. (2014); Bach et al. (2017); Sanchis et al. (2020); Alsaadi et al. (2021)
	Innovation-driven culture [Beneficial]	Whether an organization motivates innovation and consists of eager-to-learn employees that want to learn and develop low-code BPM	If this applies to an organization, it provides a fruitful ground for low-code BPM development	"*If you have a young organization with enthusiastic employees, they will enjoy experimenting with it [low-code technology]. But, if you have employees working for many years in IT with traditional systems, they will not be looking forward to working with a new technology.*" (Exploratory interview)	Exploratory interview; Aranda et al. (2012); OutSystems (2019); Sanchis et al. (2020)
Organizational	Active top management support [Key]	Whether top management will be actively involved, will provide the necessary resources, and will steer the organization through LCDP adoption	This should be gauged and agreed prior to decision-making as it is vital to the success of low-code BPM adoption	"*Top management support (TMS) appears to be especially important for technology-related adoption decisions and refers to management guidance on operating processes by providing resources and explicit directions for managing the organization*"	Trkman (2010); vom Brocke et al. (2016); Bach et al. (2017); Hsu et al. (2019); Hoogsteen and Borgman (2022)

				(Hoogsteen & Borgman, 2022)	
	Business-IT alignment [Key]	Whether there is a good business-IT alignment when considering low-code BPM	Low prior business-IT alignment can be detrimental to making the correct decision, and subsequent implementation of the technology	"*This is highlighted by the LCDP experts, with all three stating the importance of a strategic fit between business and IT objectives for citizen development adoption*" (Hoogsteen & Borgman, 2022)	Luftman and Brier (1999); Trkman (2010); vom Brocke & Schmiedel (2014); Whittle et al. (2017); Hoogsteen and Borgman (2022)
	Centralized IT governance [Key]	Whether a centralized IT governance structure is present and can be modified to support low-code development in an organization	If such structure is not in place, and cannot cost-effectively be developed, the benefits of low-code BPM might not be reached	"*Citizen development is most suitable for a centralized IT Governance structure (…) as centralized decision-making will increase clarity, coordination, and control for citizen development, which are all important aspects since they can mitigate related risks*" (Hoogsteen & Borgman, 2022)	Brown (1997); de Haes & van Grembergen (2009); vom Brocke & Schmiedel (2014); Olariu et al. (2016); OutSystems (2019); Hoogsteen & Borgman (2022)
	Integrating a new development process [Reflection]	The organization should assess whether integrating a new (low-code) development process, instead of the existing way of working, is viable	If the BPM initiative does not fall under this definition, likely another solution would be more suitable	"*(…) the results of using MDE was, after all, the point of changing a software development process*." (Hutchinson et al., 2014)	Aranda et al. (2012); Hutchinson et al. (2014); Bach et al. (2017); Frank et al. (2021)
Environmental	High-paced and changing environment [Beneficial]	Whether an organization operates in a highly variable industry which is prone to technological or regulatory changes	Organizations in such industries are more in need of responsiveness in changing requirements posed with low-code technology	*"[MDE tools] are used in various changing and sometimes unpredictable contingencies*." (Whittle et al., 2017)	vom Brocke et al. (2016); Whittle et al. (2017); Bosilj Vukšić et al. (2018); OutSystems (2019); Alsaadi et al. (2021)
	Existing external implementation partners [Beneficial]	Whether an organization already has existing partnerships with LCDP vendors or other low-code specialists	If this is true, then an organization is especially suited for LCDP adoption due to the easy-access to knowledge	"*Previous research on cloud adoption confirms the importance of these inter-organizational business networks, owing to many cloud service providers arranging change agents, providing training sessions, and addressing potential problems*." (Hoogsteen & Borgman, 2022)	Aranda et al. (2012); Bosilj Vukšić et al. (2018); Hoogsteen and Borgman (2022)

Table 3. The underlying factors, with their definitions, relation towards low-code BPM, explanatory quote and sources.

5 Refined framework – Focus group

Following the results from the literature review, the focus group represents the first empirical validation of this framework. This was also the second instance of the 'design and development' phase in our DSRM, presented in Figure 1. We use various methods in order to triangulate the findings and incorporate different views. The main goal was to validate the initial framework resulting from literature, while also eliciting framework improvements from the participant's knowledge and experience. Additionally, we wanted to gather feedback on the framework's ease of use and operationality as experienced by the participants. This chapter reports the results of the focus group and how these results were interpreted into a new, refined framework version.

5.1 Focus group overview

A two-hour focus group was performed with three participants. The participants were carefully chosen to have various backgrounds in development and implementation management related to low-code. A summary of participants is provided in Table 4. In the first part of the session, the participants introduced themselves, the context of the study and our conceptualization of low-code BPM was presented, and the participants were given time to familiarize themselves with the framework. The second part of the session included the focus group itself where the focus group had to answer this study's research question. The complete focus group protocol and materials used can be found in Appendix B and F.

#	Current Role	Experience with Low-Code and BPM
FG1	Manager in Digital Transformation and Intelligent Automation	Leading and consulting on multiple low-code projects for application development and workflow automation using various low-code development platforms
FG2	Senior consultant in Digital Sourcing and Procurement	Developing and leading the technical implementation for multiple low-code development projects for application development using various low-code development platforms
FG3	Senior manager in Operational Excellence and Digital Transformation	Leading multiple projects on process excellence initiatives including overseeing multiple low-code BPM implementations

Table 4. Overview of focus group participants

5.2 Results

Each subsequent sub-chapter presents the findings from the focus group per dimensions. Tables 5 to 10 summarize these findings. The last sub-chapter describes the participants' comments on the design and usability of the framework.

5.2.1 Goal dimension

The goal dimension concerns itself with what the BPM initiative tends to achieve. The essence of understanding the BPM initiative's goal was underlined by FG3: "*The first question actually has to be: what problem does the client want to solve, or what problem is the client-facing that they are*

considering low-code?". One of these problems is represented in the current factor: ***application for process orchestration***. All participants agreed that process orchestration is one of the main goals achievable through low-code BPM. Prior to adopting low-code BPM, FG1 recalled organizations mentioning: "*insufficient insights into our process, and the process output was suboptimal*". With process orchestration, you see that "*errors that people originally made can be prevented as the steps are digitalized […] employees experience a well design low-code BPM process to work much more pleasantly and efficiently*" (FG3). We argue that the current factor is, hence, validated but its original definition focuses on "one application" supporting the whole end-to-end process. FG1 and FG3 argued that does not have to be the case: "*…not necessarily. So you can have a process where not the whole end-to-end process is included, where pieces still are supported by packaged solutions*". Therefore, we slightly adjust the factor and its description to lay the emphasis on process orchestration, and its benefits, as a goal.

The ***iterative and long-term project orientation*** factor was derived from literature to highlight that low-code's cost- and time-efficiency benefits mostly arise in the long-term. One participant misinterpreted this factor as if an application developed through low-code had to remain the same for a long time. FG3 countered this: "*we say the world changes quickly and what is required from an application changes quickly. The idea of low-code is to tackle this issue*". After explaining the rationale behind the factor, FG3 coincided with the factor's original essence: "*whether we see low-code as a long-term solution, that is a clear yes*". To avoid any misunderstanding, we adjust the naming of the factor to resemble long-term orientation to low-code as a technology.

In the previous quote, FG3 mentions that low-code is especially suitable when an application needs to change quickly. This was underpinned by FG1 as "*[low-code is suitable] if you need more flexibility. […] The benefit is that you can develop quicker than, let's say, traditional development. Because you can develop more quickly, your go-to-market is shorter so you can get more value from your applications*". We decided to introduce speed and agility as a factor in this framework. Existing literature supports this as LCDPs' ability to develop, adapt, and deploy software quickly (Frank et al., 2021). Furthermore, it is seen as the response to fast digital solution development (Sanchis et al., 2020) through its quick adaptability (Olariu et al., 2016). Therefore we introduce a new key factor named ***Requiring speed and agility***.

The previous framework included the ***application size*** as a reflection factor. Literature suggests that application size and complexity are limited when talking about the benefits of time, flexibility, and citizen development. This tradeoff itself was not doubted in the focus group. However, the more important aspect to consider is whether the LCDP, on which the BPMS is developed, is capable of handling the application size, instead of doubting low-code ability in general. The focus group mentioned that "*You can still create very complex low-code applications where it was still a good idea*

to do it [with low-code]" (FG2) and "*if you use your low-code development platform in the cloud, the scalability will rise automatically*" (FG1). FG3 underlined this with an example: "*in the United States, there are various large companies that use it [low-code] for large processes with a lot of throughput*". In the end, FG2 explained from his development experience: "*the conversation should be more about: what you are trying to achieve, does it fit on the platform?*". Therefore, we decided to remove application size as a factor but incorporate it in the 'LCDP capability' factor described in the following sub-chapter.

Original			*New*	
Factor	**Definition**		**Factor**	**Definition**
Application for Process Orchestration	Whether the BPM initiative's goal is to have an application integrating an end-to-end process as a one-stop shop for the user		Striving for Process Orchestration	The BPM initiative's goal is to streamline an end-to-end process through intelligent automation in an application
Iterative and Long-Term Project Orientation	Whether this BPM initiative is part of a long-term commitment to LCDP for BPM and consists of multiple project iterations		Iterative and Long-Term Low-Code Commitment	This BPM initiative is part of a long-term commitment to LCDP for BPM and consists of multiple project iterations
			Requiring Speed and Agility	The BPM initiative's goal is to enhance the responsiveness of development enabling higher speed and agility to changes
Application Size	An organization needs to assess whether the loss in application scalability is less than the benefit of rapid and flexible development			

Table 5. Goal dimension factor changes

5.2.2 Process dimension

Some processes may be more suitable for low-code development than others and the literature review found two suitable types. Having a ***strategic process*** was seen as a beneficial factor as low-code BPM's customizability could be used for the strategic processes as opposed to the core processes in an organization. However, FG1 mentioned "*I do not think it necessarily has to be only strategic processes where you can deploy low-code. It goes beyond your strategy-defining processes. […] What we see is that [low-code is used when] the market cannot fulfill all the [process] requirements*". The focus group agreed that the emphasis should lie on the contrast between unique and standard processes, as for the latter "*you buy standard applications, for your HR or CRM processes, which is much cheaper*" (FG1). The essence of the original factor was similar but should be clarified. FG3 proposed an ideal solution: "*It is a bit of a definition question: do we have a strategic process? Well, at least they are unique processes*".

Secondly, ***front-office processes*** were found to be beneficial for low-code BPM as process requirements may change through varying customer demands, which low-code BPM offers capabilities for. However,

as the factor is now stated, FG3 did not agree: "*No, it is certainly still used a lot for internal [back-office] processes. I think that is still the main thing that it's used for*". The key point from the original factor is that "*If a lot of flexibility is needed in the process or there can be a lot of changes in process requirements, then you can swiftly use low-code*" (FG1). The overall assumption that these processes are often in the foreground instead of the background is correct, but the essence lies in the variability of process requirements. Therefore, we rename and redefine the factor to accentuate the variability.

Another suitable process type was highlighted by FG3, who stated: "*If those [processes] are longer chain processes that go across multiple departments, that is often a good indicator for high [low-code] potential. Because those kinds of processes are tough to manage*". Moreover, "*a process built in Low-Code BPM is already much more controlled […] because you know where processes fall off the treadmill in a large organization*" (FG3). These arguments correlate with the previously mentioned 'Striving for process orchestration' factor, thus low-code BPM seems to be useful for ***complex processes*** that span over multiple actors and (data) systems.

Although the original strategic- and front-office process factors were considered beneficial factors, we now characterize all process dimension factors as key. From the initial literature research, we presumed that all processes are suitable for low-code BPM, with some types being more suitable than others. However, the focus group accentuated that implementing low-code BPM is only worthwhile in particular use cases. Therefore, presuming that all processes are suitable would be incorrect. Moreover, all examples of business processes given in the focus group fell specifically under these three factors. Therefore, we have chosen to put all process dimension factors down as key factors.

Original			New	
Factor	**Definition**		**Factor**	**Definition**
Strategic Process	Whether a process defines an organization and can be seen as of strategic value		Unique Process	A process is specific, or even unique, to an organization with high levels of customization
Front-office Process	Whether a process has a high probability of being modified by external effects, such as changing customer demands, or other external triggers		Varying Process	A process is sensitive to future changes
			Complex Process	A process has a level of complexity where multiple departments, systems, or data streams have to be incorporated into one application

Table 6. Process dimension factor changes

5.2.3 Technical dimension

When deciding what IT can support a BPM initiative, it is imperative to consider and understand the technical aspects of potential solutions. In the case of low-code development, organizations are dependent on low-code vendors, and therefore assessing the ***LCDP's scalability and capability*** was

found to be important based on literature. Literature cited a lack of open standards and development tools, no possibility to add functionality, and an overall lack of performance to require a careful assessment. From earlier comments on 'application size', however, participants showed that with the right fit between the platform and the application requirements, LCDPs are capable of creating large, complex applications. Moreover, "*the development steps and actions may well also be simply replicated through a low-code platform*" (FG2). Therefore, the factor is adjusted to emphasize a careful assessment of LCDP's capability and whether it can achieve the required solution, instead of focusing on development standards.

The LCDP's ***compatibility with IT infrastructure*** could also be an essential requirement due to its interconnected nature and the organization's strict system requirements. Only FG1 briefly mentioned: "*This is becoming less relevant for low-code. It has a growing amount of connections with standard systems to load data in, which is becoming easier*". This correlates with earlier findings from literature where low-code BPM is designed to act as a layer between the IT infrastructure and the user. However, no other participants responded or agreed with this statement. The definition of this factor remains unchanged in the end but will now be considered a reflection factor. Future validation is needed to understand the influence of compatibility with IT infrastructure.

Thirdly, the ***security according to business standards*** of LCDPs was seen as a key consideration and a potential problem, hampering adoption according to literature. However, all focus group experts stated, concerning safety: "*[Safety] is usually not a consideration for not doing it … whereas sometimes it actually is a reason for [adopting low-code]*" (FG1) and "*[safety] is actually kind of an advantage of using the [low-code] platform, as you are already developing out-of-the-box and with care*" (FG2), the latter being acknowledged by FG3. Instead, the participants agree: "*The risk that things go wrong due to human error or people not adhering to the process may indeed be greater*" (FG3). However, the latter is not a problem of low-code technology itself, but of how it is governed in the organization. Therefore, we remove this factor but do emphasize the human error risk in the 'Centralized IT-governance' factor, discussed in a later sub-chapter.

The factors above, and existing literature, have focused primarily on the technological characteristics of low-code development. On the other hand, the participants provided valuable insights into the technological use of low-code in the IT landscape of organizations.

Firstly, applications developed through low-code BPM appear to act as a layer that unifies various existing IT applications in the organization's environment. As an example, FG3 explained: "*What [employees] previously had to do in 5 systems, they now only use one. Therefore, you do not need to switch and click between different systems*". In the overall picture of process orchestration, "*a suboptimal process output often has something to do with the fact that they have a wide variety of legacy systems that do not work well together, where many mistakes are made or where employees are not*

comfortable working, or which are also poorly connected to external sources" (FG3). OutSystems (2019) emphasizes this integration of existing systems as a core capability of an LCDP in their report. Moreover, Sahay et al. (2020) mention the connectivity to external services as a main component of LCDPs and a step in the low-code development process. We, therefore, introduce ***unifying systems*** as a key factor that underlines low-code BPM's capability to serve as a unifying layer.

Secondly, FG2 mentioned: "*you also often use it as sort of a custom layer on top of large existing packages: SAP, that you'd rather not customize in because that's complicated or expensive*". From the focus group, it became apparent that due to low-code's speed and agility, a custom layer can quickly be built on top of existing software. In a business process, for example, you add "*a little extra process digitization if that part is not possible in a packaged solution*" (FG2). FG3 provided an example: "*If they are in the process of changing their ERP, or they want to make an ERP viable a little longer, you might find that to be quite a good fit with a low-code BPM layer that you then have around your ERP*". OutSystem's (2019) report corresponds with this finding as the use of low-code to extend existing systems is growing. We, therefore, see the need to add a key factor that defines this use of ***extending systems*** through low-code.

Original			*New*	
Factor	**Definition**		**Factor**	**Definition**
LCDP's Scalability and Capability	Whether building applications through a certain LCDP, with its own limitations on scalability and capability, is workable according to organization standards		LCDP's Capability	Whether the aspired application and development can be achieved using a specific LCDP
Compatibility with IT Infrastructure	Whether the LCDP solution offers satisfactory compatibility features in an organization's IT infrastructure		[remains unchanged]	[remains unchanged]
Security According to Business Standards	Whether the LCDP solution offers satisfactory security standards according to an organization's IT governance			
			Unifying Systems	Multiple existing (legacy) systems have to be connected together to allow for process orchestration
			Extending Systems	Existing (legacy) systems' capabilities have to be expanded to support a business process

Table 7. Technical dimension factor changes

5.2.4 Organizational dimension

Some studies argue that the success of model-driven tools is more dependent on social and managerial aspects than technical ones (Whittle et al., 2017; Cabot, 2020). Therefore, the organizational dimension touches upon the organization as a whole and its management. Firstly, as adopting low-code is often

seen as a business decision (Whittle et al., 2017), there should be a good ***business-IT alignment*** between the teams and their goals. Firstly, the participants agreed that it is important but "*if the business and IT can talk healthily to each other, then by definition you have better applications than if they do not*" (FG2). The ubiquity of business-IT alignment was further pointed out by FG3: "*What I hear you saying [FG2] about that Business and IT point. Obviously nice, but yes, is it necessarily decisive [factor]? I don't think it has to be*". In the end, the group agreed that in the case of low-code BPM, a good alignment "*[makes] it easier in a low-code development process to check requirements with the business a little more regularly, so to speak. And that it is also easier for the business to change something on short notice [...] I think you get changes through faster in low-code than if you're on a kind of classical change process*" (FG3). We, thus, keep this factor but in the following version of the framework, the factor definition emphasizes the cooperation between parties.

Literature vouched for a ***centralized IT governance*** when adopting low-code. The aforementioned threat that users, possibly unknowingly, may cause security risks should be mitigated with well-defined governance. Initially, FG1 and FG2 argued respectively "*Is [a centralized IT governance] important? I thought not*" and "*Does it help? Yes. Is it necessary? Doesn't have to, especially with low-code*". The participants argued that a decentralized structure is preferred. However, after a small discussion with FG3, the group stated that "*You always have to lay down the framework and make sure that everyone uses the same standards, the same guidelines*" (FG1). In the end, you "*always begin centralized […] but from there you can evolve to a kind of hybrid or even a decentralized model where you put it more and more in the teams themselves*" (FG3). As our framework is utilized prior to low-code adoption, we keep the 'centralized'-aspect in our factor but adjust the definition as the governance can be set up while adopting, instead of being present prior to adopting low-code.

Thirdly, the current version of the framework includes ***active top management support*** as a factor. Literature highlighted aspects such as communication of opportunities, proactive guidance, and freeing resources as important elements for successful adoption. The focus group did not mention or discuss this factor during the session, leaving this factor unchanged for the following version.

Lastly, we included ***integrating a new software development approach*** as an organizational factor. MDE literature showed that there were significant differences between traditional- and model-driven development approaches, and organizations that failed to fully adopt the new approach were unsuccessful with MDE. The focus group, however, disagreed. FG3 argued that "*I don't think that [changing the development approach] is necessary at all […] If they start developing with low-code, they can start developing along the same lines*" and FG2 argued that an Agile, or non-Agile, way of working is also suitable for low-code. FG3 provided a real-life example: "*[Organization] utilizes Appian a lot which is fully integrated into their finance processes. They just use that because they want to make a lot of specific custom changes in there. And they're already developing quite a lot themselves,*

too! So, they already have a kind of way of working because they already develop their own software. That combination makes it quite a good match". From these arguments, we gather that the method and approach of developing software do not have to change. Low-code is, of course, different than coding under traditional development, but this is a factor for the social level concerning IT employees, and not for the organizational level. Therefore, we remove the factor here but implement elements into the 'Employee trust' factor described in the next sub-chapter.

All three organizational factors are considered reflection factors in the following version of the framework, as opposed to key factors in the first version. In general, the focus group participants were unsure whether these factors need to be present prior to the adoption of low-code BPM in an organization. Moreover, these factors are often solvable through educational programs or other initiatives. Therefore, we decide to define these as reflection factors.

Original			*New*	
Factor	**Definition**		**Factor**	**Definition**
Business-IT Alignment	Whether there is a good business-IT alignment while considering LCDPs for BPM		Business-IT Alignment	Whether there is a good business-IT alignment between the process owners and the low-code developers
Centralized IT Governance	Whether a centralized IT governance structure is present and can be modified to support low-code development in an organization		Centralized IT Governance	Whether a centralized IT governance structure is present, or can be set-up, to support low-code development in an organization
Active Top Management Support	Whether top management will be actively involved, will provide the necessary resources, and will steer the organization through LCDP adoption		[remains unchanged]	[remains unchanged]
Integrating a new Software Development Approach	The organization should assess whether integrating a new (low-code) development process, instead of the existing way of working, is viable			

Table 8. Organizational dimension factor changes

5.2.5 Social dimension

The social dimension relates to the employees responsible for implementing, integrating, and using low-code technology in the organization. Based on literature, it was argued that having ***existing software development knowledge***, instead of low-code-specific knowledge was important in organizations. FG1, however, disagreed: "*Of course, you need the knowledge. So you need developers that know how to develop on the platform that you have chosen*" as learning low-code 'on-the-go' in an organization is rarely done. FG1 explained: "*Sometimes they hire external parties specialized in certain LCDPs. Or in a large organization, they sometimes have some teams that can be hired temporarily*", and FG3 told:

"*So they always have an outside party that at least does the startup. I haven't heard very often that they do it themselves*". We, therefore, decide to expand this factor to 'Available IT knowledge' to emphasize that an organization needs access, in some way or the other, to IT knowledge capable of implementing low-code BPM.

Literature had provided two other social factors, namely ***trust*** and ***innovation-driven culture***. Firstly, literature showed that trust under developers and middle managers in low-code is significant as employees do not feel inclined in changing their existing way of working. Therefore, they need to be convinced of the potential capabilities and way of working with the technology. FG1 doubted trust's relevance, as: "*But maybe not necessarily very relevant because if you have IT creating an application and it's well thought out by someone from the business, I don't think it's as relevant whether all employees have trust*". However, we believe the participant missed the point that trust also concerns the IT department and whether they are convinced with the new approach for developing IT. Further arguments were not given for trust, and the innovation-driven culture was also not mentioned during the focus group, meaning they remain unchanged.

Original			New	
Factor	**Definition**		**Factor**	**Definition**
Existing Software Development Knowledge	Whether the employees in an organization have low-code skills, or at least software development knowledge to develop a BPM application through an LCDP for BPM		Available IT Knowledge	Whether the organization has access to employees with low-code skills or at least software development knowledge
Trust	Whether developers or middle managers have trust in the potential, capabilities and future of LCDPs for BPM before the adoption-decision is made		[remains unchanged]	[remains unchanged]
Innovation-Driven Culture	Whether an organization motivates innovation and consists of eager-to-learn employees that want to learn and develop with LCDPs for BPM		[remains unchanged]	[remains unchanged]

Table 9. Social dimension factor changes

5.2.6 Environmental dimension

Lastly, facets between the organization and its environment can be decisive in adopting low-code, or not. Having ***existing external implementation partners*** and being in a ***high-paced and changing environment*** is found to be beneficial according to literature, and the focus group participants seemed to agree. The earlier statements about 'Available IT knowledge' emphasize the benefit of already having an external implementation partner that has the expertise to implement low-code BPM technology. To add to this, FG2 emphasized the difference between low-code BPM development and low-code application development: "*We now really focus on low-code BPM tooling or platforms, which is, in my experience, actually always implemented through external partners, or internal teams that are certified*

[…] What you might see with PowerApps is that it's so easy that you do see more people clicking together some little flows here and there". Moreover, a changing environment was proven as the best environment for low-code technology: "*If you say It has to be ready within 3 months, yes then you may have a drive to it. But, for example, in the government, right? Then it often really takes a year or 1.5 years to get something changed. Yet they seem to accept this*." (FG3). Taking all into consideration, we keep these factors as they were and include them in the following version.

We do, however, add a new factor to the framework. From the 'unique process' factor, it became clear that the customizability of low-code is a selling point. Apart from the process itself, an IT solution could also require unique features and capabilities from the organization's requirements. FG3, for example, shared: "*In a port, for example, [company] has a very specific maintenance system for large ships. No other company has that. There they say well, we're going to build low-code for that because there is no party that has that on the shelf*". Moreover, recall the previous example of the organization that combined traditional and low-code development for their financial processes "*They used that because they want to make a lot of specific custom changes in there, so to speak, and they don't have some kind of package in the market that can do all that for them*" (FG3). McManus (2003) said that: "IS executives agree that certain unique business applications will necessitate creating software in-house, regardless of time or cost considerations". Low-code development enables the creation of these unique business applications in-house together with time- and cost-savings. Therefore, we add ***lack of existing suitable solution*** as a key factor that could motivate low-code BPM adoption.

Original			*New*	
Factor	**Definition**		**Factor**	**Definition**
Existing External Implementation Partners	An organization already has existing partnerships with LCDP vendors or other low-code specialists		[remains unchanged]	[remains unchanged]
High-Paced and Changing Environment	An organization operates in a highly variable industry which is prone to technological or regulatory changes		[remains unchanged]	[remains unchanged]
			Lack of Existing Suitable Solution	The market is not offering a solution suitable to fully tackle the BPM initiative

Table 10. Environmental dimension factor changes

5.2.7 General feedback on the framework

Although the main goal of the focus group was to validate the framework's content, some indications of the framework's usability can be derived.

Firstly, the difference between a key and a reflection factor seemed to be clear as participants sometimes questioned whether a factor was the one, or the other. At some moment, however, a beneficial factor was discussed as if it were a key or reflection factor. Also, a participant forgot the explanation behind

a beneficial factor. This is understandable as the key and reflection factors are more deterministic while a beneficial factor is more positivistic. However, the difference should be emphasized more clearly in the refined framework version.

The latter could also minimize some misunderstandings where participants confused the terminology between dimensions and factors. Now, the model was seen as a “wheel” or “ball”, which could explain why the various concepts in the framework became interlaced and vague. Therefore, in the following version of the framework, we aim to segregate various concepts more clearly for the sake of the easiness and operationality of the artifact.

5.3 Towards the refined framework

A refined framework version was created following the validation of the content and general feedback on the framework by the focus group participants. It is presented in Figure 4 and we discuss several design decisions.

In the initial framework version, the framework’s design only included the factors’ titles, not their definitions. Therefore, at the beginning of the focus group, we provided Table 3 with its definitions to the participants. This meant that the participants had to use four pages and navigating the framework was difficult. Hence, in the refined version, we combine the factors and their titles on one page, and layer or group factors from the same dimensions together to aid the framework’s operationality and ease of use.

Secondly, the feedback from the focus group demonstrated that using the different factor types interchangeably caused some confusion. In the refined version, we segregate the factors by their type in order to provide structure to the framework. Users can now interpret each ‘part’ of the framework in its own way. In the following evaluations, this hypothesis needs to be tested.

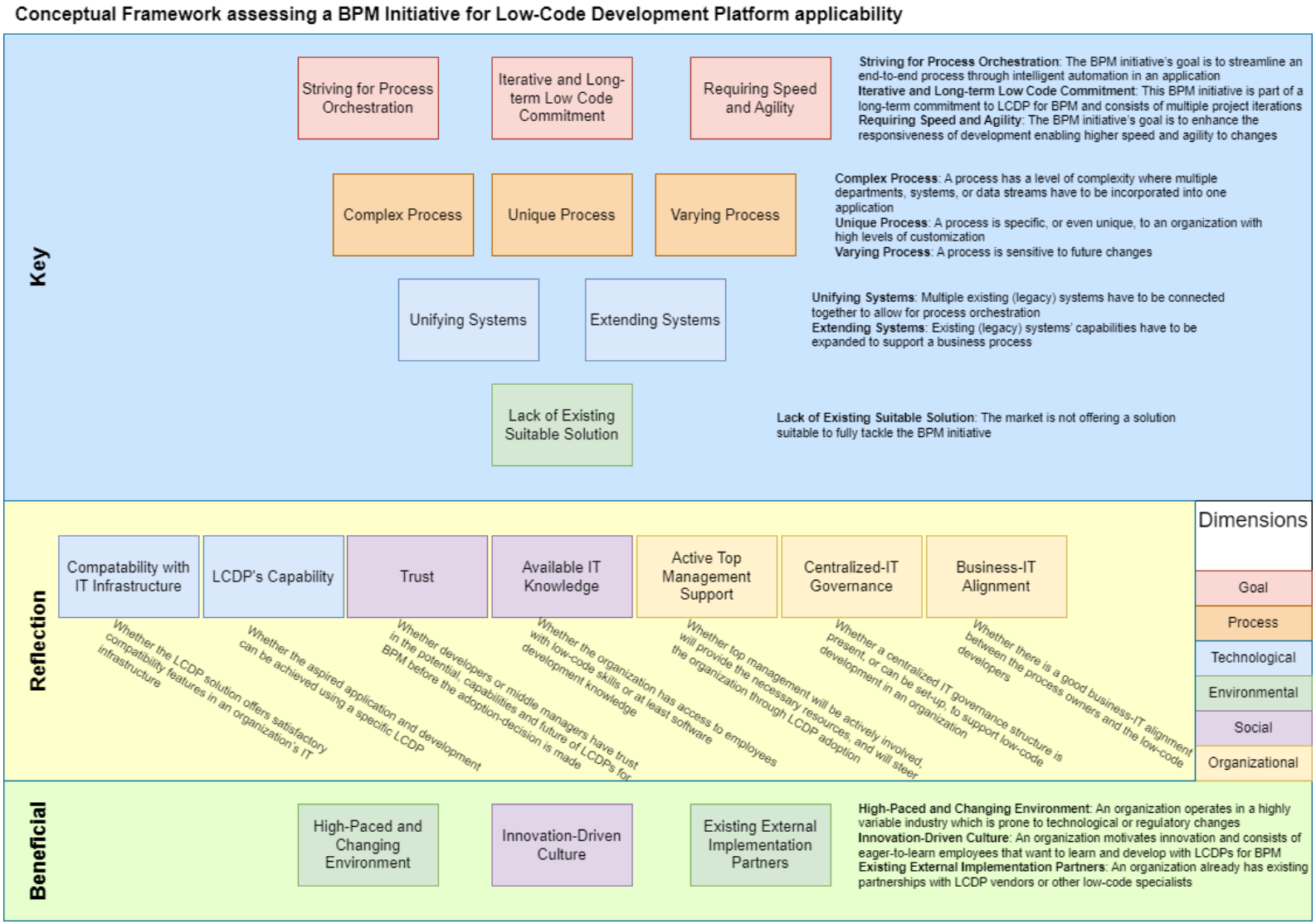


Figure 4. The second version of the conceptual assessment framework

6 Final framework – Expert interviews

Following a literature review and a focus group, we validate our framework for the third, and last, time. We employ expert interviews as the final instance of the 'design and development' phase in our DSRM (see Figure 1). As the framework becomes more complete and user-oriented following the previous validation, expert interviews allow us to go more in-depth into topics close to the participant's expertise. This further strengthens the completeness and internal consistency of the framework. As the experts have different backgrounds, we also requested feedback on the framework's usability to evaluate its usability by different end users. This chapter reports the results of the expert interviews together with the full explanation of each factor, as presented in the final framework version. Lastly, the expert's comments regarding usability and the design decisions leading to the final framework are explained.

6.1 Expert interviews overview

Eight expert interviews were separately conducted for this validation. The initial set of interview participants was found through the researcher's network resulting in a combination of low-code implementation specialists, developers, and managerial decision-makers. Afterwards, participants were asked for other contacts with experience in the field of low-code for BPM. The summary of all participants is given in Table 11.

Prior to the interview, the participants received a brief overall explanation of this study and our conceptualization of low-code BPM. In the first part of the session, each participant was asked about his or her own experience and opinion on what factors determine low-code BPM's effectiveness. Thereafter, the framework was introduced and evaluated on its content and usability. The full expert interview protocol can be found in Appendix C.

#	Current role	Experience with low-code BPM
I1	Chief Technical Officer	Setting-up, implementing and overseeing the use of Mendix in their whole organization
I2	Director and IT implementation specialist	Consulting organizations with decision-making and managing IT implementations through OutSystems
I3	Lead Appian consultant	Leading the implementation of Appian throughout a large corporate and consulting small-medium enterprises on Appian
I4	Freelance Business & Digital Transformation expert	Leading the implementation of various LCDPs in various organizations
I5	Appian developer	Building Appian applications for various organizations
I6	Interim Chief Information Officer	Managing and decision-making on IT implementations through various LCDPs
I7	Partnership manager in Intelligent Automation	Connecting organization needs with Intelligent Automation and low-code BPM vendors
I8	Lead in consulting for Intelligent Automation	Setting-up a low-code BPM department and consulting on Intelligent Automation opportunities

Table 11. Overview of expert interview participants

6.2 Results

The following sub-chapters explain the rationale behind the factor and factor definitions as seen in the final version of the conceptual assessment framework. As in the previous chapter, we summarize the findings under each dimension at the end of each sub-chapter. These can be found Tables 12-17

6.2.1 Goal dimension

PROCESS ORCHESTRATION – *"The goal is to orchestrate (a part of) a process in a system, through a defined workflow where tasks are distributed to all process actors, data from different sources is integrated into a single point of truth, and the system offers performance and case management and monitoring capabilities for stakeholders".* All the experts agreed that one of the main reasons for adopting low-code BPM is to achieve control over a business process through process orchestration. I8 exemplified this: "*The entire case was managed by a workflow, by a process, right? So the notion of tasks, and I distribute those tasks, I assign them to people, to groups, they take ownership of that. That's kind of the core of the whole process orchestration, right? People in the loop*". On the one hand, I3 explained that these platforms "*are very strong […] in making transparent what steps such a process must then go through.*", but also "*getting insightful data where process are […] and if we want to make data from different sources visible in a good way*" (I3). If an organization "*is trying to orchestrate something between several individuals, several teams…*" (I8), a low-code BPM solution seems very suitable. Lastly, when teams notice that "*We see that we are duplicating a lot of things, maybe making a lot of mistakes*" (I7), an orchestrated system makes sure that "*this follows a process to make sure that it's done correctly*" (I8). Therefore, our definition of the 'process orchestration' goal further includes the control, disparity in process actors, integration of data and systems, and monitoring capabilities.

The original 'Requiring Speed & Agility' factor is split in two as these have separate benefits and perspectives, and an organization might specifically seek one of the two.

SPEED – *"The goal is to increase the speed of development, maintenance and deployment"*. I6 provided an example where speed was the core goal: "*At [organization] the reason was that we wanted to develop relatively quickly*", and I3 substantiates this as: "*you can deploy a low-code BPM platform quickly in the short-term between systems, in a maintainable manner*". I2 argues that low-code systems differentiate themselves on speed: "*The competitive edge [of low-code systems] is found in the speed and the agility it offers […] where you talk about implementation periods of about 1 to 3 months and then you should have a product with business value*". When building a system yourself, "*your development time and effort, and chances of failure are much higher*" (I3), and as I1 said: "*I do not want to wait 6 months before something is set-up*". Hence, organizations seeking speed in development, maintenance, and deployment should consider low-code technology.

AGILITY – *"The goal is to have a more agile infrastructure that can adapt and grow with changing business needs"*. Experts unanimously named agility as one of the primary drivers for low-code BPM technology. On the one hand, "*one very nice thing about low-code is that you can build a system that evolves with you, grows with changing business. And with commercial-of-the-shelf you have that to a lesser extent*" (I2). On the other hand, "*We have used it mainly for where changes are expected quickly, for example, legislative changes. Or where the consumer demands something different all the time, use low-code for that. In a changing world you absolutely cannot serve with a standard package […] whereas, in low-code, you build something, you discard it or you modify it, and you can measure and move with it*" (I4). An organization that needs such agility can use this competitive edge of low-code BPM platforms for its benefit.

REDUCED COSTS – *"The goal is to reduce overall costs, through less maintenance, smaller teams, and lower licensing costs, as long as committed on the long-term and supported by a large business case"*. A new factor added to the framework is the goal of reducing costs. As I6 argued, "*Costs saving is often another reason to choose low-code. […] I think, generically, you can say that with low-code you have the potential to develop applications faster at a lower cost. If you have a lot of component reuse and you can develop your new application faster and faster, that is cheaper than coding an application each time […] Because you need fewer people, you have lower licensing costs, less maintenance, et cetera.*". To become cost-effective, low-code development should be part of a long-term IT strategy where the organization continues actively developing using low-code, experts state. I8 explained: "*But in general, if you buy them [LCDPs] to do one thing, then obviously that'll cost you more […] As you start scaling up and getting a few apps, then you start seeing economies of scale. Essentially it's like I'm using one platform to do multiple things, so I'm not paying for a new license each time*". Therefore, if an organization is planning to apply low-code BPM on a large business case, then in the long-term there is potential for cost saving.

In the previous version, the 'Iterative and Long-term Low-Code Commitment' factor was included. However, as pointed out by I6, "*Iterative and Long-term low-code is not a goal, that is a condition […] Why would I want to start a low-code project? Because I want better process orchestration and that I want more speed and agility. Or I want to save costs, could also be. Those are the goals why you want to apply low-code. Iterative and long-term low-code commitment is never a goal*". We tend to agree that this factor should not be interpreted as the goal of a BPM initiative. 'Iterative' is a way of development and 'Long-term Low-Code commitment' is an organizational decision. Therefore, the first is incorporated under the social dimension and the latter is defined as a factor falling under the organizational dimension, described in the next sub-chapter.

Original			*New*	
Factor	**Definition**		**Factor**	**Definition**
Striving for Process Orchestration	The BPM initiative's goal is to streamline an end-to-end process through intelligent automation in an application		Process Orchestration	The goal is to orchestrate (a part of) a process in a system, through a defined workflow where tasks are distributed to all process actors, data from different sources is integrated into a single point of truth, and the system offers performance and case management and monitoring capabilities for stakeholders
Requiring Speed and Agility	The BPM initiative's goal is to enhance the responsiveness of development enabling higher speed and agility to changes		Speed	The goal is to increase the speed of development, maintenance and deployment
			Agility	The goal is to have a more agile application that is able to adapt and grow with changing business needs
			Reduced Costs	The goal is to reduce overall costs, through low licensing costs, less maintenance and smaller teams, as long as committed on the long-term and supported by a large business case
Iterative and Long-Term Low-Code Commitment	This BPM initiative is part of a long-term commitment to LCDP for BPM and consists of multiple project iterations			

Table 12. Final goal dimension factors

6.2.2 Organizational dimension

LONG-TERM LOW-CODE STRATEGY – *"Adoption of low-code development should be part of a long-term (IT) strategy to surpass the start-up costs of laying the initial business-, data-, and IT-structure, and begin achieving the benefits of lower costs, fast and Agile development"*. As previously mentioned, we argue that the organization should embrace low-code in a long-term (IT) strategy to achieve the proposed benefits. I6 explained that "*Building one application with low-code is really a total waste of money, you really need to have a strategy […] When you start, it all goes much slower than you hope, it takes a lot of energy, effort, and frustration. But the second application the process is already better and then you can use 30% of what you built in the first application. In the third application it is even faster. Only when you get to 5/6 applications, then you suddenly see the added value of low-code*". Through a long-term strategy, benefits such as speed, agility, and reduced emerge. Moreover, "*If you train people in certain tooling, it only makes sense if that's long-term, if there's a long-term vision there. Or you say, we'll bring in a partner with whom we'll do a kind of partnership to make that a success. But still, in a partnership, you also only do that for the long term*" (I3). In the end, "*If the commitment*

isn't there, it's a waste of time" (I6). So we believe that at the core of most cited benefits, a long-term strategy should exist.

BUSINESS-IT ALIGNMENT – *"A good business-IT alignment, where the parties have pre-agreed responsibilities, expectations, and work agreements, is needed to further narrow the gap between business wishes and IT development"*. It was apparent, from the beginning, that a good business-IT alignment should exist to make low-code technology successful in an organization. Experts underlined it as "*I think [low business-IT alignment] is a very bad ground for a low-code platform. I think it wouldn't catch on*" (I2), "*It's a cultural aspect, you're suddenly going to put two cultures together and well let's work together. You have to guide that well*" (I6), and "*Certainly. Yes they have to be in agreement in the organization […] Definitely in the early stages it plays very much between the user, super user, and IT*" (I7). Moreover, low-code is seen as a tool to further bring the two departments together as well. For I1, it was one of the reasons they continued with low-code: "*Okay, are we going to continue with this? I think these benefits can be found: [...], and ability to involve the rest of the organization in the development process. So let's go, let's stick with it*". I2 summarized this point: "*So is it an enabler or is it a driver for business? Both!*". Therefore, we say that business-IT alignment is an enabler that an organization should consider, and should see as a driver forward to even more business-IT alignment.

CENTRALIZED GOVERNANCE – *"A minimal centralized governance is necessary to lay down the standards for development, user rights, and resources. With the growth of low-code adoption, this governance needs to be expanded and can be decentralized"*. Literature and the focus group has shown that governance is key when giving business and IT the opportunity to develop. Without it, security risks arise and "*If you let everyone onto the platform, you get uncontrolled growth*" (I3) and "*in a very short time span, you can get a lot of misery*" (I2). It is "*crucial to set-up certain [development] standards*" (I3) but also "*you need good governance about the money. Who are all involved? Who are developing here now?*" (I6). Governance should, therefore, look broader than just the low-code development itself. Even though you might be starting small, "*Of course, you can say, yes, if we start within a team or within a department, it is a bit of an overkill to set up a dedicated device like that. But think about it, right? […] even though you start off small, take it with you from the beginning*" (I3). Experts also agreed with the focus group's conclusion: "*In the starting phase: central, standards, laying a foundation. When building on, then decentralized is really perfect*" (I4).

TOP MANAGEMENT UNDERSTANDING & SUPPORT – *"Top management needs to understand the pros and cons of the technology, trust that benefits arise in the long-term and support the initiative with involvement and necessary resources"*. The essence of this factor is still very alike its predecessor the 'Active Top Management Support'. It remains key that the top management, making the decision to adopt low-code, is actively involved and supports the initiative with resources. Even further, "*a key*

success factor is that you have to have everyone along, that the board has to be along too, that they have to understand why it's important, and that it takes time, and it will not be so quick in the beginning" (I6). Therefore, we accentuate the understanding that the top management should have on how the cited benefits of low-code come about. As I2 explained: "*And yes, we will soon have much more flexibility and be able to build applications faster, but by then we will be a year further. While they don't realize it's not magic.... That's expectation management anyway, that's just very important*". Hence, the 'active top management support' factor has been further refined with an understanding-aspect.

Original			*New*	
Factor	**Definition**		**Factor**	**Definition**
			Long-term Low-Code Strategy	Adoption of low-code development should be part of a long-term (IT) strategy to surpass the start-up costs of laying the initial business-, data-, and IT-structure, and begin achieving the benefits of lower costs, fast and Agile development
Business-IT Alignment	Whether there is a good business-IT alignment between the process owners and the low-code developers		Business-IT Alignment	A good business-IT alignment, where the parties have pre-agreed responsibilities, expectations and work agreements, is needed to further narrow the gap between business wishes and IT development
Centralized IT Governance	Whether a centralized IT governance structure is present, or can be set-up, to support low-code development in an organization		Centralized IT Governance	A minimal centralized governance is necessary to lay down the standards for development, user rights, and resources. With the growth of low-code adoption, this governance needs to be expanded and can be decentralized
Active Top Management Support	Whether top management will be actively involved, will provide the necessary resources, and will steer the organization through LCDP adoption		Top Management Understanding and Support	Top management needs to understand the pros and cons of the technology, that benefits arise on the long-term and support the initiative with involvement and necessary resources

Table 13. Final organizational dimension factors

6.2.3 Process dimension

COMPLEX USER-CENTERED PROCESS – *"The process has a high level of human input and spans over multiple departments, systems, and data sources"*. The previous definition already highlighted that low-code BPM is very suitable for processes with dispersed actors and systems. I7 explained a similar situation where "*If the complexity is that you have many more different parties or have components that don't communicate well with each other, I would at least think about it. Low-code could indeed possibly*

be the solution for you". But, experts stated that human interaction is especially a key indicator for low-code development. I3 explained that "*The use cases that are often appropriate are processes where users are involved. So where there are steps in that somebody has to review something, approve something, and enrich data. And that can be external or internal employees*", and I8 confirmed it as: "*You have platforms that are made to simply integrate systems […]. In those, you missed the whole human interaction and orchestration bit because they're just there to exchange information between systems as opposed to having humans in the loop right? So, so that's a different use case*". Therefore, we refine the previous 'complex process' factor and add the user-centered dimension as emphasis.

UNIQUE PROCESS – *"The process requirements are highly customized, or even unique, to the organization, and it is part of the organization's key offering in the market"*. The customizability, that a low-code BPM platform provides, makes it suitable to support any variation of a business process. This gives low-code solutions an edge and "*You do it because of the unique process. Otherwise you shouldn't do it*" (I2). An example of a unique process was already given in the focus group, but I4 also added: "*There we actually chose low-code because.... what [company] did, they're the only ones doing that in the Netherlands, and only a few in Europe*". I1 explained their situation where they had to support a unique process for each of their partners, making low-code an ideal solution. Apart from just being unique, experts argue that "*The IT system has to be part of the unique proposition that the customer has, right? Being an integral part of that business, that the system should help that customer differentiate itself from the rest of the market*" (I2). Organizations should, however, not overestimate their uniqueness, as I7 said: "*I also think, every company always thinks they have such a super unique process and when we start working with low-code, I think they also very often find out, oh yeah...*". Thus, if an organization is convinced that they are working with highly customized processes that are vital to them, low-code could be a great solution.

CHANGING PROCESS – *"The process requirements are sensitive to variations in law/regulation, user-, stakeholder-incentivized or market changes"*. The agility that low-code provides makes it exceptionally suitable for changing processes. As I7 put it: "*A low-code BPM platform, on the other hand, it's made for change […] The regulator changed the rules sometimes. Oh, now this rule is not like this, and I need this to come out, and I want this information. So you have to change, you're constantly changing it*". The same was mentioned by I6: "*Is it a solution that is in your core process and does not change much? Then I would take a SaaS solution or an ERP solution, or at least standard a solution. Especially not low-code...*" and I4: "*It never changes – that might be an important point [to think about]*". Therefore, the essence of this factor remains the same as in the previous framework.

Original			*New*	
Factor	**Definition**		**Factor**	**Definition**
Complex Process	A process has a level of complexity where multiple departments, systems, or data streams have to be incorporated into one application		Complex User-Centered Process	The process has a high level of human input and spans over multiple departments, systems, and data sources
Unique Process	A process is specific, or even unique, to an organization with high levels of customization		Unique Process	The process requirements are highly customized, or even unique, to the organization, and it is part of the organization's key offering in the market
Varying Process	A process is sensitive to future changes		Changing Process	The process requirements are sensitive to variations in law/regulation, user-, stakeholder-incentivized or market changes

Table 14. Final process dimension factors

6.2.4 Technical dimension

INTEGRATING THE IT ENVIRONMENT – *"The IT supporting the process is scattered and a layer-on-top integrating the (data) systems is required to aid the users"*. Low-code BPM applications are often seen as a layer that resides above the underlying systems with which the process actors interact. I2 shared a customer's use case where a process needed multiple different systems, and "*He asked: Can you put a layer over that integrates [those systems]? And, we then did that with OutSystems, so we created that layer to all the links to the individual systems underneath*". Similarly, low-code can integrating databases and data sources: "*We were working with an organization where they're a very decentralized organization, they've got systems all over the place, they're very scattered landscape […] Data and the transparency, and drill down capability into their data is crucial for them to do their job. They don't have any data and insight into what's going on*" (I8). This is an example where low-code BPM tooling can help as it provides the ability "*to leverage information that already exists and maintain a single source of truth*" (I5). In the end, the core idea is "*pulling information from different systems, forming it into a unified process in that one layer which means you have a lot more control over how that process goes*" (I7). Therefore, an organization that has a high need for an integrated layer as described above should consider the possibilities of low-code BPM.

EXTENDING EXISTING (LEGACY) SYSTEMS – *"Existing (legacy) cannot fully support the process, so their capabilities have to be expanded by another solution"*. Logically, an organization initially judges how it can support its process in its current IT landscape. However, "*uncontrolled processes that you cannot get right in your own systems, that is where you can use low-code as a kind of band-aid*" (I4). As I3 proposed: "*You can also implement it as an interface layer over a legacy system, for example*". This way, "*With low-code, access is easier. Because, first, you actually put a layer above it and then, on the other side [in your legacy systems], you do not necessarily have to change a lot of things*" (I7).

Of course, you could decide to replace legacy systems "*But then you start converting data, which are often expensive big projects*" (I4). So, when a lot of legacy systems in an IT environment is the problem, low-code BPM systems can communicate and integrate with existing legacy systems, extending the capabilities.

Certain studies stated that the compatibility of LCDPs can be an issue in an organization's IT landscape. During the focus group, FG1 briefly doubted this but no further arguments were given against this being an important factor to consider. I1, I2, and I6 discussed this factor and disregarded compatibility as being an issue. I1 stated: "*Well there may be some trade-offs because you have IT organizations that say you have to run all on-premises. But all of that can be done. I've never actually experienced that it couldn't. You can install it all in your own stack, the way you want it*". I2 explained: "*Nowadays everything you build lands in an existing IT landscape and it has to connect to that. In fact, that's what they aim for, because they know it themselves. So they have standard modules that do an SAP integration, for example*" and I6 confirmed: "*You do have to think about it carefully. But it's definitely not a problem*". We, thus, argue that compatibility issues could have been the case in the past as low-code BPM was developing. However, now, compatibility does not have to be a factor in the model.

STANDARDIZED DEVELOPMENT – *"Quick development is possible due to the use of provided standardized building blocks. Organizations need to assess the trade-off between high maintainability, reusability and quality, opposed to limited functionality with low-code"*. The 'LCDPs capability' factor, in the previous framework version, encouraged an assessment of the platform's capability and whether it can achieve the desired solution. If you look from a performance perspective, experts argue that LCDPs do not lack in that respect. I2 explained "*Those low-code platforms run on cloud platforms, so I can always add CPUs […] It could become a problem, but it is always solvable* " and I1 shared that "*we have tested this, analyzed it, and that the limits I expected are not there yet […] Well if you take certain wrong turns, then you notice it right away*". The last point is inevitably important as "*you should always make sure that an application you build is deployed in a scalable way*" (I3). The only limitation mentioned by I3, I5, I6, and I8 is that low-code BPM systems are not designed as "*a big data program*" (I5) or "*suitable to high transaction volumes*" (I6).

If you look from a capability perspective, I5 summarized low-code development very clearly: "*The disadvantage is also very much an advantage. You're just much more limited in what you can and can't do. So you use certain functions and certain standard blocks and you link that together. But because of the blocks, you can do a lot less right away, which also means you have a lot fewer bugs. So you have a lot fewer problems to solve but it does limit what you can and cannot do*". In other words, you could be limited in what you can build, but this also increases the maintainability a lot. Moreover, standardized building blocks bring quality: "*the low-code blocks have very good quality, you already can't make mistakes in that*" (I4); and reusability: "*Reusability is a key component, so if you make a particular*

integration with a known system, you don't want each team to do that themselves, you want it set up in one place and one time" (I3). All in all, organizations should assess whether an LCDP vendor is designed for and offers the functionality to meet the system's desired requirements.

Original			*New*	
Factor	**Definition**		**Factor**	**Definition**
Unifying Systems	Multiple existing (legacy) systems have to be connected together to allow for process orchestration		Integrating the IT Environment	The IT supporting the process is scattered and a layer-on-top integrating the (data) systems is required to aid the users
Extending Systems	Existing (legacy) systems' capabilities have to be expanded to support a business process		Extending Existing (Legacy) Systems	Existing (legacy) cannot fully support the process so their capabilities have to be expanded by another solution
Compatibility with IT Infrastructure	Whether the LCDP solution offers satisfactory compatibility features in an organization's IT infrastructure			
LCDP's Capability	A process is specific, or even unique, to an organization with high levels of customization		Standardized Development	Quick development is possible due to the use of provided standardized building blocks. Organizations need to assess the trade-off between high maintainability, reusability and quality, opposed to limited functionality with low-code

Table 15. Final technical dimension factors

6.2.5 Social dimension

ATTAINABLE LOW-CODE WORKFORCE – *"The organization is able to find and employ experienced low-code development and IT management employees, and continue growing the internal low-code know-how"*. Implementing any IT solution requires sufficient know-how inside of the organization. Low-code development has lowered the barrier and is promoting citizen development through its easy-to-use drag-and-drop features. However, experts say "*that citizen development, that is a bridge too far*" (I3) and "*I tend to disagree. I think no, not anybody can do it. If you want to do it properly, then you need to know what you're doing*" (I8). However, for low-code knowledge, I6 argues: "*the whole market is still catching up in that respect. […] There are not that many people who have enough experience, so there are few people who have at least 3 years of Mendix or OutSystems experience*". I4 adds on top of this: "*and especially Appian is hot but [there are a] few good people, so don't start. Only start if you are sure you have a group of at least 3 very strong, experienced people*". The problem is "*Suppose you start using Appian in the Netherlands, there are no good people left to get because they have now been bought up by [company X], [company Y], and others*" (I4). IT does not only apply to IT knowledge, I6 argues it also concerns "*Solution architects and business analysts are going to be crucial. And you also have to have very good architects, lead developers, very good project managers*". I2 concurs with this: "*That bottleneck is not in the realization team. Your bottleneck is with the product owner and the test*

team". All in all, I1 argues this is a key factor: "*That [necessary know-how] is definitely a thing. You absolutely have to factor that into your imagination, into your decision-making*". Based on the experiences of experts and the previous knowledge from literature and the focus group, we argue that this is a key factor in the low-code market that is still growing and maturing.

INTRINSICALLY MOTIVATED EMPLOYEES – *"The involved business and IT employees need to trust and have an intrinsic motivation to adopt low-code development, where business users will be more involved with IT requirements and IT users will be more involved with translating business needs"*. This new factor incorporates various aspects of the 'Trust' and 'Innovation-driven culture' factors from the previous framework. As I3 put it: "*Because people just inherently have an aversion to new technology*", you need employees that have the drive to work with low-code technology. I1 explained that in their organization "*That [intrinsic motivation] just has to be there. If you have a lot of people, who are building macros themselves in Excel, things like that, groups like that, you just have to give better tooling right away*". As low-code moves away from traditional programming, your IT employees should support this new development. "*You cannot dump a completely new tooling on your Java group, that just does not work*" (I1) as "*developers who do a lot of Java now, for example, think that with low-code they will be very limited in the possibilities they have*" (I3). So you need intrinsically motivated IT employees, but also business employees. As I6 explained in his experience: "*Business is also going to have to put a lot more time in specifying and prioritizing user stories […] if they do not want to put the time in that, then it still is not going to work*". Therefore, you need business "*who also has some kind of intrinsic motivation to do something with it, to bring an improvement*" (I7). We refine 'intrinsically motivated employees', thus, to highlight how both IT- and business employees need to have the drive to use low-code to its full potential.

Original			*New*	
Factor	**Definition**		**Factor**	**Definition**
Available IT Knowledge	Whether the organization has access to employees with low-code skills or at least software development knowledge		Attainable Low-Code Workforce	The organization is able to find and employ experienced low-code development and IT management employees, and continue growing the internal low-code know-how
Trust	Whether developers or middle managers have trust in the potential, capabilities and future of LCDPs for BPM before the adoption-decision is made		Intrinsically Motivated Employees	The involved business and IT employees need to trust and have an intrinsic motivation to adopt low-code development, where business users will be more involved with IT requirements and IT users will be more involved with translating business needs
Innovation-Driven Culture	Whether an organization motivates innovation and consists of eager-to-learn employees that want to learn and develop with LCDPs for BPM			

Table 16. Final social dimension factors

6.2.6 Environmental dimension

LACK OF A SUITABLE OFF-THE-SHELF SOLUTION – *"The market does not offer a suitable off-the-shelf solution, that the organization is willing to adopt, to fully tackle the BPM initiative"*. Previously, the suitability of low-code BPM platforms for unique processes had been highlighted. Unique processes are rarely supported by existing packaged software as I1 gave an example: "*Before starting [company], [person] had an insurance business. His organization spent a very long time looking for a package for settling those claims to get a handle on the backend of that process. They didn't find anything on that*", I4 explained: "*I wouldn't use it when there is just perfect software that fits me exactly*". On the other hand, the emphasis that we add to this factor is that a packaged solution could exist but is not fully suitable to the organization's requirements. Therefore, I4 mentioned "*Note, or just not wanting to accept the [existing suitable solution] […] And I would put it even more extreme. Even if it is 80/90% satisfactory, I still wouldn't recognize it as suitable*". Although the latter percentages are subjective, this factor emphasizes that organizations should search for existing solutions and assess whether these fully suit their requirements. If none match, low-code development platforms offer a fully customizable solution.

VENDOR & PARTNERSHIP SELECTION – *"Low-code development relies on vendor's LCDP capability and is regularly performed with implementation partners. Organizations need to conduct a rigorous selection of vendors and partners that are experienced in their industry and reliable in the long-term"*. Implementation partners were previously highlighted as crucial for low-code development, especially for low-code BPM. Experts concurred with this: "*You do need someone with knowledge looking over your shoulder and explaining to you how to do things and how not to do them. Because if you have to do it all yourself, if you have to reinvent the wheel, it will take a long time*" (I5). Furthermore, apart from solely development in low-code, "*I would definitely recommend always having a lead designer or lead consultant, so someone who has done multiple implementations [...] for things around scalability, but also how do you set up such an application in a good way, that certain components are also reusable. And well, that in terms of security you think about this well in advance*" (I3). Besides implementation partners, assessing the vendor is also key. A technical assessment was discussed in the technical dimension, but organizations should also assess the vendor as an organization, as I6 explained: "*If you build an application in low-code and you think after two years: I don't want to work in Mendix anymore, But I want to work in another language. Or do I want to go from Mendix to OutSystems? Yes, you cannot*". On the one hand, organizations should be wary of vendor lock-in. On the other, they could assess which vendors are specialized in their industry: "*There were more Mendix partners with insurance experience than OutSystems partners with insurance experience*" (I6). Having vendors with such specializations brings advantages, as I1 explained: "*That [insurers] I think is their core group. That opens a lot of doors for us. A lot of technological assessments, they've already done those for us.*

We piggyback on that". Taking all into account, organizations need to analyze what implementation partners and vendors are available in their industry.

The previous version of the framework included the beneficial 'high-paced and changing environment' factor. Experts generally agree, as I5 said: "*Low-code is really even better in a changing environment than in a stable environment through*" and "*I think many companies struggle with it, to keep up with the speed of the market, and the demand, or expectation, that customers have*" (I3). The reason for a changing environment was often found in "*market changes or legislation changes*". Yet, we have decided to remove this as a factor as the 'Varying Process' factor covers all arguments that the experts had given. Moreover, we agree with I7: "*High-paced and changing environment? Yes, they all are these days and they will all say so*". Therefore, we suggest that having a 'Varying process' is more telling than being in a 'high-paced and changing environment'.

Original			*New*	
Factor	**Definition**		**Factor**	**Definition**
Lack of Suitable Solution	The market is not offering a solution suitable to fully tackle the BPM initiative		Lack of Suitable Solution	The market does not offer a suitable solution, that the organization is willing to adopt, to fully tackle the BPM initiative
Existing External Implementation Partners	An organization already has existing partnerships with LCDP vendors or other low-code specialists		Vendor & Partnership Selection	Low-code development relies on vendor's LCDP capability and is regularly performed with implementation partners. Organizations need to conduct a rigorous selection of vendors and partners that are experienced in their industry and reliable in the long-term
High-Paced and Changing Environment	An organization operates in a highly variable industry which is prone to technological or regulatory changes			

Table 17. Final environmental dimension factors

6.2.7 General feedback on the framework

The participants received the framework in the second half of the interview accompanied by a brief introduction. At the end of the interview, the participants were asked to evaluate the framework and give a proposed use of the framework. This sub-chapter describes the key findings.

Initially noted, at some moments during the interview, the experts lost sight of the framework's essence. This correlates with the feedback received from the focus group where the goal of the framework was a cause for questions. Therefore, we believe the framework should become more self-explanatory.

Moreover, multiple experts proposed to use the framework as a 'tick-off' list, where each factor is looked at in the context of an organization's use case. I2 explained "*Yes, you can check off on that. I think it would be nice for a lot of my clients to have*" and I7: "*Look, if I made it simple, this would be a*

very nice checklist […] for when we go to clients. So do you meet these points, is this indeed approximately what you want at key [factors]? We have reflection, well keep in mind that within the organization so you do have to start arranging these kinds of things". Two things we take away from this. Firstly, the framework meets its intended target use and can be pivoted towards a 'tick-off' or checklist model. Secondly, I7 implies simplifying the framework to make it more comprehensible to the end user. I7 further proposed "*I could have it as a slide in between the sales pitch*". For this purpose, the model would need some simplification through a redesign.

Thirdly, during the interviews, experts rarely mentioned beneficial factors unless specifically asked. Their comments were often unconvincing such as "*I don't think that's the prerequisite that you have an innovation-driven culture*" (I6) and "*I find beneficial [factors] very dependent per organization*" (I7). Moreover, I7 summarized his opinion "*I think those top two, key and reflection, I find those super interesting*". Therefore, when defining the final factors from the interviews, we have either incorporated beneficial factor aspects into other factors, or removed the beneficial factors.

6.3 Towards the final framework

The final version of the framework is presented in Figure 5. The most noticeable difference, compared to the previous version, is the lack of dimensions in the final framework. While the dimensions provided the theoretical foundation for the factors, they also categorized the factors which we imagined to be helpful for the end users. Initially, two versions of this framework were constructed where one included the dimensions, and the other did not. The researchers and various colleagues agreed that the simplified framework was easier to perceive and the dimensions did not add much to the framework. Moreover, the global 'themes' on the left still provide some context on the factor's origin.

Another difference between the final version opposed to the refined version is the lack of beneficial factors. In the previous sub-chapter, we explained why these have been integrated or removed based on feedback from the experts. Moreover, we argue that this helps to simplify the model, as was also stated by experts.

Furthermore, white boxes are added to the key factors as 'tick-off' boxes. This is included based on the overall framework feedback from experts. Reflection factors do not have this feature as these factors are more intended for understanding what low-code BPM entails and for reflecting on the use case. The segregation of factor types remains as this was understood by experts when assessing the framework.

The previous framework versions were not self-explanatory in the sense that the researchers had to explain the purpose and how to use it. Therefore, in the final version, we have added an explanatory introduction above the framework (see Appendix H).

Lastly, the framework's background colors have been simplified, also based on feedback from I6: "*I would only, just by way of design, not do yellow on yellow*" referring to organizational reflection factors in the previous framework version.

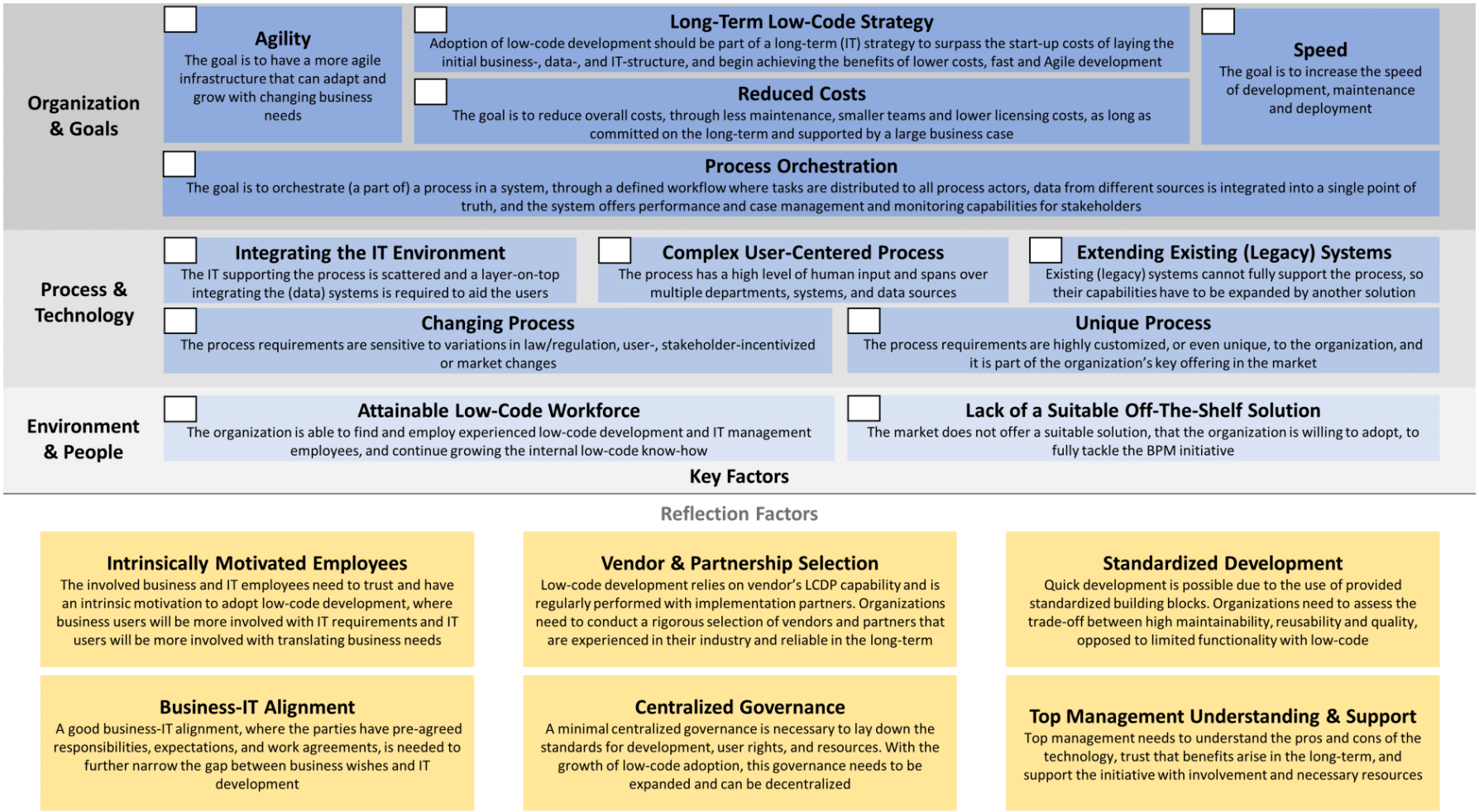


Figure 5. The final version of the conceptual assessment framework

7 Evaluation – Fictitious use case simulations

In the previous three chapters, we presented the design and development of our assessment framework. This chapter represents the fourth phase in our DSRM, where the final framework is demonstrated and evaluated by potential end users. The framework is measured on the evaluation criteria of completeness, internal consistency, operationality, and ease of use as presented in Chapter 3.1.2. Firstly, we describe what use cases were involved and which evaluation participants evaluated the framework. Thereafter, we present the evaluation results and conclude our findings.

7.1 Fictitious use case simulations overview

We evaluated the final framework by means of fictitious use case simulations. During each simulation, a participant had to assess whether a fictitious organization's use case was suitable for low-code BPM while using the framework as a reference.

For this, we used three real-life historical use cases that were anonymized, the use case descriptions of which can be found in Appendix G. Two use cases (UC1 and UC3) resulted in low-code BPM adoption in real-life. Whereas UC1 was a small organization with a very specific business model, UC3 was a large organization that needed a flexible, process orchestration solution. UC2 seriously considered low-code BPM but opted for an available off-the-shelf solution. Through the use of different real-life use cases with different adoption reasons, we aim to thoroughly evaluate all facets of our framework.

Five participants from the Digital Process Excellence team at KPMG participated in the evaluation. Each participant was randomly assigned to one of three real-life use cases. All participants had at least basic theoretical understanding of low-code BPM besides experience with other BPM-related technologies. We assume that decision-makers in organizations have similar characteristics making this a good representation of possible end users. A summary of participants is provided in Table 18.

#	Current Role	Experience with BPM	Use case
EV1	Consultant	RPA and process mining developer, and qualified in Lean Six Sigma	UC1
EV2	Senior Manager	Team lead of process mining and highly qualified in Lean Six Sigma	UC2
EV3	Consultant	RPA developer and participated in one low-code BPM project	UC3
EV4	Consultant	RPA developer and completed advanced training in low-code BPM	UC2
EV5	Manager	Overall sales experience in RPA, chatbots, and other BPM technologies	UC3

Table 18. Overview of evaluation participants

7.2 Results

Each following sub-chapter presents the results of one evaluation criteria as defined in Chapter 3.1.2. We have grouped the findings per criteria into overall themes for easier representation. Furthermore, we describe observations made during the evaluation sessions.

7.2.1 Completeness

The completeness criterion evaluates how all-encompassing and holistic the framework is.

Broad scope of factors – Generally, the participants found the framework to have a broad variety of factors and said that the dimensions represent the general topics that they would consider when assessing a low-code BPM use case. EV1 mentioned: "*The framework is full and complete. In terms of usefulness, I see good potential*", and EV2 agreed: "*If you look at the things here, you should get a good idea, as someone who knows about this, whether to proceed with it, yes or no*". EV3 summarized: "*You can ask many questions using this […] On the basis of this, I think you can give an excellent opinion whether or not it fits the organization at first glance*".

Theoretical preparation – Two participants see a possible use of the framework as some theoretical foundation when preparing for an assessment. EV5 mentioned: "*What you always do before you go to an interview is think about what you want to know. So it's nice to have a model to write down questions for yourself, so you won't be sitting there with the model herself*", explaining how the framework can be used when preparing. EV1 explained: "*In particular, it is [useful] to prepare yourself. Normally, you probably wouldn't go out with such a framework*". We, therefore, conclude that the framework is useful as a theoretical reference. EV5 also explained how the team often uses such frameworks as a first step: "*We present a certain vision or a certain model and then we dive deeper into that*". Although the latter points are more applicable to the framework's operationality, we argue that this highlights the holistic nature of the framework.

All in all, participants highly rated completeness with an average score of 4.4 and agreed that it is an all-encompassing framework to be used in the initial phases of assessing a BPM initiative.

7.2.2 Internal consistency

Internal consistency evaluates the framework's elements on clarity, whether they have the same level of abstraction and whether the relationships are consistent.

Clarity – Overall, the participants understood the factor definitions. The factor names, written in large bold letters, gave a quick, clear picture of the factors in the model as EV1 noted: "*I used it as a cheat sheet, where just the titles were enough for me to scan*". EV4 and EV5 explicitly said, after briefly reading the framework, they understood all factors and their underlying terminology. Others asked some

questions to the researchers during the session but these were mostly confirmatory. The overall wordiness of the factors was a concern, but this was generally experienced as an 'ease of use' issue.

Abstraction – The factor types and the three layers comprising the key factors were understood and accepted. However, EV1 struggled with the scope of each factor: "*Intrinsic motivation of employees I personally find too selective and could be expressed at a higher level, for example, the factor 'human being'. This also applies to the other factors*". During the assessment of the use case, EV1 did not question this factor during the simulation.

Factors relations – The various relationships between the factors caused the most concern. Various factors overlapped and how the factors were related to each other was unclear. EV5 noted: "*For example, 'Agility' and 'Changing process'. Those are somewhat in line with each other 'Lack of Suitable Solution'.... Yes, of course, that is another addition of the other*" and EV4 mentioned: "*The relationships are [clear] but what these relationships exactly entail is not revealed in this framework*". Moreover, the uniqueness of these factors for low-code BPM was doubted: "*Which questions/blocks are really specific to low-code BPM and which ones can you also apply generically to a digital stack? Because I see a lot of things that I think of, yeah, if you take away low code and I put chatbots, RPA, or something else there, then that also applies*" (EV5).

With these concerns, internal consistency scores lower with a 3.7. Regarding the abstraction level, we agree that the 'Intrinsically motivated employees' could be too specifically centered on one aspect, while the description draws a broader picture. The 'Agility' and 'Changing process' confusion is also avoidable. Whereas the first focuses on an organizational goal and the latter on a process characteristic, we agree that the definitions now seem alike. Lastly, some participants questioned how distinguishing certain factors were to low-code BPM. Although it is true that factors, such as 'Business-IT alignment', are more generally applicable to any technology adoption, the description of each factor explains why, and how, the factor is relevant in the context of low-code BPM. Possibly, the framework's factors need more elaborate descriptions and elaboration on their background knowledge.

7.2.3 Operationality

Operationality judges the framework's ability to assess the suitability of low-code BPM.

Assessment checklist – Some comments falling under the completeness criterium suggested the framework be used as a theoretical reference or for a 'first glace' assessment. However, some participants proposed other applications as well. EV1 used the framework as a checklist during the conversation, as explained: "*I took out that framework again during the interview to see, okay do I have all the points, have I walked through it all [...] then I'm also sure that I can give good advice instead of doing it on gut feeling*". The integrated tick-off boxes were also used by most participants. Some ticked

them off during the preparation, some added plus or minus signs, but EV2 also suggested: "*you could also score 1 to 10 in what measure of importance, or high, low, medium*", proposing a checklist-like use of the framework.

Key and reflection factors – Participants noted that the difference in factor types allows them to prioritize the assessment. The key factors can be discussed during a preliminary assessment, while the reflection factors are more relevant in further conversations. Multiple participants decided to primarily focus on the key factors, such as EV2: "*We have some reflection, I think well that's relevant later on. The blue [key] factors are my primary drivers, so that's what I need to focus on*" and EV5: "*The reflection factors are more if you're heading towards implementation for example already*".

Framework objectivity – An issue that most participants faced was how the subjective nature of the framework hampered its operationality. The framework is currently more theoretically-centered around presenting the factors rather than being applicable for the practical use of assessing low-code BPM suitability. Therefore, EV5 noted: "*I think it is a good method to do an initial exploration of low-code BPM but to really determine if a customer case is suitable, I suspect a more in-depth analysis of the organization is needed*". As an example of the above, EV3 literally asked: "*So how does it finally come out, say, to a certain assessment of good or not good*".

In the end, operationality received a good 4.2 average from the participants. Having a structured, theoretically substantiated guiding model clearly aids the end users. Implementing an operational way of assessing a BPM initiative would greatly improve the framework in its operationality. One possible way of achieving this is by quantifying the factors and making them more measurable. This is further discussed in the following sub-chapter.

7.2.4 Ease of use

Lastly, the ease of use criterium gauges how the framework is experienced in use.

Time sensitivity – Each simulation lasted 30 minutes in total, in which the participants had to familiarize themselves with the use case, and the framework, and provide an assessment. Predominantly, the participants struggled to fully grasp the information in this timeframe. Consequently, some factors or other aspects were misinterpreted. After the assessment, EV2 shared: "*Having done it once already, I can also much more easily think of sub-questions. Now, sometimes I got stuck*". EV1 referred to the amount of information to comprehend: "*If you practice this a few times you will have the framework better in your mind and you can also work with it even better [...] during the conversation it is a little too 'wordy'*". Now, some participants went through the factors one-by-one asking a question. If the participants were given more time to fully comprehend the complete model, we suspect a more natural conversation could have flown.

Structured layers – The structuredness of the framework positively affected the ease of use. Not only the interpretation of key and reflection factors, but also the three 'global' themes composing the key factors helped. For EV2, it helped to structure the way of thinking: "*This layer definitely helped me keep the structure [...] Why do you want to work with it? What type of process and some IT principles helped with this layer*". EV5 mentioned it could help determine whom to ask certain questions: "*Those organization and goals, you might want to test those a little higher up [at the organization]. You could also use those process and technology [factors], for example, to do process selection together with people a little further at the operation*".

Factor quantifiability – As previously mentioned, quantification of factors could further elevate this framework as multiple participants asked how a missing factor or how the factor weight should be interpreted. For example, EV4 asked: "*Do all the factors weigh the same? […] Does it mean that, let's say, one out of 12 is missing, that you have to seriously consider what that means?*", and EV3 asked: "*There is no score assigned to the factors, if I read this correctly?*". This goes together with the aforementioned 'framework objectivity' where the current framework is highly interpretable.

In conclusion, ease of use scored a bit lower than operationality with a 3.9. Currently, this framework is more appropriate as a theoretical reference framework than an ad-hoc framework for instant use. To further increase the easiness of use, the factors need a more concise, objective, and possibly quantitative, approach.

7.3 Conclusion

Following the evaluation, we conclude the following:

- The framework represents a complete picture and can serve as a theoretical foundation behind an assessment of low-code BPM applicability. End users can apply it as a reference when analyzing the use case.
- End users that see the framework for the first time need ample time to fully comprehend the content of the framework. For these users, the framework is not suitable to ad-hoc use.
- The framework could be used to do an initial assessment of a BPM use case. However, for a thorough and a more in-depth assessment, the framework needs more objective, possibly quantitative, measures. This would greatly benefit the operability of the framework.

8 Discussion

The aim of this study is to create a framework that organizations can use to assess their BPM initiative on low-code BPM suitability. In the previous chapters, we have presented how the conceptual assessment framework has been constructed, validated, and evaluated for this aim. This chapter discusses these findings and relates these to earlier research. Lastly, we acknowledge several limitations of this study.

8.1 Implications

This thesis has two main implications for theory regarding low-code's application in BPM initiatives and for the evaluation of low-code suitability. Moreover, our framework provides a practical use for organizations seeking a BPMS solution to digitalize their business processes. The following sub-chapters describe these implications.

8.1.1 Low-code and BPM

Firstly, this study is one of the first to directly research low-code in the BPM context. To our knowledge, only Waszkowski (2019) presented a self-built BPMS through low-code and Cai et al. (2022) propose a method for small-scale process automation through the use of low-code. Both publications show the potential of applying low-code development for BPM purposes but do not analyze how low-code's specifics can be employed for BPM purposes, specifically to digitalize business processes. Other publications merely mention BPM as a possible application of low-code (Bock & Frank, 2021a). It is surprising that this is such an unresearched topic as Luo et al. (2020) found the BPM and process automation domains among the most used in low-code practice.

Our results help to understand when and how low-code can be applied in BPM initiatives to achieve its full potential. The evaluation showed that the framework succeeds in serving as a theoretical foundation describing low-code BPM through a broad array of factors. Completeness of the framework was rated the highest by participants throughout all four evaluation criteria. Although the participants are not low-code BPM experts, they have many years of experience in process automation and using innovative technologies for BPM purposes. Together with the fact that our framework is constructed on the basis of two existing theoretical frameworks, we argue that the resulting artifact is strongly substantiated theoretically and empirically. We encourage researchers to use and extend this scientific base when studying the low-code BPM proposition.

8.1.2 Low-code suitability assessment

Secondly, our study extends current knowledge on the effective use of low-code in organizations. Various studies have been performed to describe low-code and the potential applications of LCDPs (Sahay et al., 2020; Luo et al., 2021; Bock & Frank, 2021a; Frank et al., 2021). Moreover, studies have found the low-code adoption reasons and the organizational benefits and disadvantages (Sanchis et al., 2020; Alsaadi et al., 2021). But under what preconditions these benefits can be achieved, had not been researched (Frank et al., 2021). Our study presents these preconditions in the form of 18 factors focused on BPM initiatives aiming to digitalize business processes.

Interestingly, some disadvantages or reasons hampering the adoption of low-code identified in current literature do not correspond with empirical findings in this study. Low-code's scalability (Sahay et al., 2020; Bock & Frank, 2021a), security (Alsaadi et al., 2021; Hoogsteen & Borgman, 2022), and integration with the IT environment (Koplowitz, 2017) are cited as important considerations for adoption, but experts in the focus group and interviews disagreed. The consensus was that scalability concerns of LCDPs are negligible as long as the system architecture is well set up in the beginning. Security and compatibility with the IT environment are, contrarily, the advantages of low-code instead of disadvantages. Two reasons could possibly explain this discrepancy. Firstly, it seems possible that developments in low-code technology could have surpassed these technical issues since the previous publications. The low-code market is developing rapidly as more and more organizations adopt LCDPs (Vincent et al., 2020). To become market leaders, vendors continue developing their products to resolve issues that their customers face. Secondly, these issues could be less present when discussing low-code BPM specifically, instead of other uses of LCDPs. For example, the scalability of low-code BPM implies the size and throughput of the process that the platform can handle. Experts in this study argued that platforms offer enough scalability options to support very large processes. On the other hand, using LCDPs for application development could mean having to deal with large data throughput. Experts in this study agreed that high data transaction volumes could be a limitation in using LCDPs. This could explain the differences between our findings and literature.

8.1.3 Implications for practice

As mentioned, the aim of this study is to provide a practical artifact for organizations to use. Many decision-makers face difficulties in finding and selecting the right technologies to support their operations (Denner et al., 2018). Therefore, our findings are presented in a graphical framework, following the guidelines by Miles et al. (2014), that decision-makers can apply to their use case. The framework is supported by a written description elaborating on the framework's purpose, as can be found in Appendix H. Empirical validations and evaluations have been performed using potential end users of the framework. Based on the evaluations, we argue that the framework can be utilized by

organizations to do an initial assessment of low-code BPM suitability for their business processes. Moreover, implementation specialists and consultants can use the framework to prepare and validate a more thorough assessment of an organization's BPM case. There are, however, some limitations to our framework discussed in the following sub-chapter.

8.2 Limitations

Inevitably, there are several limitations of this study, both in our research and the final artifact. We address these in the following sub-chapter.

Primarily, the framework has never been tested in a real-life use case where an organization sought to digitalize its business processes. As our framework's main purpose is to help organizations in practice, such an evaluation would have provided invaluable input towards advancing the framework. It was, however, not possible to find a real-life use case during the timeframe of this study. Instead, we simulated such a scenario in the KPMG team where there was a possibility that the participants would have prior knowledge of the used use cases. The appropriate measures were taken, as seen in Chapter 3.1.5, but one participant admitted to recognizing what organization was behind the real-life case. We argue that this had a minimal impact since the evaluation focused on the framework's use, and not performance. However, we cannot say this for certain. All in all, an extensive real-life evaluation of the framework would have been ideal to evaluate the framework.

Additionally, the focus group and the fictitious use case simulations involved participants solely from within KPMG. On the one hand, these participants have high expertise levels in low-code development, low-code BPM, and other BPM technologies. On the other hand, we run the risk of selection bias by having an unrepresentative sample of our intended end users (Heckman, 1979). For the design and development, selection bias is less of a concern due to triangulation with other methods. This risk is the highest for the framework's evaluation as this was the sole method to evaluate the framework.

Apart from the limitations of our study, we also discuss various practical limitations of our low-code BPM assessment framework. First and foremost, this framework is focused solely on low-code BPM. Although some factors in the framework could be important in other LCDP applications, our framework cannot be generalized to represent low-code criteria in general. Moreover, the 'BPM initiative' cited in our research question referred to the digitalization of a business process through a BPMS. However, low-code BPM could also be used as a supplement to, or supplemented by, other technologies (Bock & Frank, 2021a). For such use cases, our framework would be less reliable.

Predominantly, the framework's evaluation has shown that the framework is not yet functional to perform a thorough assessment of an organization's use case. The framework is complete and can serve as a theoretical model for organizational decision-makers. However, it is not functioning in a way that

organizations can efficiently use it to do a complete assessment. For the latter, the framework needs to become less subjective and it would require a specific operative way to use it. Moreover, the framework's performance in correctly assessing a use case would need to be evaluated as well.

9 Conclusion

In times of digital transformation when organizations are urgently looking to digitalize their business processes, low-code BPM emerges as an effective solution. Low-code vendors advertise the easiness and potential benefits of their product while scientific literature nuances these benefits with concerns on customization, vendor lock-in, and complexity. In the meantime, organizations struggle to evaluate which solution is the most effective to advance their business processes. Existing literature does not provide guidance on assessing whether low-code BPM can be that solution. To solve this gap, this study proposes the low-code BPM assessment framework to help organizations assess the suitability of their BPM initiative to be supported by low-code BPM, through the following research question:

***RQ**: "How can organizations assess whether a low-code development platform is suitable to support their business process management initiative?"*

In order to answer the main research question, we have stated two sub-questions which have been answered throughout this study.

***SQ1**: "What characteristics of a business process management initiative are essential when assessing the suitability of low-code BPM?"*

In Chapters 4, 5, and 6 we present how in three iterations, including a literature review, a focus group, and expert interviews, we derived 18 factors. These factors represent characteristics of a BPM initiative to consider by organizations when assessing the suitability of low-code BPM. Each factor has a definition based on theoretical and empirical findings. Throughout the analysis of each factor, we found a division between key factors and reflection factors. The 12 key factors describe aspects of a BPM initiative that are ideal for low-code BPM, emphasizing their importance. The 6 reflection factors represent important considerations that organizations need to be aware of. However, these are not as important in decision-making as key factors. We have visualized all factors on a one-page, visual framework.

***SQ2**: "How does the created framework help organizations to assess the suitability of low-code BPM?"*

In Chapter 7, we evaluate how the framework performs on the pre-defined evaluation criteria of completeness, internal consistency, operationality, and ease of use. The evaluation showed that the framework provides a structured and comprehensive set of factors, that organizations can draw upon to conduct an initial assessment regarding low-code BPM for a use case. However, a lack of objective measurements and with the relations between the factors being ambiguous, there are points of improvement in terms of functionality.

Figure 5 presents our answer to the main research question. We hope that organizations can utilize the framework and this helps them to make confident decisions regarding low-code BPM.

9.1 Future research

There remains ample room for further research building on our study. We discuss three main paths and present various ideas that could inspire subsequent studies.

Framework reliability – The current framework could be evaluated more thoroughly to validate the factors and improve the framework's reliability. First of all, an evaluation of the framework conducted at a real-life organization facing a process digitalization problem would provide a more realistic setting than our fictitious use case simulations. Additionally, an evaluation focusing on other evaluation criteria, such as testing the performance of the framework, would illustrate its full capability. For example, it would be interesting to see a longitudinal study with real-life business cases. Lastly, a quantitative evaluation of the factors' relevance could provide further verification.

Framework development – The current framework can also be developed further through new features. The evaluation showed that a lack of objective measures made it difficult to do an in-depth assessment. Having more tacit, possibly quantitative, indicators for each factor can help end users derive stronger conclusions when using this framework. Moreover, the relations between various factors, and the similarity of some, made it more difficult to comprehend the framework. If the relations between the factors could be clarified, or different factors could be combined simplifying the artifact, the framework could become more effective. In the future, we also see the potential for a tool that incorporates this framework and provides a score of low-code BPM suitability.

Low-code assessment – This framework focused solely on applying low-code for BPM use cases. However, as was highlighted by Frank et al. (2021), similar frameworks should also be developed for other applications of LCDPs. Moreover, the use of LCDPs in combination with other technologies is another area for future research. Lastly, as one of the evaluation participants mentioned, similar approaches to our study could be applied to other process automation or orchestration technologies, such as chatbots or process mining.

As can be seen, there are enough ways to further advance this research. Our study is a first step to comprehend and recognize when low-code can advance organizations further. Hereby, we have focused on the digitalization of business processes. We encourage researchers to use and extend this knowledge to further develop low-code understanding.

References

Ackx, S. (2014). Emerging technologies, disrupt or be disrupted. In: *ISSE 2014 securing electronic business processes* (pp. 177-187).

Adrian, B., Hinrichsen, S., & Nikolenko, A. (2020). App development via low-code programming as part of modern industrial engineering education. In: *International Conference on Applied Human Factors and Ergonomics* (pp. 45-51). Springer, Cham.

Aier, S., & Fischer, C. (2011). Criteria of progress for information systems design theories. *Information Systems and E-Business Management*, *9*(1), 133-172.

Alsaadi, H. A., Radain, D. T., Alzahrani, M. M., Alshammari, W. F., Alahmadi, D., & Fakieh, B. (2021). Factors that affect the utilization of low-code development platforms: survey study. *Revista Română de Informatică Și Automatică, 31*(3), 123–140.

Aranda, J., Damian, D., & Borici, A. (2012). Transition to model-driven engineering. In: *International Conference on Model Driven Engineering Languages and Systems* (pp. 692-708). Berlin, Germany: Springer.

Bach, M. P., Vukšić, V. B., & Vugec, D. S. (2017). Individual's resistance regarding BPM initiative: case study of the insurance company. *Naše gospodarstvo/Our economy, 63*(4), 29-39.

Bock, A. C., & Frank, U. (2021a). Low-code platform. *Business & Information Systems Engineering*, *63*(6), 733-740.

Bock, A. C., & Frank, U. (2021b). In search of the essence of low-code: an exploratory study of seven development platforms. In: *2021 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)* (pp. 57-66). Fukuoka, Japan: IEEE.

Bosilj Vukšić, V., Brkić, L., & Tomičić Pupek, K. (2018). Understanding the success factors in adopting business process management software: Case studies. *Interdisciplinary Description of Complex Systems: INDECS*, *16*(2), 194-215.

Bowen, G. A. (2009). Document Analysis as a Qualitative Research Method. *Qualitative Research Journal, (9)*2, 7-40.

Bratincevic, J. (2020). When and How To Modernize Core Applications Using Low-Code Platforms. *Cambridge, MA*: *Forrester report*.

Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative research in psychology*, *3*(2), 77-101.

Brkić, L., Tomičić Pupek, K., & Bosilj Vukšić, V. (2020). A Framework for BPM Software Selection in Relation to Digital Transformation Drivers. *Tehnički vjesnik, 27*(4), 1108-1114.

Brown, C. V. (1997). Examining the Emergence of Hybrid IS Governance Solutions: Evidence from a Single Case Site. *Information Systems Research*, *8*(1), 69–94.

Bucchiarone, A., Cabot, J., Paige, R. F., & Pierantonio, A. (2020). Grand challenges in model-driven engineering: an analysis of the state of the research. *Software and Systems Modeling, 19*(1), 5-13.

Cabot, J. (2020). Positioning of the low-code movement within the field of model-driven engineering. In: *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings* (pp. 1-3). New York, NY: Association for Computing Machinery.

Cabot, J., Clarisó, R., Brambilla, M., & Gérard, S. (2017). Cognifying model-driven software engineering. In *Federation of international conferences on software technologies: applications and foundations* (pp. 154-160). Cham, Switzerland: Springer.

Cai, F. Z., Huang, S. Y., Kessler, T. S., & Fottner, F. J. (2022). A Case Study: Digitalization of Business Processes of SMEs with Low-Code Method. *IFAC-PapersOnLine, 55*(10), 1840-1845.

Cicman, J., Bratincevic, J. & Rymer, J., (2021). Forget About Build Versus Buy; Your Choice Is Customize Or Compose. *Cambridge, MA*: *Forrester report*. Retrieved from https://www.forrester.com/.

Crabtree, B. & Miller, W. (1999). A template approach to text analysis: Developing and using codebooks. In: B. Crabtree & W. Miller (Eds.) *Doing qualitative research* (pp. 163–177). Newbury Park, CA: SAGE Publications.

de Haes, S., & van Grembergen, W. (2009). An exploratory study into IT governance implementations and its impact on business/IT alignment. *Information Systems Management, 26*(2), 123-137.

Denner, M. S., Püschel, L. C., & Röglinger, M. (2018). How to exploit the digitalization potential of business processes. *Business & Information Systems Engineering, 60*(4), 331-349.

Dumas, M., Rosa, M. L., Mendling, J., & Reijers, H. A. (2018). *Fundamentals of Business Process Management*. Berlin, Germany: Springer.

Frank, U., Maier, P., & Bock, A. (2021). Low code platforms: Promises, concepts and prospects. A comparative study of ten systems. *ICB-Research Report*, No. 70.

Heckman, J. J. (1979). Sample selection bias as a specification error. In: *Econometrica: Journal of the econometric society* (pp. 153-161).

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS quarterly*, *28*(1), 75-105.

Hoogsteen, D., & Borgman, H. P. (2022). Empower the Workforce, Empower the Company? Citizen Development Adoption. In: *Proceedings of the 55th Hawaii International Conference on System Sciences* (pp. 1-10).

Hsu, H. Y., Liu, F. H., Tsou, H. T., & Chen, L. J. (2019). Openness of technology adoption, top management support and service innovation: a social innovation perspective. *Journal of Business & Industrial Marketing*, *34*(3), 575-590

Hutchinson, J., Whittle, J., & Rouncefield, M. (2014). Model-driven engineering practices in industry: Social, organizational and managerial factors that lead to success or failure. *Science of Computer Programming*, *89*, 144–161.

Iho, S., Krejci, D., & Missonier, S. (2021). Supporting Knowledge Integration with Low-Code Development Platforms. In: *ECIS 2021 Research Papers*, 38.

Imgrund, F., Fischer, M., Janiesch, C., & Winkelmann, A. (2018). Approaching digitalization with business process management. In: *Proceedings of the MKWI*, (pp. 1725-1736).

Jacob, S. A. & Furgerson, S. P. (2012). Writing interview protocols and conducting interviews: Tips for students new to the field of qualitative research. *The Qualitative Report, 17*(42), 1-10.

Kent, S. (2002). Model Driven Engineering. In: M. Butler, L. Petre, and K. Sere (Eds.) *Third International Conference on Integrated Formal Methods (IFM 2002)* (pp. 286–298). Berlin, Germany: Springer.

Konopik, J., Jahn, C., Schuster, T., Hoßbach, N., & Pflaum, A. (2022). Mastering the digital transformation through organizational capabilities: A conceptual framework. *Digital Business*, *2*(2), 100019.

Koplowitz, J. R. (2017). Vendor Landscape: BPM Platforms For Digital Automation. *Cambridge, MA*: *Forrester report*. Retrieved from https://www.forrester.com/.

KPMG (2021). *Absorbing change with Low-Code BPM* [Whitepaper]. Retrieved from https://home.kpmg/nl/nl/home/insights/2021/02/absorbing-change-with-low-code-bpm.html.

Kraus, S., Durst, S., Ferreira, J. J., Veiga, P., Kailer, N., & Weinmann, A. (2022). Digital transformation in business and management research: An overview of the current status quo. *International Journal of Information Management, 63*, 102466.

Krueger, R. A., Casey, M. A. (2014). *Focus Groups – A Practical Guide for Applied Research*. Thousand Oaks, CA: SAGE Publications.

Leidecker, J. K., & Bruno, A. V. (1984). Identifying and using critical success factors. *Long range planning, 17*(1), 23-32.

Lin, L. M., & Hsia, T. L. (2011). Core capabilities for practitioners in achieving e-business innovation. *Computers in Human Behavior, 27*(5), 1884-1891.

Luftman, J., & Brier, T. (1999). Achieving and sustaining business-IT alignment. *California management review, 42*(1), 109-122.

Luo, Y., Liang, P., Wang, C., Shahin, M., & Zhan, J. (2021). Characteristics and Challenges of Low-Code Development. In: *Proceedings of the 15th ACM / IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)* (pp. 1-11).

Mamudu, A., Bandara, W., Wynn, M. T., & Leemans, S. J. (2022). A Process Mining Success Factors Model. In: *International Conference on Business Process Management* (pp. 143-160). Springer, Cham.

March, S. T., & Smith, G. F. (1995). Design and natural science research on information technology. *Decision support systems, 15*(4), 251-266.

McManus, D. J. (2003). A Model of Organizational Innovation: Build versus buy in the decision stage. *The International Journal of Applied Management and Technology*, *1*(1), 29-44.

Metrôlho, J. C. M. M., Araújo, R., Ribeiro, F., & Castela, N. (2019). An approach using a low-code platform for retraining professionals to ICT. In: *Proceedings of EDULEARN* (pp. 7200-7207).

Meuser, M., and Nagel, U. (2009). The expert interview and changes in knowledge production. In *Interviewing experts* (pp. 17-42). Palgrave Macmillan, London.

Miles, M. B., Huberman, A. M. & Saldaña, J. (2014). *Qualitative Data Analysis – A Methods Sourcebook* (4th ed.). Thousand Oaks, CA: SAGE Publications.

Mohagheghi, P., Dehlen, V. (2008). Where Is the Proof? - A Review of Experiences from Applying MDE in Industry. In: I. Schieferdecker, A. Hartman (Eds.) *Model Driven Architecture – Foundations and Applications* (pp. 432-443). Berlin, Germany: Springer.

Moran-Ellis, J., Alexander, V. D., Cronin, A., Dickinson, M., Fielding, J., Sleney, J., & Thomas, H. (2006). Triangulation and integration: processes, claims and implications. *Qualitative research, 6*(1), 45-59.

Olariu, C., Gogan, M., Rennung, F. (2016). Switching the Center of Software Development from IT to Business Experts Using Intelligent Business Process Management Suites. In: V. Balas, L. Jain, B. Kovačević (Eds.) *Soft Computing Applications. Advances in Intelligent Systems and Computing*, 357. Cham, Switzerland: Springer.

OutSystems (2019). The State of Application Development - Is IT Ready for Disruption? [report]. *Boston, MA: OutSystems*. Retrieved from https://www.outsystems.com/.

Patel, K., & McCarthy, M. P. (2000). Digital transformation: the essentials of e-business leadership. *McGraw-Hill Professional*.

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of management information systems, 24*(3), 45-77.

Plattfaut, R., Borghoff, V., Godefroid, M., Koch, J., Trampler, M., & Coners, A. (2022). The Critical Success Factors for Robotic Process Automation. *Computers in Industry, 138*, 103646.

Prat, N., Comyn-Wattiau, I., & Akoka, J. (2015). A taxonomy of evaluation methods for information systems artifacts. *Journal of Management Information Systems, 32*(3), 229-267.

Ravasan, A. Z., Rouhani, S., & Hamidi, H. (2014). A Practical Framework for Business Process Management Suites Selection Using Fuzzy TOPSIS Approach. In: *Proceedings of the 16th International Conference on Enterprise Information Systems* (pp. 295-302).

Ravesteyn, P., & Batenburg, R. (2010). Surveying the critical success factors of BPM-systems implementation. *Business Process Management Journal*, *16*(3), 492–507.

Richardson, C. & Rymer, J. R. (2014). New development platforms emerge for customer-facing applications. *Cambridge, MA*: *Forrester report*.

Sahay, A., Indamutsa, A., Di Ruscio, D., & Pierantonio, A. (2020). Supporting the understanding and comparison of low-code development platforms. In: *2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)* (pp. 171-178). Portorož, Slovenia: IEEE.

Sanchis, R., García-Perales, S., Fraile, F., & Poler, R. (2020). Low-Code as Enabler of Digital Transformation in Manufacturing Industry. *Applied Sciences, 10*(1), 12.

Staron, M. (2006). Adopting Model Driven Software Development in Industry – A Case Study at Two Companies. In: O. Nierstrasz, J. Whittle, D. Harel, G. Reggio (Eds.) *Model Driven Engineering Languages and Systems* (pp. 57-72). Berlin, Germany: Springer.

Štemberger, M. I., Bosilj-Vukšić, V., & Jaklić, M. I. (2009). Business process management software selection–two case studies. *Economic research-Ekonomska istraživanja, 22*(4), 84-99.

Trkman, P. (2010). The critical success factors of business process management. *International journal of information management, 30*(2), 125-134.

Van Veldhoven, Z., & Vanthienen, J. (2022). Digital transformation as an interaction-driven perspective between business, society, and technology. *Electronic Markets, 32*(2), 629-644.

Venable, J., Pries-Heje, J., & Baskerville, R. (2016). FEDS: a framework for evaluation in design science research. *European journal of information systems, 25*(1), 77-89.

Vincent, P., Natis, Y., Kimihiko, I., Wong, J., Ray, S., Jain, A. & Leow, A. (2020). Magic quadrant for enterprise low-code application platforms. *Stamford, CT: Gartner Report*. Retrieved from https://www.gartner.com/.

Vom Brocke, J. & Schmiedel, T., (2014). Ten principles of good business process management. *Business process management journal, 20*(4), 530-548.

vom Brocke, J., Zelt, S., & Schmiedel, T. (2016). On the role of context in business process management. *International Journal of Information Management, 36*(3), 486-495.

Waszkowski, R. (2019). Low-code platform for automating business processes in manufacturing. *IFAC-PapersOnLine, 52*(10), 376–381.

Whittle, J., Hutchinson, J., Rouncefield, M., Burden, H., & Heldal, R. (2017). A taxonomy of tool-related issues affecting the adoption of model-driven engineering. *Software & Systems Modeling, 16*(2), 313-331.

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., & Wesslén, A. (2012). *Experimentation in software engineering*. Berlin, Germany: Springer.

Wong, J., Iijima, K., Leow, A., Jain, A. & Vincent, P. (2021). Magic quadrant for enterprise low-code application platforms. *Stamford, CT: Gartner Report*. Retrieved from https://www.gartner.com/.

Woo, M. (2020). The rise of no/low code software development—No experience needed?. *Engineering (Beijing, China)*, *6*(9), 960.

Xu, L. D., Xu, E. L., & Li, L. (2018). Industry 4.0: state of the art and future trends. *International journal of production research*, *56*(8), 2941-2962.

Yin, R. K. (2009). Designing case studies: Identifying your case(s) and establishing the logic of your case study. In: R. K. Yin. (Eds.) *Case study research: Design and methods* (pp. 25–65). Thousand Oaks, CA: SAGE Publications.

Appendix

A – Exploratory interview protocol

The interview protocol is based on the guidelines by Jacob and Furgerson (2012).

Goal: The primary goal is to gather practical data on BPM initiative factors important for LCDP implementation. A secondary goal is to have confirmation of the identified organizational problem (assessment of business case suitability towards LCDPs).

Approach: Semi-structured introduction, unstructured overall interview. The questions are primarily guided by the identified 6 dimensions. For the rest, any interesting information is followed upon.

Interviewee: 'Digital transformation'/'Citizen-development' Consultant at KPMG.

Amount of time: 45 minutes.

INTRODUCTION

Start with

- An introduction of myself and my role in KPMG
- An overall explanation of my research and the core research question.
- Explanation of the goal of this interview: "To have a first practical view on how low-code development platforms are applied, what their benefits are, and what important factors differentiate low-code from other platforms."

Ask the interviewee whether the interview can be recorded and excerpts can be used in this research.

Remind the interviewee to sign the **informed consent** form and explain that at any moment, they could stop the interview.

Then, define the term 'LCDP', and 'low-code BPM' as understood in this research.

- Low-code development platforms: "A platform with modeling and visual tools where the user interface, business logic, workflow and data handling can be constructed through easy-to-use component assembly and through modeling"
- Low-code BPM: "Low-code BPM supports rapid application development for end-to-end case management, with the workflow model and engine at its core, enabling process automation, integration, monitoring and enhancement with intelligent automation technology through easy-to-use component assembly and modeling"

Continue with introductory questions:

1. What is your position in <company> and for how long?
2. What is your experience with low-code tooling?

3. How much experience have you had, and in what roles?
4. For which end-goals have you used low-code tooling?
5. Have you used low-code tooling in the context of BPM initiatives?

OVERALL OPEN-ENDED QUESTIONS

Depending on the answer to question 5, the questions can be either for low-code in general, or for low-code in BPM. For example, in no specific order:

- Can you tell me more about the organizations/businesses that adopt LCDPs? (e.g. culturally, size, industry, etc.)
    - Could every type of organization create applications with low-code technology?
- Can you tell me more about the users acquiring LCDPs? (e.g. innovative, technical, etc.)
    - Could every type of worker create applications with low-code technology?
- What kind of end-results have you seen from the adoption of LCDPs? (e.g. applications, process automations)
- Have you seen a failed LCDP adoption and what were the reasons?
- What are the main benefits of LCDP? What are the main risks of LCDPs?
- What kind of consequences may come from adopting LCPDs? (e.g. organizational change)
- Do you believe the low-code solution is one for the future? Why yes/not?

If acquainted with Low-Code BPM, most questions can be rephrased through "Low-code BPM". Additional questions:

- Can you tell me what kind of processes are suitable for LCDP support?
- Can you tell me what kind of processes you have seen being supported by LCDPs?

Ending

6. Are there any things you would like to share before ending this interview?

If not, thank the interviewee for their time and answers to your questions. Then:

7. In the future I am interested to hear from other experts in the field of low-code for BPM. Do you have any contacts that may be interested?
8. Do you have any questions regarding this interview or my study?
9. I will transcribe this interview and process several points that you mentioned in my assessment framework. If you would like to receive a copy before publication of the framework, please let me know. I will share a copy once the report has been written!

"Thank you again for your time!"

B – Focus group protocol

Purpose: We want to find what elicited framework factors have empirical and practical backing, and what other, new factors may play a role in the decision-making process when considering Low-code-BPM support.

Approach: In a hybrid online/in-person setting, we collect multiple low-code (BPM) consultants and familiarize them with the research and current framework. Then, we want to incentivize a conversation and discussion between the consultants/managers, based on their practical experience, about the question: "When is a solution built through low-code BPM the best option?". The group session will be recorded and moderated to steer it inside the scope of our research.

Practicalities: 1,5 hours, online/in-person session

Preparation:

- Ask participants to think about (1) what low-code (BPM) implementations they have done in the past, (2) what the deciding factors were in choosing for low-code, and (3) what the ideal business case is for a low-code (BPM) implementation
- Send out explanation and consent form, print out artifacts, book and prepare the room

Participants: 3 / 4 KPMG consultants experienced with low-code implementations

PROTOCOL

Focus Group Introduction (5 minutes) - with slides

- Introduce myself
- Ask for brief introduction of each participant: name, role and experience regarding low-code (BPM)
- Explain the study aims + the research question: "*How can organizations assess the suitability of their business process management initiative to be supported by a low-code development platform?*"
- Explain the current state of the research: "*Now, it's based on literature. This is the first empirical evaluation.*"
- State the goal of session:
    - "*To openly discuss the factors important when considering low-code BPM, by evaluating the existing framework"*
    - "*The usability of the framework is up for later evaluations.*"
- Explain role of the participants: "*You were selected due to your experience with the technology and the low-code offering. With your varying backgrounds in low-code, we hope to elicit a broad spectrum of viewpoints on what factors make the difference when opting for low-code solutions.*
- Elaborate upon openness: "*We incentivize you to speak up, complement, or disagree with each other, and also with me. Any information is valuable and I will not be offended if my current*

conclusions appear to be totally incorrect. If so, please bring it up and we will see where it ends up."

- Ask for consent for audio and video recording. Explain anonymity and privacy concerns.

Framework Introduction (10 minutes) - with printed out framework & definitions

- Explain what the (6) dimensions are and what a factor is.
- Explain the 3 types of factors.
- Explain our 'definition of low-code BPM' and explanation of the framework structure
- Provide printed documents and a red, orange, and blue pen. Ask participants to think about the factors and make notes in (and add experiences):
    - Red: a wrong definition, or wrong application of the factor towards low-code BPM, or the factor does not play a role.
    - Orange: a comment/alteration to a definition, or the application towards low-code BPM can be refined. Subjects can also add comments or personal experiences in orange.
    - Blue: any additional factor that has not been mentioned regarding LCDPs for BPM

Visualize framework comments (2 minutes) – with large print out of framework in A2

- Ask each participant to add a red or orange pin on the print out, visualizing all the comment themes for the upcoming discussion.

[TO TEAMS FORMAT] - Solutions

- Use Miro to project the framework (see Appendix F for materials). Ask participants to mark parts of the framework with a pin (virtually)

FOCUS GROUP

Start with a dimension and ask what discussion points people had regarding a factor. Then incentivize a discussion, with possible questions:

Technical dimension (exp. 10 minutes) - Example questions to engage the discussion

- "Why do developers hesitate when thinking about low-code adoption?"
- "What technical aspects have motivated/prohibited organizations to adopt low-code development?
- "What organizational IT strategy would benefit low-code adoption?"

Process dimension (expected 10 minutes) - Example questions to engage the discussion

- "What process characteristics are the first that you consider when thinking about a low-code BPM implementation?"
- "What do you think is an ideal type of process to automate with low-code BPM?"

Goal dimension (exp. 15 minutes) - Example questions to engage the discussion

- "What BPM initiative aspects have been a red flag when talking about low-code BPM?"
- "What long-term effects of low-code (BPM) adoption have you seen?
- "What is the first high-level goal that comes to mind that low-code BPM can achieve?"

-- BREAK --

Social dimension (exp. 20 minutes) - Example questions to engage the discussion

- "Can every development department adopt low-code BPM?"
- "What have middle managers/developers said to you about what (de)motivated them to adopt low-code BPM?"
- "What difficulties have you encountered when introducing low-code to employees and middle managers? "
- "How would you describe a development department that adopts low-code technology?"

Organizational dimension (exp. 20 minutes) - Example questions to engage the discussion

- "Can every organization adopt low-code BPM?"
- "What difficulties have you encountered when introducing low-code to top management?"

Environmental dimension (exp. 10 minutes) - Example questions to engage the discussion

- "In what industries has low-code been adopted in your experience?"
- "Can an organization adopt low-code BPM successfully on its own?"

General discussion incentivizing questions during the session:

- Can you give a practical example (of your point)?
- As a group decide on a rating from 1 to 5 on importance of each factor.
- What are the barriers that organizations face (regarding dimension/factor)?
- What is the first reason you think about when an organization rejects a low-code implementation (regarding dimension)?
- When do you think: "this needs a low-code BPM solution"?
- How did an organization overcome this (factor/issue)?

ENDING (10 MINUTES):

Summarizing group decision 'games':

- Give a personal top 5 and bottom 5 factors (at the end of the session)

Finalizing the session:

- Thanking for participation
- Summarizing with a few surprising notes
- Explaining that the research will continue into further expert interviews:
    - "*Do you have any contacts that might be happy to share their knowledge on this topic?*"
- End.

C – Expert interviews protocol

Purpose: We want to find more empirical and practical evidence regarding elicited and possible new factors in the framework, as the participants have different perspectives regarding the IT decision-making process.

Approach: Semi-structured, audio (and maybe) video recorded

Practicalities: Online or offline (depending on the possibilities).

Preparation: Ask participants to think about (1) what low-code (BPM) implementations they have done in the past, (2) what the deciding factors were in choosing for low-code, and (3) what the ideal business case is for a low-code (BPM) implementation

Participants: 8, with different perspectives

PROTOCOL

Interview Introduction (5/10 minutes)

- Introduce myself
- Explain the study + the research question: "*How can organizations assess the suitability of their business process management initiative to be supported by a low-code development platform?*"
    - Explain our 'definition of low-code BPM'
- Explain the current state of the research: "*The framework is based on literature and an evaluation by a panel of low-code experts. However, the framework is still being refined.*"
- State the goal of session: "*The goal of this evaluation is to find more empirical and practical evidence regarding elicited and possible new factors in the framework.*"
- *Explain approach*:
    - "First, I would like to hear about your experiences with low-code development and I'll probably have some follow-up questions."
    - "Secondly, I will present the framework, consisting of dimensions, factors and factor types, and explain how it attempts to aid the decision-making process. It will be an introduction but let me know your opinion. Then, we will discuss your thoughts on the contents of the framework, but also the easiness of use!"
- Ask for consent for audio and video recording. Explain anonymity and privacy concerns.

Interviewee introduction (10/15 minutes)

- Brief introduction of the participant:
    - Name
    - Company's services
    - Role in company
    - Experience regarding low-code (BPM) development / What platforms have you worked with?

- If with a low-code (BPM) end user:
    - Brief explanation of system(s) being developed using low-code?
    - How was the decision made to start developing using low-code? Based on what?
    - What motivated and demotivated this decision-making process?
    - What met, or hasn't met, your expectations since?
- If with a Low-code (BPM) implementation specialists / consultants:
    - What are the main reasons that clients adopt low-code (BPM) development?
    - How do you decide if low-code development is an applicable fit?
    - What has caused the biggest struggles when implementing low-code development in organizations?

Discuss the framework contents (20 minutes):

- Present the framework
    - What the dimensions are and their definitions
    - What the factor definitions are and its relation to decision-making
- Follow-up questions:
    - "*Do you see how these dimensions are intercorrelated and cover the whole question domain?*" "*Are there any unclarities?*"
    - "*Do you understand what is measured with each factor?*"
    - "*Can you think of a practical example concerning each factor?*"

Goal dimension – sample questions

- "*What should the organization aspire to achieve with low-code development?*"
- "*What is the main aim (or top 3) that a low-code BPM solution is going to achieve for an organization?*"

Process dimension – sample questions

- "*Can each business process be supported by a low-code BPM solution?*"
- "*What kind of processes can better be tackled with other solutions?*"

Technical dimension – sample questions

- "*What technical concerns have hampered low-code BPM adoption?*" / "*What has low-code BPM adoption achieved technically in your/an organization?*"

Social dimension – sample questions

- "*Can each team/department adopt a low-code BPM solution?*" / "*What would be your dream- or nightmare-team?*"
- "*What have employees said, or how have they reacted, to (possible) adoption of low-code BPM in their organization?*"

Organizational dimension – sample questions

- "*What had to change in your/an organization before low-code BPM could get adopted?*" / "*What changed after adopting?*"
- "*Can you name examples of organizational types that would struggle with low-code BPM adoption?*"

Environmental dimension – sample questions

- "*In what sense does the industry of an organization affect the possibilities of low-code BPM?*"
- "*Could an organization start with low-code on its own? What 'outside support' would it need?*"

Discuss the framework in its use (10 minutes) - sample questions:

- "*What tools do you use to come to a decision regarding an IT solution?*"
- "*In what sense would this fit in your toolkit for such decision making? Would this be useful? Compared to other solution?*"
- "*Being critical, what needs to be improved for this framework to make it easy to use?*"
- "*Would you feel this would make you more confident regarding adopting a low-code tool, or not?*"

Finalizing the session:

- Thanking for participation
- Summarizing with a few surprising notes
- Explaining that the research will continue into further expert interviews:
    - "*Do you have any contacts that might be happy to share their knowledge on this topic?*"
- End.

D – Fictitious use case simulations protocol

Purpose: We want to see how the final framework performs in its use according to the earlier defined evaluation criteria: completeness, internal consistency, operationality, and ease of use.

Approach: Each simulation starts with a participant and the researcher. The participant receives the use case description and the low-code BPM assessment framework. After familiarizing themselves with the materials, the use case expert joins the room. The simulation commences and the participant is allowed to ask questions to the use case expert to further expand knowledge on the use case. After 15 minutes, the assessment concludes and the participant provides his verbal assessment of the use case. Thereafter, the participant is briefly interviewed and asked to fill in a survey

Practicalities: Online or offline (depending on the possibilities).

Preparation: Ask participants to read the information form about this session

Participants: 5 participants, 3 use case experts (where each simulation includes 1 participant and 1 use case expert)

PROTOCOL

1. **Participant + Researcher** (2 minutes): Go through introduction letter, explain timetable (15 minutes for use case and framework & 15 minutes for questions & reflection), and sign Consent Form
2. **Participant + Researcher** (4 minutes): Introduction of Use Case description.
    - START RECORDING -
3. **Participant + Researcher** (9 minutes): Framework explanation
    - Only explain what framework can be used for
    - Let participants read through definitions themselves
    - Possibility to ask questions about the framework.
    - INVITE EXPERT -
4. **Everyone** (15 minutes): Expert questions moment
5. **Everyone** (2 minutes): Advice and explanation
6. **Participant + Researcher** (3 minutes): Reflection 'framework in use'. Example questions
    - Did you find the framework easy to understand?
    - In what ways did the framework help you make an assessment, or in what ways was it difficult to use?
    - How did you find the framework's ease of use?

SURVEY

1. I could effectively use the framework to assess the suitability of low-code BPM in my current or future projects (*1 - Ineffective, 5 - Effective*)
2. I would be able to effectively explain the key features of low-code BPM using the framework (*1 - Not capable, 5 - Very capable*)
3. The framework is easy for me or my (potential) customers to use to assess the suitability of low-code BPM (*1 - Difficult, 5 - Easy*)
4. The framework provides an easy-to-understand overview of the key features of low-code BPM (*1 - Difficult, 5 - Easy*)
5. The framework provides a complete picture of the factors I would consider when assessing the suitability of low-code BPM (*1 - Incomplete picture, 5 - Complete picture*)
6. The framework represents all the main themes I would consider when assessing the suitability of low-code BPM (*1 - Incomplete, 5 - Complete*)
7. The factors in the framework, and the relationships between the factors, are clear to me (*1 - Unclear, 5 - Clear*)
8. The factors, their definitions and the types of factors complement or contradict each other (*1 - Contradictory, 5 - Complementary*)

E – Codebooks

The codebooks represent the initial coding performed after analyzing the dataset. Following the methodology by Braun and Clarke (2006), the results following 'Phase 2: Generating initial codes' are presented.

LITERATURE REVIEW

Readers might find Table 3 more clear as opposed to the processed version of the literature review codebook seen below.

Code	Definition	Example
Time	LCDPs allow to develop quickly	"[LCDPs are able] to develop new digital solutions much faster than their competitors"
Process Excellence	Low-code BPM achieves quality improvements	"Minimization of inconsistencies/errors"
Organizational Commitment	The organization should commit on the long-term	"Organizations which structure their work on a strict project-by-project basis may find it much more difficult to successfully trial MDE"
Complexity	Low-code BPM remains complex	"Building business applications with LCDPs is still a complex process"
Automation	Low-code BPM is suitable for process automation	"LCDPs favored in automated processes and workflows"
Scalability	LCDPs are not suitable to broad scale	"LCDPs are designed mainly to address small apps"
Value-adding processes	Low-code BPM should be employed for specialized processes	"These applications automate an enterprise's most important business process(es) — those that define its purpose"
Process flexibility	Low-code BPM offers flexibility and can automate various processes	"Diversity of organizational processes requires a customized approach toward change"
Costs	Low-code can reduce costs	"[LCDPs have] lower IT costs"
Digital transformation	Low-code is a tool during digital transformation	"LCDPS increase digital transformation"
IT Governance	Low-code requires a level of IT governance for it to be successful	"Citizen development is most suitable for a centralized IT Governance structure at portfolio level"
Top management decision	The decision to adopt low-code often comes from top management	"Since technological innovations are often the result of top-down decision-making, management automatically plays a crucial role"
Business-IT alignment	Low-code requires proper business-IT alignment in the organization	"To adopt organization-wide, successful alignment between the business and IT is needed"
Responsiveness	Low-code BPM allows organizations to be more responsive	"Increasing responsiveness to business demands"
Innovative	An innovative culture is more suitable towards low-code	"Organizations in which frequent innovation is considered desirable and important have a more positive attitude toward citizen development"
Third-party support	Employing a partner could be beneficial	"External organizations that specialize in LCDPs are often more aware of the less-visible elements that impact citizen development adoption and success"

Vendor lock-in	There is a risk of vendor lock-in	"There is a lack of standards in this domain by hampering the development and collaboration among different engineers and developers"
Competitiveness	Low-code is used in competitive environments	"Motivation comes from need (responsiveness, competitiveness) rather than experimentation"
Platform capability	The LCDP should be assessed for its ability	"If all requirements can be satisfied within the framework of a LCP"
Compatibility	LCDPs can struggle with compatibility	"Problems integrating apps developed by LCDP with other systems"
Security	LCDPs can struggle with security	"Development outside of the IT domain is a real threat to data security and compliance issues and means that businesses are more likely to encounter cyberattacks"
Privacy	LCDPs can struggle with privacy	"Increasing information privacy due to internal development"
Trust	Trust is important for low-code	"Distrust of LCDP capability"
Citizen developer	Low-code might be too difficult for citizen development	"Some modelling aspects are still tricky for non-technical people"
Lack of understanding	Employees need to understand the technology	"Against BPM due to … fear of job loss … dissatisfaction with logic … conflicts and lack of communication"
Lack of willingness	Employees need to be willing to work with the technology	"Switching to the unknown is met with hesitance"
Reputation	The LCDP vendor's reputation is important to assess	"BPMS vendor's reputation and maturity are uniform and score the highest among categories of this dimension after BPMS adoption"

FOCUS GROUP

This codebook includes the transcription of the focus group, notes taken by the researchers, and the comments provided by the participants on the virtual board during the focus group.

Code	**Definition**	**Example**	**Amount**
Process orchestration	Organizations aim for process orchestration	"Most of the times, they say: We want to apply more process orchestration"	7
Long-term low code adoption	Low-code is not an ad-hoc, short-time solution	"If you want to use low-code on the long term method? Definitely yes"	3
Agility	Low-code improves agility	"The second point we frequently mention is agility"	2
Security	Low-code does not have security issues	"Well, it even is an advantage of using LCDPs"	7
Performance	LCDPs themselves don't struggle with performance	"It's more about whether your requirements fit the platform"	5
Extending systems	LCDPs can be employed to extend existing systems	"You have an off-the-shelf product that you want to customize"	5
Unifying systems	LCDPs can be employed over existing systems	"It's often used as a layer-on-top of existing systems"	4
Compatibility	LCDPs are easily compatible	"LCDPs have a lot of connection with existing standard systems"	1

Development process	LCDPs don't require a separate development process	"When they develop through low-code, they can follow the same method"	8
Business-IT alignment nuance	Low-code needs business-IT alignment	"I think it's always good for organization, a relationship between business and IT"	8
Centralized IT governance	Low-code development requires centralized IT governance	"You always need a framework"	6
Lack of existing solution	LCDPs can be employed when no existing solution exists	"What we see a lot, is when the market doesn't offer a solution"	8
Existing implementation partners	Low-code BPM needs to be done with implementation partners	"Especially with low-code, we see that partners are used"	6
Existing development knowledge	Low-code knowledge is required	"Of course you need proper low-code knowledge"	4
Front vs. Backoffice processes	Low-code BPM isn't specific to front- or back office	"We see it used more and more for front office, but still back office is most popular"	4
Responsive process	A process requires flexibility	"If a process needs flexibility"	2
Exceptional process	For a critical process	"They might use it for unique processes"	2
Broad processes	A large scale process	"If those are long, chained processes over multiple departments"	1

EXPERT INTERVIEWS

This codebook includes the transcriptions of the expert interviews and notes taken by the researchers.

Code	**Definition**	**Example**	**Amount**
Speed and Agility	Low-code's largest benefits are speed and agility	"The competitive edge is on speed and agility"	36
Process orchestration	Low-code BPM is meant for process orchestration	"The role of LCDPs is indeed to orchestrate or build an app on top of systems"	27
Long-term Low-code commitment	Low-code BPM should only be implemented on the long-term	"It only makes sense on the long-term"	17
Iterative development	Low-code development should work iteratively	"We are entering an Agile process"	12
Lower costs	Low-code saves costs	"Cost saving is often a reason to choose for low-code"	11
Data transparency	Low-code enables to have a more transparent picture of your data	"Data and the transparency and drill down capability"	6
Process transparency	Low-code enables to have a more transparent picture of your process	"You're forced to think again about how your process is structured"	6
Compliance	Low-code is often used in compliancy processes	"Around compliancy there is a lot to do"	3

User experience	Low-code BPM improves the user experience	“You’re able to customize the experience”	2
Unique (selling point) process	Low-code BPM should be used for unique processes	“you do it because of the uniqueness of your process”	14
Complex process	Low-code BPM should be used for complex processes	“Just the case of having multiple parties involved is suited for low-code BPM”	12
Varying process	Low-code BPM should be used for varying processes	“and being flexible, able to move along with changes”	10
Volume	Low-code BPM should be used for processes with high volumes	“very low level transactional, that’s probably not what I want to do [with Low-code BPM]“	7
User participation	Low-code BPM should be used where users are involved with the process	“the use cases that are suitable are processes where users are involved”	6
Unifying systems	Low-code BPM allows to connect systems	“often in combination with portals, where you need to connect applications, so application integration”	22
LCDP’s capability	The ability of a Low-code BPM platform should be assessed	“Performance could be a problem, but it’s always solvable”	21
Maintainability	Low-code BPM systems are easy to maintain	“It works better, it’s easier to maintain”	14
Extending systems	Low-code BPM allows to extend existing systems	“so extending the capability of the underlying system”	14
Security	Low-code BPM is a safe platform	“LCDPs are technically safer than other solutions”	9
Standardization	Low-code BPM improves standardization	“every team is doing the same, so you set it up in one place”	7
Compatibility	Low-code BPM is compatible with other systems	“I’ve never experienced that it wasn’t compatible	5
Limited functionality	Low-code BPM limits the possibilities	“you work in the scope of the LCDP”	5
Not for big data	Low-code BPM is not suitable for big data transactions	“It’s not a big data program”	5
External partners	Low-code BPM development can better be outsourced	“I would definitely advise to work with a lead designer or lead consultant”	18
Lack of suitable solution	Low-code BPM should be used when no suitable solution exists	“they chose for Low-code BPM because they didn’t find a solution”	11
Changing market	Low-code BPM should be used in competitive environments	“Everyone thinks he is working in a competitive environment”	8
Popularity in market	The right LCDP can be chosen by popularity	“We chose Mendix because there were a lot of insurers”	3
Available IT knowledge	It’s key to have employees with low-code experience	“My biggest concern: where do I find the right people for”	36
Intrinsically motivated employees	Motivated employees are needed to make Low-code BPM a success	“The intrinsic motivation should just be there”	9
Trust	Trust is needed before adopting Low-code BPM	“Trust is a much more important factor with low-code”	9

Innovation-driven environment	Low-code BPM should only be employed in an innovation-driven environment	"I don't think it is a concern, maybe a nice to have"	5
Governance	Low-code BPM requires centralized IT governance	"You need good governance, also concerning money"	19
Business-IT alignment	Low-code BPM requires good business-IT alignment	"Bad business-IT alignment is very bad breeding ground for low-code BPM"	18
IT strategy	Low-code BPM should be part of an IT strategy	"Is low-code a strategy that the organization wants to follow?"	5
Active top management support	Low-code BPM requires active top management support	"They have to understand why it is important, that it takes time"	3
Framework usability	Anything concerning the usability of the framework	"It's a very nice overview"	7

FICTITIOUS USE CASE SIMULATIONS

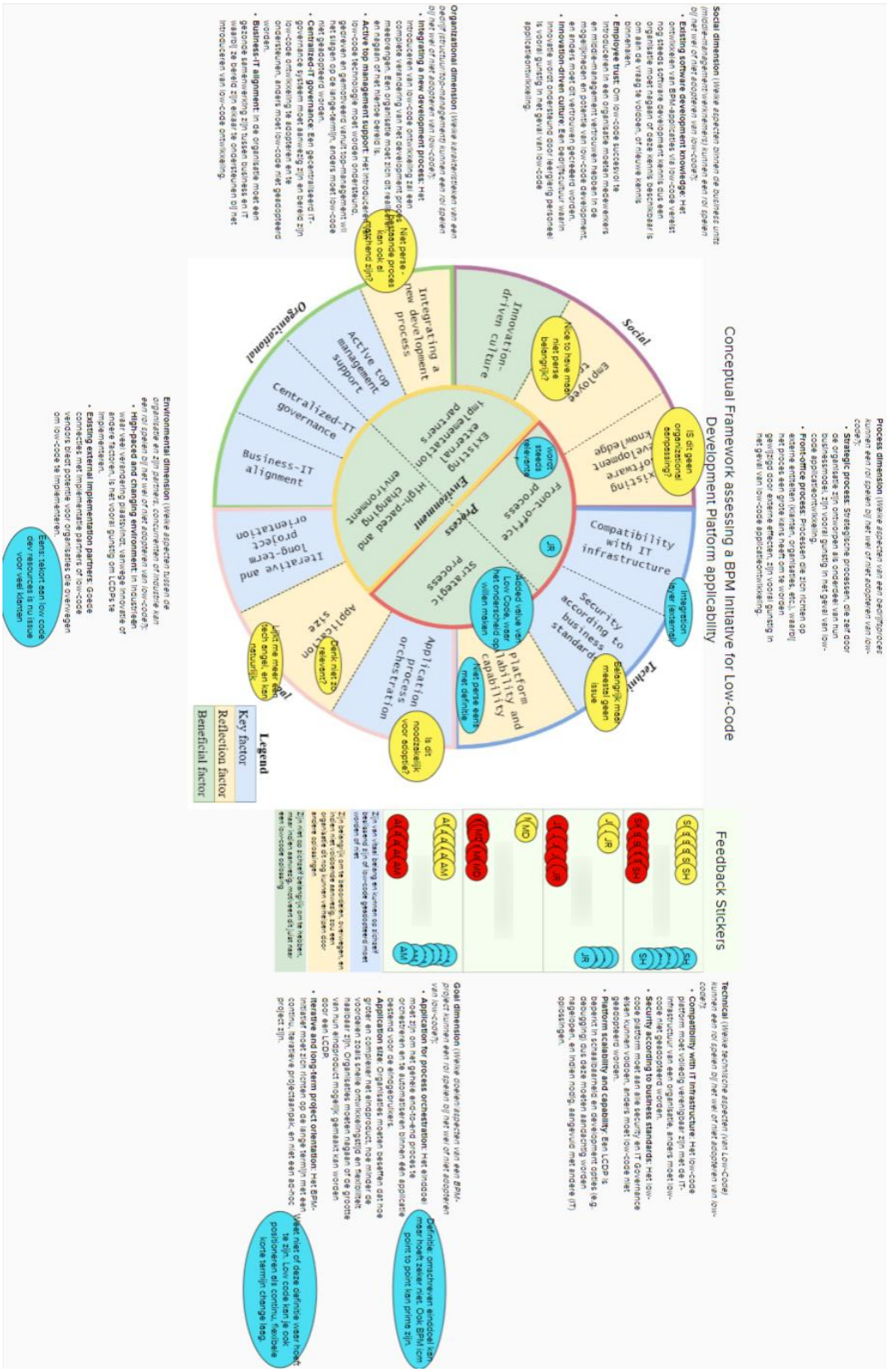
This codebook includes the transcriptions of the fictitious use case simulation, notes taken by the researchers and observations made during the simulations.

Code	Definition	Example	Amount
Key vs. Reflection	The difference between key- and reflection factors	"The reflection factors are more if you're heading towards implementation for example already"	16
Checklist	The use as a checklist	"Well it is a sort of list of things that you can tick-off and that's what I find very useful"	11
Time to comprehend	The time needed to comprehend the framework	"This session was quite quick, that complicates things. If you have more time you can prepare more thoroughly"	10
Tick-off boxes	The tick-off boxes were used	"Based on the description, I can tick off some boxes already"	8
Broad scope	The overall scope of the framework	"Framework is full and complete. In terms of deployment, I see good potential."	7
Factor quantifiability	The factors need a quantitative measurement	"Right, so there are no scores attached to these factors?"	7
Preparation	The framework can be used during preparation	"Yes it is especially useful for yourself, to prepare before going"	7
Factor terminology	The terminology used to describe each factor	"Maybe it would work better for me if it were stated in a question-form"	6
Layering in key factors	The three layers (dimensions) in the key factors	"The structure with the three layers appeals to me"	6
Factor interpretation	The interpretation of factors in general	"So all these, if you tick a box, then it is a positive factor for low-code BPM?"	4
Internal consistency	The internal consistency between various factors	"There are a couple of things where I wonder if they are double"	4
Wordy	The wordiness of the framework	"To be used for a quick model, it is a little too wordy"	3

Provided structure	Whether the framework allowed to be used structurally	“It helped me to structure the questions and to understand what the use case meant”	3
Selling	The framework’s use as a sales mechanism	“We can use it as a theoretical model”	2

F – Focus group materials

The collaborative, virtual board used in Miro



G – Fictitious use case simulations materials

USE CASE 1 (UC1) DESCRIPTION

Use Case 1 – PrivateBank

(Fictieve) naam	PrivateBank
Industrie/Sector	Bankensector
Belangrijkste product en/of dienst	Business-to-Business Kredietverlening
Grootte van de organisatie	100-500 werknemers

Eerst volgt er een omschrijving van het proces, daarna volgt het bijbehorende vraagstuk.

Het proces

Het van begin- tot eind faciliteren van het kredietverleningsproces: vanaf het moment dat de klant belt voor informatie tot het uitbetalen van het krediet, en alle stappen daartussen (o.a. aanbod, afsluiten bijbehorende financiële producten, opmaken contracten, acceptatie).

Het vraagstuk

Het proces wordt nu uitgevoerd door middel van Excel sheets die geprint, ondertekend, doorgegeven, verwerkt en gearchiveerd worden binnen PrivateBank. Eventuele aanpassingen worden bijgeschreven op het geprinte leningenoverzicht en doorgevoerd in het leningenregister. Dit leidt tot verschillende problemen:

1. Het proces wordt niet afgedwongen, verschillende medewerkers voeren op een andere wijze het proces uit;
2. Er is slecht zicht op iedere case: waar welk krediet in het proces zit etc. De hoge mate van uitzonderingen en zijtakken in het proces zorgt voor een onoverzichtelijk verloop;
3. Hierdoor is het ook niet mogelijk om KPIs te definiëren en te monitoren;
4. Key controls, bijv. contractbevestiging door Senior, kunnen niet worden afgedwongen;
5. Het is niet eenvoudig om data te controleren (wat er is afgesloten o.b.v. de Excel sheet en wat er in het leningenregister is geregistreerd)

PrivateBank zoekt een IT-oplossing om bovenstaande problemen op te lossen.

USE CASE 2 (UC2) DESCRIPTION

Use Case 2 – ElecSupply

(Fictieve) naam	ElecSupply
Industrie/Sector	Elektronica
Belangrijkste product en/of dienst	Verkoop en distributie van elektronische producten
Grootte van de organisatie	>5000 medewerkers

Eerst volgt er een omschrijving van het proces, daarna volgt het bijbehorende vraagstuk.

Het proces

Het van begin- tot eind faciliteren van het facturenverwerkingsproces: vanaf het ontvangen van de inkoopfactuur tot de betaling, en alle stappen daartussen (o.a. verwerken factuurgegevens, matchen van factuur aan bestelling, betaalverzoek opstellen)

Het vraagstuk

Het proces kenmerkt zich nu door veel handmatig werk en overdracht. Het is daarbij een ongestructureerd en niet afgedwongen proces terwijl er juist veel uitzonderingen zijn in het verwerken van de facturen. Veel communicatie verloopt via de mail, bijvoorbeeld, bij afwijkende aantallen of prijzen van een inkoopfactuur. Tenslotte opereert ElecSupply internationaal waardoor ze rekening moeten houden met verschillende wetgeving in hun proces.

Al met al is de wens om het proces waar mogelijk te harmoniseren en simplificeren. Vervolgens is het doel om de happy-flow (het standaard proces) te automatiseren zodat het team zich kan focussen op de uitzonderingen en de complexe facturen. De oplossing moet daarbij flexibel zijn om ook binnen verschillende regio's toegepast te kunnen worden. ElecSupply wil het proces dus standaardiseren en automatiseren en alles afvangen in één applicatie.

USE CASE 3 (UC3) DESCRIPTION

Use Case 3 – LogisticMaintenance

(Fictieve) naam	LogisticMaintenance
Industrie/Sector	Publieke Sector
Belangrijkste product en/of dienst	Logistieke dienstverlening
Grootte van de organisatie	>5000 medewerkers

Eerst volgt er een omschrijving van het proces, daarna volgt het bijbehorende vraagstuk.

Het proces

Het van begin- tot eind faciliteren van het logistieke (bestel)proces: vanaf de vraag naar een product tot aan het plaatsen van de bestelling, en alle stappen daartussen (o.a. checken huidige voorraad, opvragen van de beschikbaarheid, prijs en levertijd, valideren aan wet- en regelgeving, goedkeuring bestelverzoek).

Het vraagstuk

Het proces wordt nu volledig handmatig uitgevoerd. De logistiek medewerkers zijn nu genoodzaakt om meerdere (data) bronnen zelf te raadplegen en nagaan waardoor ze soms uren van hun dag bezig zijn. De instantie kampt al met een personeelstekort waardoor deze inefficiëntie dringend moet worden opgelost. Daarbij wordt het proces, met verschillende key controls (o.a. checks door Seniors of valideren aan regelgeving), nu niet afgedwongen.

Het doel is om met IT-ondersteuning de databronnen en systemen te integreren zodat medewerkers veel sneller en efficiënter de juiste informatie kunnen vinden. Daarbij streeft de organisatie naar een uniform en efficiënt proces, waarin alle nodige acties en checks worden afgedwongen. Door middel van één overkoepelde IT-applicatie wil LogisticMaintenance een digitalisatie slag doorvoeren in het logistieke (bestel)proces.

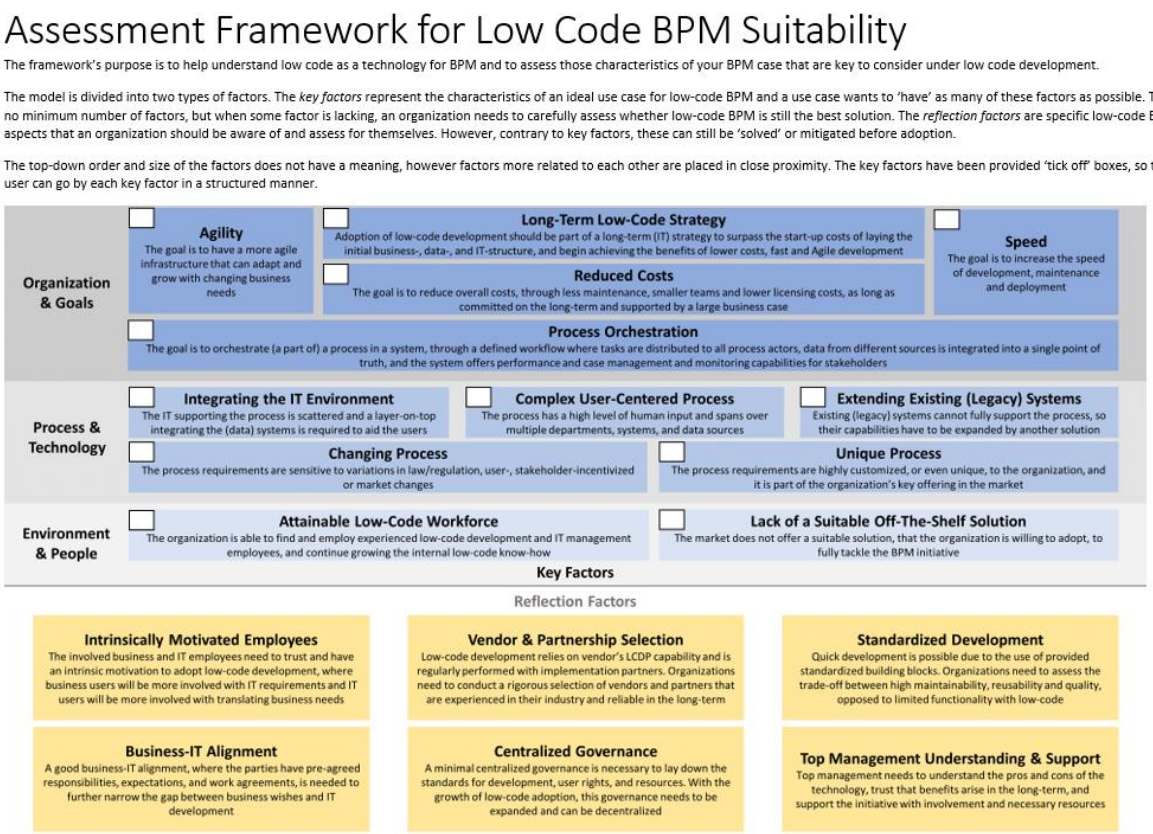
H – Framework including description

The following text was depicted above the actual framework when handed out to evaluation participants. The image below visualizes this.

The framework's purpose is to help understand low code as a technology for BPM and to assess those characteristics of your BPM case that are key to consider under low code development.

The model is divided into two types of factors. The *key factors* represent the characteristics of an ideal use case for low-code BPM and a use case should 'have' as many of these factors as possible. There is no minimum number of factors, but when a factor is lacking, an organization needs to carefully assess whether low-code BPM is still the best solution. The *reflection factors* are specific low-code BPM aspects that an organization should be aware of and assess for themselves. However, contrary to key factors, these can still be 'solved' or mitigated before adoption.

The top-down order and size of the factors does not have a meaning, however factors more related to each other are placed in close proximity. The key factors have been provided 'tick off' boxes, so that a user can go by each key factor in a structured manner.



I – Scientific paper submission

A FRAMEWORK TO ASSESS THE SUITABILITY OF LOW-CODE FOR BPM

Research Paper

Abstract

Organizations across all industries seek efficiency, digitization and automation of their business processes in current times. Low-code development platforms (LCDPs) promise time and cost reduction through rapid and easy-to-use application assembly. Even so, many organizations struggle to understand and identify digital solutions that can advance their business processes. Therefore, we propose a conceptual framework for organizations to assess their business process management (BPM) initiative for LCDP suitability. The framework is developed through a study of literature, a focus group, and expert interviews, resulting in 18 factors to be considered by organizations. An evaluation using fictitious use case analyses showed that the model was well-received, especially with regard to its completeness and operationality. To the best of our knowledge, this is the first work studying organizational adoption of low-code for the sake of BPM initiatives.

Keywords: Low-code, BPM, MDE, Digital Transformation

1 Introduction

A widespread increase in digital technology adoption has transformed the demands of businesses' customers and employees (van Veldhoven and Vanthienen, 2022; Konopik et al., 2022). Not only do organizations need to react with new digital technologies, they need to digitize their business processes to be able to respond to changing requirements (Denner et al., 2018). To support large, end-to-end processes in a flexible and adaptive way, Business Process Management Systems (BPMSs) are proposed as a suitable solution during digital transformation (Xu et al., 2018; Brkić et al., 2020). BPMSs allow to orchestrate and automate a process, improving process performance and organizational agility (Ravasan et al., 2014; Dumas et al., 2018; Xu et al., 2018).

A fundamental decision that organizations face when implementing a BPMS in their IT infrastructure is whether to build it in-house or acquire a packaged solution from a vendor (Ravesteyn and Batenburg, 2010). This dilemma, known as the traditional 'build vs. buy' decision (Hung and Low, 2008), has elaborately been studied in literature (Rands, 1993; McManus, 2003). Recently, a new type of solution has emerged: Low-code. It is becoming increasingly popular as it promises to combine the flexibility of building a solution, and the efficiency of buying one (Cicman et al., 2021).

Low-code is a method for assembly of Information Technology (IT) applications by eliminating most hand-coding for developers (Richardson and Rymer, 2014). Low-code development platforms (LCDPs) embrace low-code and offer a visual-based platform where the user interface, business logic, workflow, and data handling can be constructed rapidly through easy-to-use component assembly and modeling (Richardson and Rymer, 2014; Metrôlho et al., 2019; Sahay et al., 2020; Sanchis et al., 2020). Low-code's origins lie in Model-Driven Engineering (MDE) where models form the core of code generation and system development (Cabot, 2020; Bock and Frank, 2021a).

Low-code is seen as a major facilitator of organizations going through digital transformation (Sanchis et al., 2020) and it is often applied in process automation solutions (Luo et al., 2020). Yet, organizations struggle to understand and identify the digital solutions that can advance their business processes (Denner et al., 2018). No study to date has provided indicators of when low-code can be a suitable solution to organizations (Bock and Frank, 2021a). In this paper, we aim to close that gap for LCDPs

aimed at BPMS development, hereafter referred to as 'low-code BPM'. We propose a conceptual framework for organizations to assess the suitability of their BPM initiative to be supported by low-code BPM.

In the following section, we further conceptualize our definition of low-code BPM and provide theoretical background. Thereafter, we illustrate how Design Science is employed to create and evaluate the conceptual framework. In Section 4, the framework's structure and content is presented. The results from the framework evaluation are formulated in Section 6. Lastly, in Section 7, the results of the study are discussed together with indications for future research.

2 Theoretical background

Academic literature is yet to provide a clear conceptualization of an LCDP (Bock & Frank, 2021a) and low-code BPM. In this section, we formulate our definition of low-code BPM. Subsequently, we elaborate on the benefits, disadvantages and adoption reasons of low-code BPM.

2.1 Conceptualization of low-code BPM

LCDPs are platforms that allow rapid application development through low-code (Vincent et al., 2020), an approach where standardized, high-level functional components can be assembled easily in a visual designer, instead of textual coding (Metrôlho et al., 2019; Sahay et al., 2020; Sanchis et al., 2020). Generally, we found that LCDPs consist of three main functionalities: (1) they allow the modeling of (data) system structures and business processes, (2) they provide capabilities to design custom graphical user interfaces (GUIs), and (3) they offer flexible integration with external systems (Sahay et al., 2020; Vincent et al., 2020; Bock and Frank 2021a). Furthermore, Frank et al. (2021) present a classification of LCDPs into four groups: 'basic data management platforms', 'workflow management systems', 'extended GUI- and data-centric IDEs', and an all-encompassing fourth 'multi-use platforms' group. The first group offers features found mostly in data management systems and the third focuses on developing web- or mobile applications, both outside of our scope. As BPMSs were originally known as workflow management systems (Dumas et al., 2018), we, logically, consider low-code BPM to fall under the 'workflow management systems' group. 'Multi-use platforms' with an extensive BPM focus are also included as these include workflow management system characteristics.

Frank et al. (2021) propose that workflow management systems specialize in workflow automation, through conceptual modeling language like Business Process Model and Notation (BPMN) or other structures. Moreover, these platforms provide additional support for workflow execution, and integrations with internal and external systems (Bock & Frank, 2021a). The latter can be systems in the organization's IT landscape or AI (Frank et al., 2021), machine learning (Koplowitz, 2017), or other automating services that the platform provides. Lastly, low-code BPM provides functionality to build analytics dashboards to monitor process performance (Waszkowski, 2019). We combine overall LCDP features with 'workflow management system' specializations to define low-code BPM in this study as: *Low-code BPM supports rapid application development for end-to-end case management, with the workflow model and engine at its core, enabling process automation, integration, monitoring and enhancement with intelligent automation technology through easy-to-use component assembly and modeling.*

2.2 Background on low-code BPM

Cai et al. (2022) have shown that automating business processes is possible through low-code and that it reduces manual workload and improves IT flexibility. Waszkowski (2019) describes the design of a "BPM low-code platform" that represented similar functionalities as described in our low-code BPM conceptualization. For the rest, literature on the use of low-code for BPM is scarce.

Alternatively, several studies provide evidence of low-code's benefits and disadvantages in general. Low-code can increase software development speed and adaptability in organizations (Sahay et al., 2020; Sanchis et al., 2020; Frank et al., 2021). This is achieved through complexity reduction in low-

code (Alsaadi et al., 2021) which, in turn, allows employees with in-depth business knowledge to be involved (Sahay et al., 2020; Iho et al., 2021). In the case of BPM, this allows experienced process owners to develop and adapt the system to better suit the business process needs. Furthermore, cost reduction, increased maintainability, and improved system quality are seen as reasons for adopting low-code in organizations (Alsaadi et al., 2021; Bock and Frank, 2021a; Frank et al., 2021; Luo et al., 2021).

Literature also discusses the shortcomings of low-code. From a technical perspective, LCDPs have struggled with scalability issues, lack of customization on design and layout, and lack of interoperability between LCDPs (Sahay et al., 2020; Luo et al., 2021). Organizational considerations, such as distrust in the technology's abilities, concerns about a low-code vendor 'lock-in', or a steep learning curve have also hampered its adoption (Sahay et al., 2020; Sanchis et al., 2020; Alsaadi et al., 2021).

For organizations, the question arises under what circumstances the cited benefits outweigh the disadvantages as low-code is not suitable for all problems, in all organizations (Frank et al., 2021). Failing to recognize the right digital technologies can, instead, result in a loss of competitiveness for organizations (Konopik et al., 2022). For other digital technologies in BPM, such as robotic process automation (Plattfaut et al., 2022) or process mining (Mamudu et al., 2022), researchers have investigated what factors allow to assess the suitability of that technology for an organization's problem. We, now, study these for low-code BPM.

3 Research method

Our conceptual assessment framework has been constructed following the Design Science approach, as it "*creates and evaluates IT artifacts intended to solve identified organizational problems*" (Hevner et al., 2004, p. 77). To structure our research approach, we used the Design Science Research Methodology (DSRM) by Peffers et al. (2007) visualized in Figure 1.

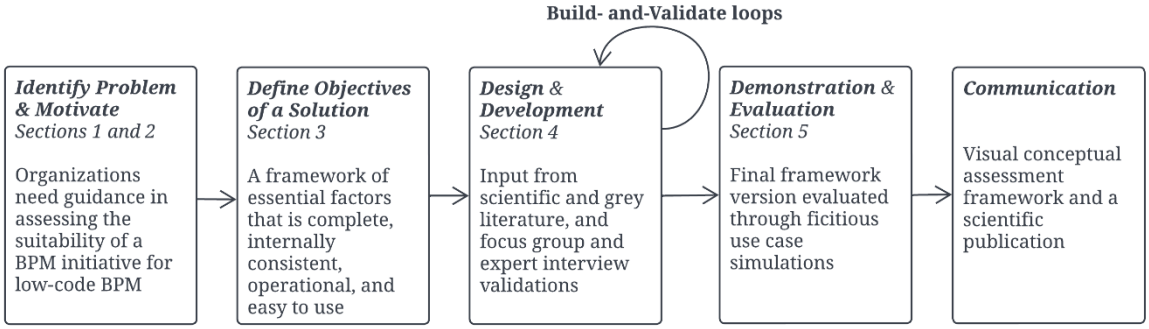


Figure 1. The Design Science Research Methodology (Peffers et al., 2007) applied in this study

The first step in the DSRM has been elaborated upon in the previous sections. Organizations need guidance in understanding and evaluating low-code BPM's suitability for their BPM initiative. In the following subsections, we discuss the other steps of the DSRM.

3.1 Solution objectives

We design an artifact, in the form of a conceptual framework, to help organizations to assess the suitability of their BPM initiative to be supported by low-code BPM.

Peffers et al. (2007) state that the criteria on which the artifact will be evaluated should be specified beforehand. We have combined relevant evaluation criteria for models and methods cited by March and Smith (1995), as our framework is both a model and a method to be used in an assessment. The chosen criteria are centered around its purpose: Our framework should provide a complete and consistent picture of all BPM initiative characteristics that differentiate low-code BPM from other solutions, and should enable organizations to assess the suitability of low-code for their business process management initiative through an easy-to-use framework. Furthermore, we have combined the definitions by Prat et

al. (2015) to clarify our four evaluation criteria: completeness, internal consistency, operationality, and ease of use.

3.2 Framework construction

In the design and development step, we constructed our framework. The initial version is based on a literature review, whereafter, two build-and-validate loops further refined the framework, as can be seen in Figure 1. The latter is presented in the following subsection.

We have studied and analyzed scientific literature regarding low-code, BPMSs, and MDE, and grey literature on low-code. Literature on BPMS and MDE has been included as low-code BPM aims to construct a BPMS, and low-code has evident origins in MDE. We have included grey literature as many acclaimed market research firms have been analyzing the advancement of low-code in recent years. With their focus on the industry's adoption of low-code, and a scarcity of scientific literature (Sanchis et al., 2020), these reports provide rich data. Only reports from renowned market research firms, large vendors, consultancy firms, or other implementation specialists have been selected. The final set contains 41 studies and documents, and has been used to synthesize a first conceptual assessment framework version.

3.3 Framework validation and evaluation

We have validated our framework in two rounds. The initial framework version has been validated by a focus group, as focus groups can explore a range of different ideas and represent various perspectives among a group of people (Krueger and Casey, 2014). This broadness is deemed useful as a first validation. The core question that the focus group had to answer is this study's research question: "How can organizations assess the suitability of their business process management initiative to be supported by a low-code development platform?". The participants have been carefully chosen to have various viewpoints and backgrounds related to low-code development, as presented in Table 1. Based on the input from the focus group, we developed a refined version of the framework.

#	Current Role	Experience with low-code BPM
FG1	Manager in Digital Transformation and Intelligent Automation	Leading and consulting on multiple low-code projects for application development and workflow automation using various low-code development platforms
FG2	Senior consultant in Digital Sourcing and Procurement	Developing and leading the technical implementation for multiple low-code development projects for application development using various low-code development platforms
FG3	Senior manager in Operational Excellence and Digital Transformation	Leading multiple projects on process excellence initiatives including overseeing multiple low-code BPM implementations

Table 1. Focus group participants.

Thereafter, the refined framework has been validated through eight low-code expert interviews. Expert interviews enable more in-depth analyses of topics in which the participant has expertise (Meuser and Nagel, 2009), allowing further refinement of the framework. As criterion sampling, each interviewee has to (*1*) have cooperated on at least one implementation of a low-code solution for BPM, and (*2*) have completed at least one low-code vendor's basic development training. The summary of all participants is given in Table 2. Input from the expert interviews has resulted in the final version of the framework, as presented in this paper.

#	Current role	Experience with low-code BPM
I1	Chief Technical Officer	Setting-up, implementing and overseeing the use of Mendix in their whole organization

I2	Director and IT implementation specialist	Consulting organizations with decision-making and managing IT implementations through OutSystems
I3	Lead Appian consultant	Leading the implementation of Appian throughout a large corporate and consulting small-medium enterprises on Appian
I4	Freelance Business & Digital Transformation expert	Leading the implementation of various LCDPs in various organizations
I5	Appian developer	Building Appian applications for various organizations
I6	Interim Chief Information Officer	Managing and decision-making on IT implementations, including LCDPs, in various organizations
I7	Partnership manager in Intelligent Automation	Connecting organization needs with Intelligent Automation and low-code BPM vendors
I8	Lead in consulting for Intelligent Automation	Setting-up a low-code BPM department and consulting on Intelligent Automation opportunities

Table 2. Expert interview participants.

In the next step of the DSRM, the definitive framework has been demonstrated and evaluated through fictitious use case simulations. An interview and survey at the end of these simulations gauges the criteria from Section 3.1. The entire evaluation method has been described in Section 6.

3.4 Data analysis

We have used thematic analysis as a systematic method, described by Braun and Clarke (2012), to analyze our qualitative dataset, consisting of literature and transcribed validations and evaluations. We used a combination of deductive and inductive coding to derive the framework's contents.

A deductive coding approach uses a pre-existing framework or theory to later find themes of interest in the dataset (Crabtree and Miller, 1999). At first, we have identified two existing models that provided scientifically substantiated dimensions that could be used to categorize our factors. These dimensions are presented in the next section. Thereafter, we analyzed the literature, with the dimensions in mind, and labeled it with codes of interest. We searched and reviewed our codebook for themes, categorized them under the dimensions, and gave the themes a definition and a description, according to the methodology by Braun and Clarke (2012). These themes are eventually included as factors in our conceptual assessment framework.

The analyses of the validations and evaluation followed a similar procedure. However, after deriving the themes during analysis, we revisited the existing factors and their substantiation. As we conducted and analyzed each method sequentially, contradictions and discrepancies can occur when integrating results (Moran-Ellis et al., 2006). Therefore, throughout our results, we present our rationale on how we reconciled such discrepancies.

4 Low-code BPM assessment framework

The definitive conceptual assessment framework is presented in Figure 2. It includes 18 factors, representing various topics to assess or consider, divided into two factor types. The *key factors* represent the characteristics of an ideal use case for low-code BPM and a use case should 'have' as many of these factors as possible. There is no minimum number of factors, but when a factor is lacking, an organization needs to carefully assess whether low-code BPM is still the best solution. The *reflection factors* are specific low-code BPM aspects that an organization should be aware of and assess for themselves. However, contrary to key factors, these can still be 'solved' or mitigated before adoption. These two factor types divide the framework.

As mentioned earlier, two existing theories support our framework. Whittle et al. (2017) present a taxonomy of MDE tooling considerations that shape successful MDE adoption and use. Due to low-code's origins in MDE, these themes represent the technological side of our framework. Vom Brocke et al. (2016) provide the BPM context for this framework through the 'morphological box to identify the

context of BPM'. By combining the distinct themes in these models, we end up with six dimensions under which each factor in our framework falls.

The dimensions can be found, sometimes rephrased, in the framework as layers separating the key factors. Moreover, key factors have white tick-off boxes so users can structurally go through the framework. We have not included these features for the reflection factors, as these are more intended for a general understanding of important low-code BPM considerations.

In the following subsections, we present how findings from literature, and input from the focus group and expert interviews are incorporated into the 18 factors. Last, the framework's design features are substantiated.

4.1 Goal dimension factors

This dimension describes what organizational goals low-code BPM is especially suitable for.

Process Orchestration (Key) – Integration of the whole end-to-end process is a common problem for organizations in current times (Xu et al., 2018). Our expert interviewees argued that low-code BPM allows this, clearly illustrated by I8: "*The entire case was managed by a process workflow. So the notion of tasks. And I distribute those tasks, I assign them to people, to groups, and they take ownership of that. That is kind of the core of the whole process orchestration, people-in-the-loop. But I have the system driving the business logic, as who should be doing what now, instead of people doing that themselves*". This corresponds with the characteristics of workflow management systems as described by Bock and Frank (2021b) and our low-code BPM conceptualization. A core component to achieve such process orchestration is low-code BPM's integration with the IT environment (Frank et al., 2021). I3 illustrated this with a practical low-code BPM example: "*A know-your-customer process, that often involves multiple departments, having to pull data from multiple sources and bring it all together into one bundle [...] that is a clear process orchestration initiative*". Low-code BPM allows process orchestration to reduce inefficiencies, improve performance, and provide transparency in the process.

Speed (Key) – Low-code development enables organizations to develop products in a short time (Sahay et al., 2020; Alsaadi et al., 2021; Sanchis et al., 2021) and low-code vendors explicitly mention development speed as a key benefit of low-code (OutSystems, 2019). All experts agreed that this is one of the key goals achieved with low-code, also in the case of BPM, as mentioned by I2: "*The competitive edge [of low-code] is found in the speed and the agility it offers […] where you talk about implementation periods of about 1 to 3 months*". I6 had seen it being a decision point: "*At [organization] the reason [for adopting low-code] was that we wanted to develop relatively quickly*" (I6). Low-code BPM can be the answer for organizations that need to develop IT solutions quickly (Sanchis et al., 2020).

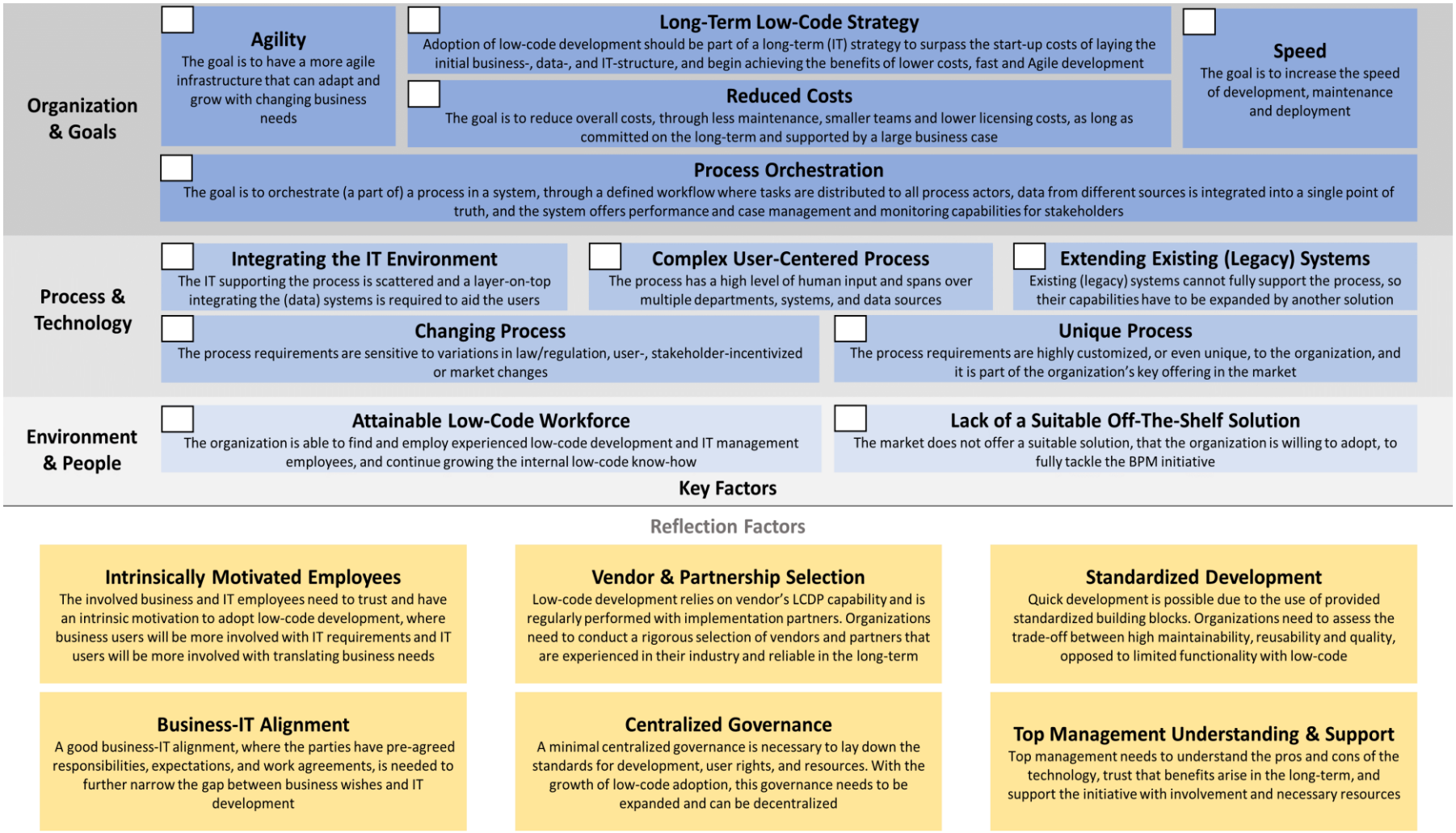


Figure 2. Low-code BPM assessment framework.

Agility (Key) – Besides speed, a key reason for low-code adoption is the increase in organizational responsiveness to changes (Alsaadi et al., 2021; Frank et al., 2021; Cai et al., 2022). Organizations have employed low-code BPM "*where process changes are expected quickly, for example, legislative changes. Or where the consumer demands something different all the time*" (I4), but low-code also allows to "*build a system that evolves with you, grows with changing business*" (I2). Business users can be involved in application management which allows organizations to adjust to changing market conditions more easily (Olariu et al., 2016). Therefore, low-code development is suitable for organizations that require high levels of agility in their IT landscape. An important reflection, concerning speed and agility, is the effects of standardized development discussed further.

Reduced Costs (Key) – Cost-effectiveness is also possible through the employment of low-code in the long-term (Luo et al., 2021; Cai et al., 2022). Through the reuse of proven components, each subsequent project has the potential to reduce overall costs (Cicman et al., 2021). I6 suggested: "*If you have a lot of component reuse, you can develop your new application faster and faster, that is cheaper than coding an application each time […] Because you need fewer people, you have lower licensing costs, less maintenance, et cetera*". To become cost-effective, low-code development should be part of a long-term IT strategy where the organization continues actively developing using low-code, experts state. I8 explained: "*As you start scaling up and getting a few apps, you start seeing economies of scale. Essentially, it is like I am using one platform to do multiple things, so I'm not paying for a new license each time*". Organizations seeking lower IT development costs can consider low-code as a solution.

4.2 Organizational dimension factors

This dimension includes organization-wide characteristics that are essential under low-code BPM.

Long-Term Low-Code Strategy (Key) – MDE adoption literature shows that organizations working on a project-by-project basis unjustly disregard the technology while organizations with a progressive and iterative project approach succeed (Hutchinson et al., 2014; Whittle et al., 2017). Experts, such as I6, also emphasize this: "*Building one application with low-code is really a total waste of money, you really need to have a strategy […] Only when you get to 5/6 applications, then you suddenly see the added value of low-code*". Not only the benefits depend on long-term adoption, as I3 illustrated: "*If you train people in certain tooling, it only makes sense if that is on the long-term, if there's a long-term vision there. Or you say, we'll bring in a partner with whom we'll do a partnership to make it a success. But still, in a partnership, you also only do that for the long-term*". Therefore, it is key that organizations incorporate low-code in a long-term strategy used for, possibly, multiple solutions in the future.

Business-IT alignment (Reflection) – Lacking business-IT alignment is seen as a deep-rooted obstacle for any new technology implementation in an organization (Luftman and Brier, 2019). Even further, good Business-IT alignment has been essential for MDE adoption as it is often a business decision, where the business- and IT goals are to be aligned (Whittle et al., 2017). Multiple experts agreed to the importance of business-IT alignment, especially in the starting phase as I6 illustrated: "*It is a cultural aspect, you are suddenly going to put two cultures together and say, well, let's work together. You have to guide that well*". Apart from its presence prior to adoption, low-code development in an organization gives the "*ability to involve the rest of the organization in the development process*" (I1), raising the business-IT alignment. This was a reason for adopting low-code BPM in I1's organization. Organizations ought to assess themselves on whether their business-IT alignment is suitable for low-code BPM adoption.

Centralized Governance (Reflection) – IT governance entails the guidance and structures in place to sustain the IT infrastructure in enabling an organization's strategy and objectives (de Haes and van Grembergen, 2009). Proper governance is deemed important in BPM initiatives (vom Brocke and Schmiedel, 2014) as with MDE adoption in organizations (Aranda et al., 2012) to result in long-term benefits for an organization. Most experts agreed that a certain level of governance is needed, as I3 explained: "*If we start with a team or within a department, it is a bit of an overkill to set up a dedicated device like that […] but if you let everyone onto the platform, you get uncontrolled growth, so it is crucial to set-up certain [development] standards*". Hoogsteen and Borgman (2022) argued that a centralized governance structure is the most suitable form for low-code development. Experts expanded on this notion, as I4 stated: "*In the starting phase: centralized, standards, laying the foundation. When*

building onwards, then decentralized is really perfect". Therefore, organizations should be aware that implementing low-code BPM entails setting up a centralized governance structure in the organization.

Top Management Understanding & Support (Reflection) – Especially in the case of technology adoption, top management support is vital (Hsu et al., 2019). Active top management support includes, for example, communication of opportunities or proactive guidance through technology training as opposed to pure decision-making with passive support (Hoogsteen and Borgman, 2022). However, various experts explained that organizations struggle to fully grasp the technology and how efforts can be translated into results, as I7 explained: "*In the beginning, you will not notice anything. Because what are you doing? You are putting a layer on top of what you already had. It does not give any new functionality*". Therefore, I6 argued that understanding is key: "*The board has to be along too, they have to understand why it is important, and that it takes time, and it will not be so quick in the beginning*". Therefore, this factor accentuates the understanding and support that the top management should have on how the cited benefits of low-code come about.

4.3 Process dimension factors

All process-related characteristics suitable for low-code BPM support are discussed in this dimension.

Complex User-Centered Process (Key) – BPMSs support modeling processes exactly as they are performed (Ravasan et al., 2014). Process selection is, therefore, key as organizations can endlessly map business processes. Dumas et al. (2018) proposed various criteria including the process size where at least three different actors have to be involved. A certain level of process complexity is mentioned by FG3 as key for low-code BPM: "*Longer chain processes spanning multiple departments are often a good indicator for high [low-code BPM] potential. Because those kinds of processes are tough to manage*". Experts agreed that the full potential of low-code BPM is unlocked with such complex processes. Moreover, I7 emphasized the user, and how low-code BPM imposes a standardized, clear way of sharing data, in such complex processes: "*If the complexity is that you have many different parties or components that do not communicate well with each other, that use Excel exports […] low-code could indeed possibly be the solution for you*". Therefore, organizations orchestrating complex, user-centered processes are well suited to low-code BPM.

Unique Process (Key) – Make-or-buy literature states that core processes are better supported by acquired packaged software while strategic systems should be developed in-house (McManus, 2003). In line herewith, low-code's customizability is especially useful and becomes cheaper than adapting packaged solutions (Bratincevic, 2020), for unique processes. Experts concurred with the above, as FG1 explained: "*we see that [low-code is used when] the market cannot fulfill all the [process] requirements […] you buy standard applications for your HR or CRM processes, which is much cheaper*". Uniqueness, however, is often hard to measure objectively. I2, therefore, further emphasized uniqueness: "*The system […] has to be an integral part of that business, that the system should help that customer differentiate itself from the rest of the market*". Low-code BPM is, therefore, ideal for highly customized business processes, part of the organization's key offering.

Changing Process (Key) – Low-code BPM provides a flexible modeling approach that, together with development speed, allows constant change in the process (Koplowitz, 2017). Therefore, processes that have a tendency to vary become compelling for low-code BPM. FG1 saw in his own experience: "*If a lot of flexibility is needed in the process or there can be a lot of changes in process requirements, then you can swiftly use low-code*". I6 drew a comparison to non-changing processes: "*Is it a solution that is in your core process and does not change much? Then I would take a SaaS solution or an ERP solution, or at least a standard solution*". All in all, apart from complex and unique processes, low-code BPM suits organizational processes that are sensitive to changes.

4.4 Technical dimension factors

This dimension discussed technical considerations and characteristics specific for low-code BPM.

Integrating the IT Environment (Key) – BPMS compatibility with the surrounding IT infrastructure is essential for successful implementation (Bosilj Vukšić et al., 2018). Low-code BPM, as the 'process orchestration' factor showed, allows to flexibly integrate (data)systems underlying the process (Sahay et al., 2020; Frank et al., 2021) to create an orchestration layer over the IT landscape. I7 explained how

this helps organizations to manage their process: "*Low-code BPM is a layer over your systems […] where you can pull information from different systems, forming it into a unified process in that one layer which means you have a lot more control over how that process goes. […] If something changes in the process, or it does not work, or you want to improve something, you do that in the above layer*". Therefore, I8 sees low-code being applied in: "*An organization where they're a very decentralized, they've got systems all over the place, they have a very scattered landscape […] Having one platform in which they can connect everything, to some degree, is very powerful*". Hence, low-code BPM is a solution for organizations with a scattered IT landscape, supporting the process which needs integration.

Extending Existing (Legacy) Systems (Key) – In a report by OutSystems (2019), a common use of low-code was found to be extending existing systems' functionality. As low-code BPM serves as a layer-on-top of your IT infrastructure, functionality can be added in that layer as well. FG2 recounted a client case: "*You can use it as a custom layer on top of large packaged solutions, in SAP where you do not want to customize due to complexity and costs. […] You can add some extra process digitalization if it is not possible in a packaged solution*". Moreover, Sahay et al. (2020) argued low-code being used for the integration and extension of legacy systems as well. This allows to communicate and use the functionality of legacy systems without making any adjustments in the settings. I4 explained: "*With large companies, you have a choice: Am I going to remove the legacy systems? Then you are going to have to convert which are expensive projects. But you can also offer a solution through low-code where you make Appian communicate with your legacy. That is easy to do and you can let your legacy be as it is*". Therefore, if organizations have systems that they do not want to adjust for the sake of process orchestration, low-code provides an unintrusive way of communicating with (legacy) systems.

Standardized Development (Reflection) – I5 summarized low-code development as: "*The disadvantage is also very much an advantage. […] So you use certain standardized blocks and you link them together. Because of the blocks, you are limited in what you can do, which also means you have a fewer bugs. So you have a fewer problems to solve but it does limit what you can and cannot do*". On one side, concerns about the ability of LCDPs have previously deterred adoption (Sanchis et al., 2020; Alsaadi et al., 2021; Luo et al., 2021). On the other side, experts argued that maintainability, reusability, and quality are increased due to the use of standardized blocks. I4 summarized these benefits through the following analogy: "*You can imagine it is much easier to build a castle with Lego blocks than with sand. Because those low-code blocks have great quality, you can hardly make any mistakes*". Organizations should, thus, carefully assess if their software development requirements are met by the LCDP.

4.5 Social dimension factors

These factors concern employees and middle-managers responsible for implementing low-code BPM.

Attainable Low-Code Workforce (Key) – Although low-code vendors promote that any business user can develop on their platform, low-code development still requires a technical background (Frank et al., 2021; Hoogsteen and Borgman, 2022) and has a steep learning curve (Luo et al, 2021). In the meantime, experts argued that experienced know-how is essential when implementing low-code. I6 illustrated the current problem with this: "*There are not that many people who have enough experience. […] Those that have are very expensive and rare. […] The whole market is still catching up in that respect*" (I6). All experts that directly led low-code implementations (FG2, I1, I2, I3, I4, I6) emphasized the difficulties in finding experienced employees. For example, I4 mentioned: "*Suppose you start using Appian in the Netherlands, there are no good people left to get because they have now been bought up by [company X], [company Y], and others*". Moreover, it does not only concern developers, but also solution architects and business analysts. Being able to attain a low-code workforce is a key decision-making point when considering low-code BPM.

Intrinsically Motivated Employees (Reflection) – MDE studies have shown that developers mistrusting the technological capabilities employed work-arounds, hereby defeating the purpose of the technology (Hutchinson et al., 2014), or reverted to traditional development (Aranda et al., 2012). Mistaken beliefs in BPMS capabilities previously caused fear of job loss, dissatisfaction, and conflicts among employees (Bach et al., 2017). The attitude of both business users and developers is also important for low-code BPM adoption. Regarding developers, I1 explained: "*you cannot dump a completely new tooling on your Java-group, that just does not work*", while regarding business users,

I6 emphasized: "*Business is going to have to put a lot more time in specifying and prioritizing user stories […] if they do not want to put the time in that, than it still is not going to work*". It is important organizations assess for themselves how their employees foresee the implementation of low-code BPM.

4.6 Environmental dimension factors

This dimension relates to topics that are outside of the boundaries of the organization.

Lack of a Suitable Off-The-Shelf Solution (Key) – McManus (2003) stated: "*Executives agree that certain unique business applications will necessitate creating software in-house, regardless of time or cost considerations*". Low-code's customizability offers to cater to such unique business applications (Alsaadi et al., 2021). In our scope, a unique business application can be needed to suit a 'unique process', as seen with an earlier factor. However, albeit a solution for the process exists, I4 explained: "*Or you just do not want to accept a solution. Even if the solution fits for 80/90%, then I would still not consider it suitable*". Although the percentages are subjective, a lack of a suitable off-the-shelf solution can be a key indicator for low-code BPM adoption. I1 explained how this played a role in an organization choosing low-code: "*The organization spent a very long time looking for a package for settling claims and to get a handle on the backend of that process. They did not find anything for that*". Therefore, a lack of a suitable off-the-shelf solution is a key indicator that low-code BPM is a beneficial solution.

Vendor & Partnership Selection (Reflection) – BPMS literature highlights the vendor's reputation, knowledge, and experience in serving clients as an important contributing factor (Bosilj Vukšić et al., 2018). As organizations are dependent on the vendor's platform, and fear vendor lock-in (Alsaadi et al., 2021), a careful vendor selection should be conducted (Cai et al., 2022). Moreover, an implementation partner is often proposed for their knowledge, as I5 explained: "*you do need someone with knowledge looking over your shoulder […] if you have to do it all yourself, if you have to reinvent the wheel, it will take a long time*" (I5). Moreover, FG2 emphasized this importance in the scope of low-code BPM: "*We now really focus on low-code BPM tooling or platforms, which is, in my experience, actually always implemented through external partners*" (FG2). Organizations ought to realize the importance behind rigorous selection of vendor and implementation partnerships when considering low-code BPM.

4.7 Visual design of the framework

The framework went through three design iterations to come to the final version. In the process, various design decisions have been taken, further explained in this section.

The focus group showed how different factor types used interchangeably caused confusion. By segregating the factors based on factor type, we integrated structure into the framework. Users can now interpret each 'part' of the framework in their own way. Moreover, multiple experts proposed using the framework as a 'tick-off' list. I2 proposed: "*Yes, you can check off on that. I think it would be nice for a lot of my clients to have*" and I7 envisioned: "*Look, if I made it simple, this would be a very nice checklist […] for when we go to clients. So do you meet these points, is this indeed approximately what you want with key [factors]? We have reflection, well keep in mind that within the organization so you do have to start arranging these kinds of things*". Therefore, white boxes are added to the key factors as 'tick-off' boxes. Lastly, we have decided not to explicitly include the dimensions in the framework. Initially, two variations of the final framework were made with and without color-coded dimensions. The researchers and various colleagues agreed that the simplified framework was easier to perceive and the dimensions did not add much to the framework. Moreover, the global 'themes' on the right still provide some context on the factor's origin.

5 Framework evaluation

We evaluated the framework on the criteria posed in Section 3.1 with five fictitious use case simulations. In each simulation, a participant evaluated a fictitious use case's suitability for low-code BPM.

Historical, real-life use cases from a consultancy firm were used for these simulations. Prior, the manager of each use case explained the business problem that the organization had faced and a use case

description was formulated together with the researchers. As the participants were consultants in the same firm, each description was anonymized, so it could not be traced back to the actual projects. During each simulation, we introduced the low-code BPM assessment framework and the use case description to the participant. Moreover, the participant could interview the manager for more in-depth insights on the use case. The participants assessed each fictitious use case's suitability for low-code BPM using the framework. After the simulation, each participant gave their assessment, its rationale, experience while using the framework, and the participants filled a survey measuring the evaluation criteria on a 1 to 5 scale. This section presents how the framework performed during the fictitious use case evaluations.

Completeness was rated highly by the five participants with an average score of 4.4/5. The participants agree that the framework's dimensions cover the general topics that they would consider when assessing a use case. Moreover, one participant explicitly emphasized the broad spectrum of questions that can be asked using the framework and: "*On the basis of this, I think you can give an excellent opinion whether or not it fits the organization at first glance*". Another participant proposed to utilize this model as a theoretical model, whereafter specific factors are further explored in an organization's case. In general, participants agreed that it is a complete framework for an initial use case assessment of an organization.

Internal consistency scored lower with a 3.7/5. The factor definitions, and how the factors should be interpreted through key- and reflection factor types, were understood and accepted by the participants. However, participants struggled with some factors overlapping in definition, such as 'Agility' and 'Changing Process'. Whereas the first focuses on an organizational goal and the latter on a process characteristic, we agreed that these could be confusing. This is a potential improvement for future research discussed in the next section. Lastly, some participants questioned how specific certain factors were to low-code BPM. Although it is true that some factors, such as 'Business-IT alignment', are more generally applicable to any adoption of technology, the substantiation of each factor explains why, and how, the factor is relevant in the context of low-code BPM.

Operationality received an average of 4.2/5 from the participants. Participants see the framework as a useful fundamental theory during, or when preparing for, an interview. Some used the tick-off boxes for this purpose, while others added plus- and minus signs. One participant noted the following regarding his approach: "*I briefly grabbed that framework during the interview to see like, okay do I have all the points? Did I walk through it all? Then I know for sure that I can give good advice instead of doing it on gut feeling*". Moreover, participants noted that the difference in factor types allows them to prioritize the assessment and, possibly, initially omit reflection factors. Nevertheless, some participants argued that the framework leaves much to interpretation which makes it too subjective to do a thorough assessment. An often-recurring theme was the quantification of factors, explained in the next paragraph.

Ease of use scored a bit lower than operationality, with a 3.9/5. Predominantly, the participants struggled to fully grasp the framework and perform an assessment in the 30-minute session. Consequently, some factors or other aspects were misinterpreted. One participant shared: "*Having done it once already, I can also much more easily think of sub-questions. Now, sometimes I got stuck*". Therefore, instead of an ad-hoc framework that can be used instantly, this framework is more suitable for careful studying of a use case. More objective quantification of factors could further elevate this framework. Multiple participants asked how a missing factor or how the factor weight should be interpreted. Future research could find specific indicators of how an organization is performing at a certain factor. This would help in interpreting the framework but also in making it more operational to do an actual assessment.

Concluding from the evaluation, we argue that our framework is useful for an initial judgement of a use case's suitability for low-code BPM. However, it would help users to make the factors less interpretable and specify how factors are related to each other to allow a more rigorous and objective evaluation.

6 Discussion and conclusion

In times when organizations are urgently looking to digitize, optimize, and automate their business processes, low-code BPM emerges as an effective solution. Low-code vendors advertise the easiness of

their product and potential benefits while scientific literature nuances these benefits with concerns on customization, vendor lock-in, and complexity. Meanwhile, organizations struggle to identify the right solutions to advance their business processes, but existing literature does not provide guidance on assessing whether low-code BPM can be that solution. To solve this gap, we proposed the low-code BPM assessment framework. It consists of 18 factors to be considered by organizations regarding low-code BPM support and illustrates important considerations during implementation.

The scientific contribution of our study is twofold. Firstly, our study is one of the first to research low-code in the context of BPM. Cai et al. (2022) proposed a method for process automation through use of an LCDP. However, their method assumes the low-code decision to have been made and focuses on a small-scale automation. Still, Luo et al. (2020) found the process automation and BPM domains among the most-used in practice, making this a notably unresearched area. Therefore, our study contributes by forming a scientific understanding of this application and we encourage researchers to use and extend this scientific base when studying low-code BPM. Secondly, our study extends current knowledge on the organizational use of low-code. Various studies have been performed on low-code's features (Sahay et al., 2020; Frank et al., 2021; Luo et al., 2021) and to elicit the low-code adoption reasons (Sanchis et al., 2020; Alsaadi et al., 2021). But none describe the criteria for effective organizational use of low-code (Frank et al., 2021). Our framework presents these criteria through 18 factors in the BPM context.

Secondly, our study extends current knowledge on the use of low-code in organizations. Various studies have been performed on low-code's characteristics (Sahay et al., 2020; Frank et al., 2021; Luo et al., 2021) and to elicit the reasons for low-code adoption (Sanchis et al., 2020; Alsaadi et al., 2021). But none describe the criteria for effective use of low-code in organizations (Frank et al., 2021). Our framework presents these criteria in the form of 18 factors focused on the BPM context.

Our study also has practical contributions. Firstly, the framework can be utilized by organizations to assess the potential of low-code BPM for their processes. The evaluation showed that the framework serves as a theoretical map to understand which elements are key and which need to be reflected upon. Secondly, implementation specialists and consultants can use the framework to prepare and validate a more thorough assessment of an organization's BPM case leading to an IT adoption decision.

Several limitations of this work ought to be discussed. The reliability of qualitative research in evaluating information systems decreases due to the subjectivity of researchers' interpretations (Kaplan & Maxwell, 2005). To combat this threat to reliability, we have used both scientific and grey literature from various domains, and employed various empirical analyses to triangulate the data gathered throughout this study. We argue that this represents a credible picture of significant low-code BPM factors. Nonetheless, low-code itself has never clearly been defined, making the study of a specific type of LCDP potentially inconclusive. Interviewees could misinterpret the object of study, threatening construct validity. Therefore, we have conceptualized our definition of low-code BPM based on existing literature and communicated it to all participants in this study. Lastly, the framework's evaluation consisted of a rather small sample of five participants. Albeit true that a larger evaluation sample size could be more fruitful, the comments made by the participants became increasingly identical, indicating theoretical saturation. Moreover, earlier comments by experts coincided with the evaluation results.

Still, an evaluation of the framework conducted at an organization facing an IT adoption decision would greatly benefit the reliability of the framework, forming one direction of future research. Another would be a quantitative study to validate the framework elements and their relationships in between. This would further strengthen the reliability of the framework. To improve the operational potential of this framework, studies can focus on providing each factor with measurements. Organizations would, then, be able to rely on object factor measurements, instead of subjective analyses of their BPM initiative. Finally, research can focus on the adoption criteria of other LCDPs outside of the BPM scope. Our study is a first step to help understand and recognize when low-code can advance organizations further. We encourage to use and extend this knowledge in further developing low-code understanding.