An Investigation of Why Low Code Platforms Provide Answers and New Challenges

Edona Elshan
University of St.Gallen, Switzerland
edona.elshan@unisg.ch

Ernestine Dickhaut
University of Kassel, Germany
ernestine.dickhaut@uni-kassel.de

Philipp Ebel
University of St.Gallen, Switzerland
philipp.ebel@unisg.ch

Abstract

Although the idea of low code development is not new, the market for these oftentimes platform-based development approaches is exponentially growing. Especially factors such as increasing affinity for technology development across all user groups, consumerization of development, and advancing digitalization are opening a new target group for the low code movement. The broad application possibilities of low code, as well as the benefits, are therefore getting more important for businesses. Especially for small and medium-sized enterprises (SMEs), low code constitutes a promising avenue to survive and succeed in the rapidly changing world. However, a clear understanding regarding the application of this paradigm of software development in SMEs is still missing. To provide a coherent understanding of the phenomenon low code in SMEs, we review extant literature and conduct interviews, identifying potential application domains and conceptualizing the benefits and challenges of low code from a holistic perspective.

Keywords: Low-code platforms, no-code, opportunities, challenges.

1. Introduction

"*Software is eating the world*" (Andreessen, 2011).

In today's world, companies must now undergo digital transformation to remain competitive and hence survive in the market (Bexiga et al., 2020). To push the digital transformation, we observe, that many companies are developing digital products. Alt et al. (2020) argue that this maxim is about much more than the mere automation of corporate processes and business models. For example, increasing digitization also further establishes the logic of software in companies, creating additional exponential growth mechanisms that far exceed the purely linear input-output function. While in the 2000s, this digitalization was only on the rise, in the present time, it is becoming an ever-greater necessity. Especially in a so-called VUCA environment, the call for agility at all levels of the company is getting louder and louder (Baran & Woznyj, 2020). Thus, digital transformation can be essential, especially for small and medium-sized enterprises (SMEs), in order to survive and succeed in the rapidly changing world - especially due to their thereby increased innovative and relational capacity (Troise et al., 2022). While concepts such as DevOps, which bring together the development and operation of software solutions and thus make them more efficient, these technical aspects must subsequently be merged with the business-relevant processes in the sense of BizOps (coupling of IT and business goals) in order to adapt enterprise software more precisely to companies (Fokaefs et al., 2017). However, this is fraught with difficulties. Cost overruns, conflicting project requirements, overly long development times, or even a lack of business-IT alignment are examples of these (Charette, 2005). Low code platforms (LCPs) may be a manner to overcome these hurdles. Unnecessary revisions, which cost a lot of time and money, can be avoided by having the necessary expertise at the beginning of the development process. Programming environments that allow non-coders, such as business developers with domain know-how, to program simple systems may empower non-developers to write small programs on their own (Waszkowski, 2019). The systems can quickly be adapted with little effort, which is beneficial for agile system development. Within companies, these LCPs are part of a larger trend of technology democratization (Brinker, 2018), referring to any undertaking that traditionally required coding but can now be accomplished by a business user. Thus, they set the goal of empowering users to develop their own software solutions as well as to be able to quickly make adjustments to enterprise software to meet rapidly changing requirements due to the increased productivity in the development phase (Bock & Frank, 2021). This theoretically enormous potential is also recognized by Gartner (companies specializing in IT process comparisons) reaffirmed. They estimate the share of applications developed by low code to reach 65% by 2024 (Vincent et al., 2019), with at least four platforms being used by about 66% of all large companies

URI: https://hdl.handle.net/10125/103380
978-0-9981331-6-4

(Koksal, 2020). Resulting in the following research question:

RQ: What opportunities and challenges of low code platforms can be identified in SMEs?

We conceptualize with our analysis what potentials are associated with the use of low code. In the following, we provide our theoretical background, describe our methodology approach, and present our findings. Thus, we analyze opportunities and challenges that come with the use of low code from a practical viewpoint. We hope this work will support and stimulate future research; by providing recommendations to facilitate the use of low-code platforms.

2. Theoretical Background

2.1. Low Code Development Platforms

Forrester characterizes low-code platforms as: "*products and/or cloud services for application development that employ visual, declarative techniques instead of programming and are available to customers at low- or no-cost in money and training time to begin, with costs rising in proportion of the business value of the platforms*" (Richardson et al., 2017, p. 438).

The terminology is not uniform due to the technological advancements and developments in the research. In addition to low-code platforms (LCPs), terms such as low-code application platforms (LCAP) or low-code development platforms (LCDP) can also be found in literature (Bock & Frank, 2021). In the following, the term low-code platform (LCP) will be used.

Regardless of their designation, they allow users of the platform to develop applications based on a graphical user interface (GUI) without having to demonstrate competence with hard-coded programming techniques themselves (Waszkowski, 2019). In the background, code is usually generated automatically, which can be adapted by IT personnel (Wang et al., 2021). Nevertheless, this assumes a certain familiarity with documentation and an understanding of the syntax of the code (Lugovsky, 2021). Their focus is primarily on the development of databases, enterprise processes, and the user interface (UI) for web applications (Waszkowski, 2019). The goal and rationale of LCPs is typically to help organizations build and deploy applications without the need for a well-trained team of developers (Tariq, 2021). However, LCPs are not just for use by non-professionals: software developers also use no-code platforms such as Jenkins for continuous building and testing of software releases (Lethbridge, 2021). There are numerous LCPs on the market, with OutSystems, Appian, and Mendix being examples of popular platforms (Gartner Inc, 2022). These all differ in their specifications and features. Sahay et al. (2020) analyzed eight of the most popular LCPs and identified the following four layers that they all have in common:

- Application layer consists of the graphical environment (GUI) with its toolboxes and widgets, through which programmers develop applications.
- Applications are connected to various external services via the service integration layer using application programming interfaces (APIs) and authentication mechanisms.
- Data Integration Layer allows heterogeneous data to be merged and processed homogeneously.
- Deployment layer provides the applications and takes them into production. Depending on the platform, the applications can run either on a cloud or on-premise infrastructure.

2.2. Low Code versus No Code

No-code is also a relevant concept when it comes to enabling non-IT personnel to develop business-relevant applications. Unlike low-code, no-code requires absolutely no understanding of computer science (Tariq, 2021). Their work witnesses strongly resemble those of low-code. They create a development environment that can be used in real-time, and indicates what the output of the development will look like (Hurlburt, 2021). Through their graphical user interface (GUI), users can select, arrange, configure, and connect elements from built-in program libraries or third-party plugins (Lethbridge, 2021). This simplified way of working is similar to the way a child would assemble Lego bricks to build structures (Hurlburt, 2021). While this approach also applies to LCPs, Shah (2021) describes these Lego building blocks in the no-code domain as prefabricated components, similar to using PowerPoint. In LCPs, the building blocks can also be visualizations of functions to additionally meet specific business workflows or user-defined requirements. In addition, depending on the vendor, scripting languages can also be provided on the platform to extend the ability to develop all sorts of front-end as well as backend applications.

On the contrary, no-code platforms do not provide scripting languages and are thus limited to purpose-built backend applications. Examples of no-code platforms include Shopify-through, which companies can build online sales-and WordPress-through, which websites can be built (Lethbridge, 2021). Vincent et al. (2019) characterize no-code platforms as part of the LCP market due to the great similarities between the two technologies. The term is mainly used for marketing and positioning purposes (p. 2). For this reason, no-code aspects are also included in the following when using the term LCPs.

3. Method

To answer our research questions and identify opportunities and challenges of LCPs in SMEs, we rely on a qualitative research design using expert interviews (Babbie, 2015; Mayring, 2015), as theoretical research in this vein is still nascent. Thus, we seized the technique of semi-structured interviews (Longhurst, 2003) with practitioners from the field. We chose a semi-structured interview guideline to conduct the interviews leaving the interviewees enough room to express their own ideas.

Next, we purposefully selected the interviewees to achieve a high level of variation. To still maintain comparability between the interviews, the interviewees were selected from representatives of small and medium-sized enterprises who deployed and use(d) LCPs in their organizations. Thus, the experts were selected as a sample of convenience from the authors' professional networks, projects, or LinkedIn research. To ensure quality, each expert in our interview study had at least a few months of work experience at their organization. We contacted 63 potential interview partners, of which 44 % accepted to participate in our study. Thereby, in total, we interviewed twenty-eight experts over the timespan of three months (February 2022 up to the end of April 2022). The high participation is a first indicator of the importance of the research area and the interest of the enterprises.

The interviews were conducted via the video conferencing tools Zoom and MS Teams and lasted up to 108 minutes. We stopped conducting interviews when no new insights were revealed in the last interviews according to the theoretical saturation by Glaser and Strauss (2017). Each interview was recorded and transcribed using the qualitative data analysis software MaxQDA (2022). Transcripts were checked for correctness and manually corrected where needed. The interviews were coded independently by two of the authors, and the content was transferred into codes and subcodes. Using a consensus approach, differences in coding were discussed and resolved until a consensus view was reached.

In a last step, we translated the final coding into English while keeping the meaning because the interviews and analysis were conducted in German. The associated outcomes are described below.

4. Results

4.1. Opportunities

Link between IT and business departments. A great opportunity of LCPs is their ability to break down the often-existing tension between IT and business departments and, as a link, to bring these two areas closer together (INT 1, 26; INT 5, 30, 38; INT 19, 51). In the process, IT is shifting more toward the end-users, thus relinquishing classic development activities, while the business departments are becoming more and more IT-savvy (INT 4,21). This is primarily due to the fact that, ideally, the most technically skilled employees are used for low-code projects, and they automatically move closer to IT in a piecemeal fashion (INT 8, 31). This can break down the silo thinking of an organization and increase the quality of the end result (INT 28, 13). In addition, the platform itself has a supporting role in this. Communication between IT and business is often characterized by misunderstandings due to the different perspectives and approaches to solving problems (INT 5, 30). LCPs give both parties a common language through their visual expression, which is more comprehensible for both sides (INT 15, 30). Employees of the business department recognize significant challenges of the IT department's work - especially how difficult it is to formulate well-thought-out logic - through the possibility of being able to develop applications themselves with LCPs. As a result, the business unit's appreciation of IT increases, which fundamentally changes the way they interact with each other and has a positive effect on the corporate culture (INT 3, 61).

The traditional conservative method of programming applications would be unthinkable without well-trained IT personnel (INT 16, 52). LCPs, with their easy-to-understand interface, give companies the opportunity to significantly lower the barrier to entry for new IT forces (INT 4, 11). As a result, companies can draw from a significantly larger talent pool of technically suitable employees and take steps to overcome the problems of the growing IT skills shortage (INT 1, 26; INT 4,11; INT 5, 30; INT 6, 52; INT 7, 43, INT 13, 57;). Nevertheless, situations can also arise in which existing IT employees leave the company because they would rather program more exciting applications than simply assemble building blocks or maintain low-code applications (INT 5, 17; INT 7, 45).

The aforementioned expansion of the developer pool not only adds new developers; even the minimal saving of two to three developers can have drastic effects (INT 7, 38). Through LCPs, less demanding tasks can be outsourced to the business unit in order to be able to harvest so-called "low-hanging fruits," while the IT unit can once again focus more on its actual core topics (INT 2, 45; INT 4, 71; INT 5, 60; INT 7, 38). In addition to relieving the IT staff, many steps regarding the operation of the developed applications - they are operated via the cloud - or the security against hacker attacks - the security aspects are already analyzed by the

providers - can be outsourced to the platform (INT 6, 42). The IT area has the opportunity to leave the role of administrator taken in the past and to act as an innovator again increasingly (INT 9, 53).

Digitization and Modernization. Digitization is an important success factor that will be essential for companies from all sectors in the future (INT 5, 34; INT 6, 82). LCPs make it possible for companies of all types to tackle digitization, regardless of their budget, by simply developing applications (INT 5, 62) or IT competence can show them (INT 4, 27; INT 10,71). Through them, projects can be realized that would not have been started before (INT 3, 57). For example, it has even enabled entire legacy systems to be developed from scratch (INT 5, 56).

The use of LCPs makes it possible to automate all processes that, until recently, ran via e-mail or paper (INT 2,45). This can result in efficiency gains in the daily internal work process with significant positive effects in the longer term (INT 2, 45; INT 3, 125; INT 5, 56; INT 7, 92; INT 9, 81). In addition, employees can transfer their repetitive, dull tasks in the work process to applications and thus focus on more interesting, varied aspects of their work in the sense of job enrichment and work as knowledge workers in accordance with their human abilities (INT 8, 23).

In addition, LCPs also offer companies the opportunity to replace obsolete but nevertheless used programs such as Excel or Access, which are no longer being further developed by Microsoft (INT 12, 55). As a result, the unprofessional Excel sheets often found in departments, which have been built up over the years and in some cases have become business-relevant, can be replaced by reliable applications (INT 1, 42; INT 2, 45; INT 7, 43; INT 8, 15).

This digitization and automation of work processes can also have a positive effect on their quality. The standardization of work processes can eliminate sources of human error that occur, for example, when employees have to sort apart large data sets by hand (INT 8, 21). In addition, empowering employees in the departments can encourage them to rethink and reflect on their existing work processes and improve them in the spirit of BizOps (INT 4, 41; INT 18, 27). This is primarily due to the lack of a computer's ability to reflect, through which the employees be forced to constantly optimize their work processes in order to minimize errors in the application (INT 8, 21). Additionally, for the development of an application, a logic of the process must be formulated, whereby individual work steps are revised and new ones added or omitted (INT 13., 21). This documentation of the work steps performed automatically through code also makes it possible to better track processes in retrospect (INT 18, 23).

Higher Client Satisfaction. When working with LCPs, a role reversal occurs in the development project. The employees of the business departments are no longer the sole commissioners of an application but at the same time developers (INT 12, 33). This has major implications for the fulfillment of the requirements of business applications since the customers themselves know best what their applications must achieve (INT 2, 45; INT 6, 60; INT 7, 64). Additionally, the merging of the roles reduces the communication effort and, thus, the risk of misunderstandings (INT 3, 57). After completion of the applications, the customers are also significantly more satisfied since they were involved in the development themselves or carried it out constantly (INT 15, 23).

Speed. LCPs make it possible to deliver results significantly faster through their visual software development compared to classic development approaches (INT 4, 25; INT 6, 22; INT 10, 29). Nevertheless, while complying with all essential quality criteria, the factor of ten times faster, as promised by most providers, cannot be achieved. (INT 5, 30). However, the speed cannot be generalized but varies depending on the project situation and the precision of the formulation of the requirements (INT 6, 42). Nevertheless, LCPs also help to make requirements analysis more efficient (INT 17., 22). The integration of many approval workflows has an additional negative impact on speed (INT 23, 57).

Initial prototypes of an application that already meet the majority of the requirements can be developed within one day (INT 26, 22; INT 7, 10). This represents a new approach to application development. Results can be delivered quickly and can subsequently be subjected to a practical test; without spending much time on planning (INT 2, 21; INT 4, 11) and leading to an increase in the number of applications. However, the higher number of applications makes it necessary to filter out high-quality solutions that make up the minority (INT 22, 89).

LCPs offer the possibility to reuse artifacts developed from previous projects, which has major implications for the speed of development of future applications (INT 9, 31). Such artifacts can range from already created interfaces to other backend systems (INT 17, 7) to components that were developed by external parties specifically for the marketplace integrated by some providers (INT 15, 74).

The speed of development mentioned above is not limited to prototypes. Adaptations to the application can be made very quickly and easily; these could not be asked for in initial workshops due to unavailable data or the lack of anything tangible (INT 2,21; INT 6, 64; INT 9, 17). In small solutions, the citizen developers themselves can integrate their changes in a short time

(INT 3, 85). This rapid adaptability implies a better user experience (UX), as developers can quickly improve the user interface (UI), for example (INT 2, 57). In addition, LCP vendors are constantly working on making design elements easier to create and CSS classes easier to implement (INT 5, 48).

Typically, the end-users of the applications are often unable to express their wishes clearly and in a way that IT staff can understand. They often tend not to make any demands on the applications but simply to describe the work process because, for them, this automatically results in the requirements (INT 8, 13). However, since IT personnel need requirements in the style of "if-then-else thinking" in order to be able to formulate a suitable logic for the best possible implementation of the application, such assignments often fail due to communication (INT 11,13). By dealing with development, employees learn to express themselves more purposefully and to recognize which aspects of the desired application are important and which can be neglected (INT 6, 22).

Empowerment of Employees. LCPs enable employees throughout the company to actively promote digitization with minimal training and thus empower them to help shape the future (INT 4, 39). This is due to the simple way of developing applications, which can be compared to building with Lego bricks (INT 4, 13). In addition to this visual development, LCPs also greatly simplify technical aspects of programming. They provide users with the entire build pipeline, which can be used to work productively right from the start (INT 1, 14). In the background, they take care of most of the technical steps that need to be considered in traditional programming, allowing Citizen Developers to focus on the essentials, namely user stories (INT 15, 19). While the program code can be overwhelming for many newcomers (INT 14, 80), LCPs can be described as a cockpit, guiding contributors through the development process through best practices, feedback (INT 25, 19), or predefined templates (INT 10, 15).

This empowerment can trigger a real euphoria among employees of a company (INT 20, 29). Even those who were previously prevented from proactively shaping digitization in the company due to the high technical barrier to entry are given the opportunity, allowing the company to position itself much more broadly and to drive forward genuine democratization (INT 4, 25). This often results in a corporate cultural rethinking in which old barriers and boundaries are broken down, and imagination is given free rein (INT 24, 41).

The structure of development projects with LCPs also differs from conventional development. While flexibility is gained in the classic development model through agile project management methods, the dimension of flexibility that is possible with low code is not achieved (INT 6,52). In this context, the process of application development is clearly more iterative because initial prototypes are developed quickly, then tested in practice, and improved again with the feedback received (INT 8, 19). When external low-code developers are hired, dynamics often emerge in which the involvement of project management is not needed, and the project's contributors come together directly (INT 7, 60). This creates a new form of agile application development.

Competitiveness. An improvement in the competitiveness of companies is also seen in connection with LCPs. The above-mentioned increase in the talent pool of IT specialists not only counteracts the shortage of IT personnel. Independent development with low code in the specialist departments can promote the important digitization competence in companies in order to be able to position themselves for the future (INT 1, 52).

The rapid development and adaptability of applications mentioned above not only have a positive effect on meeting the requirements of such applications. LCPs allow companies to adapt quickly and in short development cycles to new needs from external stakeholders such as suppliers or customers (INT 1, 46; INT 4, 59; INT 9, 51). The above-mentioned possibility to develop prototypes quickly is also suitable for startups to validate business ideas quickly and without much effort on the market and to test whether there is sufficient demand (INT 1, 50).

With the help of LCPs, companies can adopt enterprise software to the business processes in the sense of BizOps (INT 4,45). Compared to the procurement of standard software such as ERP systems, the software can be tailored according to the process steps instead of vice versa (INT 1, 30). In the future, this may become relevant, especially for companies in niche markets where the agility of a company is essential since they cannot differentiate themselves from their competitors through standard software (INT 4, 57).

LCPs are tools that can be used to drive innovation (INT 1, 50; INT 4, 69). In the process, as mentioned above, the barrier to entry for Citizen Developers is drastically lowered. While there were already enough ideas for optimizing the work process without LCPs, they now enable employees to implement their ideas quickly and easily (INT 3, 117). This gives rise to a large number of new applications, most of which are not (yet) operational or useful (INT 2, 89). Among them, however, there are also pearls that need to be identified since they usually make a significant contribution to innovation (INT 12., 89).

4.2. Challenges

Dependency. A major challenge that arises in the course with LCPs is the dependency on the vendor (vendor lock-in) (INT 5, 32; INT 6, 40; INT 9, 41). This is mainly due to the fact that most LCP vendors do not offer automatic code generation for the customer (INT 16, 38). The code of the developed applications is now stored in an untraceable file format - like a black box where transparency is lacking - and is only interpreted in terms of a kind of runtime environment during development (INT 14, 22). In addition, the case can also occur when an LCP generates the source code but does not allow the deployment of the code outside the provider's environment (INT 9, 59). However, the future change of LCPs may become significantly more difficult due to changing platform requirements (INT 5, 32).

Due to the novelty of the LCP market, there is also the risk that vendors suddenly stop operating their platform, as was already the case with Google App Maker (INT 9, 57). In addition, this dependency creates the practical obligation to accept price increases by the provider simply - Atlassian, for example, drastically increased the price of their subscription - otherwise, all apps already developed would be lost (INT 26, 40).

Resistance. When deciding to implement LCPs in organizations, there may be resistance from the entire company. This is primarily due to a misconception in connection with LCPs (INT 7, 32). For example, low code is often perceived as a new technology that poses a threat to employees' jobs due to the accompanying increase in automation and digitization (INT 4, 37). Especially in this context, it is therefore essential to involve the works council of a company in good time and to convince it of the benefits of low code (INT 19, 69). Differences in speed and agility can also lead to discrepancies with the usual way of working (INT 27, 32).

In addition to these problems in mindset at the overall organizational level, it can be especially difficult to convince IT personnel of the concept of low code (INT 5, 17; INT 7, 45; INT 9, 49; INT 10, 105). This is because developers often see themselves as artists who paint their pictures themselves and do not want to have this painted automatically by a tool (INT 9, 49). This tool can give them additional maintenance tasks instead of developing and realizing new exciting ideas, which can make them feel underchallenged and subsequently leave the company (INT 27, 45). In addition, changes in salary and the anxiety that they have to use LCPs can also lead to dissatisfaction (INT 25, 17).

In the case of Citizen Developers, fear and excessive demands predominate at the beginning (INT 2, 81; INT 3, 43). When it comes to software development, they are often reminded of difficulties with mathematics from their school days (INT 3, 43). After the first points of contact and training, this initial respect can be overcome (INT 2, 81; INT 3, 43).

Limited Functionality. Another challenge that arises in connection with LCPs is the fact that the scope of application of LCPs is (still) limited. Thus, in addition to LCPs, there will still have to be classic development work in a company (INT 2, 101; INT 17, 39). At present, they are no longer suitable for a certain degree of complexity of the business processes to be mapped (INT 2, 49; INT 8, 33). For this reason, LCPs are more suitable for completed developments that do not require much further development (INT 2, 25; INT 10, 97). However, this is platform dependent. Other interviewees, however, describe LCPs as a powerful tool that can be used to develop solutions ranging from small automation applications to complex customer portals (INT 1, 16; INT 6, 40; INT 7, 94; INT 9, 91). However, it is recommended to always use LCPs in the main business area of a company in order to become more agile and faster in the core processes and subsequently to expand the USP (INT 6, 76). Furthermore, LCPs alone cannot solve problems; they are merely a suitable tool to drive innovation (INT 1, 50). In addition, a suitable plan is needed to master a company's challenges (INT 4, 69). A criteria catalog that divides use cases for applications into different categories and thus checks planned applications for their suitability with LCPs before development work begins can lead to more structure in this context and subsequently prevent errors (INT 23, 43).

Nevertheless, working with LCPs also remains basically the same: "data in, data process, data out or display" (INT 15, 32). This results in limitations in the platforms that must be taken into account in application development (INT 1, 22). However, solutions that work in classic development may not be suitable in part because they may not be consistent with the platform logic (INT 16, 22).

Shadow IT. Empowering employees from the specialist departments to develop their own solutions creates new problems for shadow IT. After all, departments could develop their own applications for every little thing, and in the end, it would no longer be possible to maintain an overview (INT 5, 36; INT 27, 39). In addition, synergies could be lost due to possible duplication of application development (INT 2, 21). However, the negative consequences of shadow IT can have even more serious consequences. In principle, applications must always go through various processes before they are operated in order to ensure security aspects or data protection conformity, which are usually not taken into account in the case of shadow IT (INT 25, 37). The emergence of shadow IT can partly be

attributed to the fact that employees are simply not aware of this concept (INT 5, 36). For example, they may unknowingly use personal employee data in applications that are not privacy compliant (INT 2, 37; INT 4, 28). A suitable measure to prevent problems in this regard could be to have citizen developers sign predefined terms of use before using the platform, to provide preventive education, and to define responsibilities (INT 3, 21). It is also important to define the rights of citizen developers on the platform, which can, for example, prevent access to certain data (INT 8,7). Increasing shadow IT can also make it difficult to achieve a uniform appearance in terms of corporate identity on the various applications (INT 2, 73). In addition to a uniform UI, the various applications should also be developed according to the same guidelines in order to be able to ensure a uniform style in programming (INT 8, 5). In addition, this may also make governance aspects relevant to guarantee the necessary future support of the software solutions (INT 1, 26; INT 4, 29, INT 10, 34). In any case, it is essential to establish a watchdog for low code in the company to bring a certain degree of control and governance to Citizen Development (INT 5, 36; INT 7, 64; INT 18, 5; INT 26, 28). If this is not adhered to, software proliferation can arise. If new solutions are constantly implemented without integrating them properly into the overall software of a company, problems such as the classic Excel sheet problem can arise: individual employees develop a software solution in the form of an Excel sheet that is expanded more and more over time until it finally reaches a high degree of complexity, making the solution no longer comprehensible for others (INT 1, 24; INT 17, 65; INT 28, 27). Depending on the size, these can become relevant software for employees of a company, which creates additional problems with regard to ensuring reliability as well as quality maintenance of the application (INT 3, 65).

Nevertheless, LCPs can also be a way to keep a company's shadow IT in check. For example, due to their cloud-based structure, they offer the possibility to view all developed applications, which provides an additional form of control (INT 3, 65; INT 8, 15).

Technical Debts. The aforementioned opportunity to develop applications faster can also have a negative impact on organizations. This speed means that fewer resources are used to adhere to programming best practices in addition to essential data protection, compliance, and governance rules, although quality can suffer as a result (INT 2,57; INT 5, 25; INT 8, 21). Thus, it is easy to save time and costs in the short term, but these have a negative impact in the long term, for example, in maintenance (INT 28, 27; INT 10, 63).

Regardless of attention to quality criteria, software maintenance issues can also arise because of the way LCPs generate code. When working with LCPs, new branches in particular - some of which have the same code - are added to the code instead of defining them by a higher-level object class (INT 3, 69). This goes against the established practice in software development to minimize redundancies in the code, which makes maintenance even more difficult because all redundancies have to be taken into account (INT 11, 69). In addition, for pro-code developers, changing the code of applications in productive use represents a greater risk due to the additional overhead than simply adding new branches to the existing code (INT 10,111). The result can be a landscape of enterprise software that creates many different data silos that cannot communicate with existing systems (INT 27, 45).

Testing of low-code applications is also made more difficult because unit tests are not possible, but the entire process must be run through in each case (INT 10, 45). However, the need to test the applications is minimized by the fact that all components of the modular system have already been tested by the provider beforehand and are thus already more error-resistant from the ground up compared to classic software development (INT 16, 42). Nevertheless, errors can creep in, which must be prevented by establishing suitable quality control processes in companies - such as conducting a separate technical review before applications are used productively (INT 3. 43).

User Interface. The intention to make applications developed with LCPs beautiful and easy to use can be an additional challenge (INT 1, 32; INT 5, 32; INT 7, 58; INT 10, 41). Low-code applications tend to have the same user interface (UI) (INT 5,32). The user interface (UI) of such an application can be distinguished from custom solutions with ease by a trained eye (INT 1, 32). This is due to the fact that the platforms, as mentioned above, are based on certain modularity. By adhering to this specific platform logic in the creation of the UI, significant time and personnel can be saved (INT 1, 18). Additionally, design guidelines can be a suitable measure to make applications consistent across the enterprise as well as to promote an efficient way of working (INT 9, 69; INT 10, 51). However, if special requirements are placed on the UI, this can only be achieved with considerable additional effort (INT 10, 41). Even for companies such as Google, which are known for their high-quality and easy-to-use products, familiarization with the LCP at hand is a complex matter (INT 7, 100). Also, companies working with LCPs tend to neglect the added value of good usability of the applications, which means that this topic often gets short shift in development (INT 5, 48). A simple method to counteract this could be to establish an additional role in the team of developers whose dedicated task is to optimize usability (INT 1, 30; INT 5, 50).

5. Discussion

A central promise of LCPs is the drastic increase in the speed of application development. Mendix and OutSystems, for example, assume a development time that is ten times faster than conventional software development (Mendix, 2022; OutSystems, 2022). It was also evident from the interviews that initial prototypes, which can already fulfill a large part of the applications, could be developed within one day. This speed can be associated with a significant loss of quality, which means that although fast results can be achieved in the short term, these will be relativized in the future by increased maintenance efforts. Due to the fast realization of results, applications can also be immediately adapted to customer needs, thus better meeting customer requirements.

LCPs lower the entry barrier for employees from specialist departments due to their higher level of abstraction in software programming (Hintsch et al., 2021). As a result, companies can access a larger talent pool of software developers and consequently counteract the shortage of IT specialists. In addition, LCPs offer, among other things, a common language between the IT and business departments and thus enable the exchange of knowledge across departments, which promotes innovation within the company (Iho et al., 2021). In practice, it can also be observed that citizen developers gain a better understanding of IT processes by working with LCPs, thus bringing the often diametrically opposed perspectives of IT staff and business unit employees closer together and subsequently preventing misunderstandings. This newly acquired IT knowledge often encourages employees from specialist departments to reflect on their work processes and improve them in line with the clear application logic. In theory, LCPs also take over essential work steps of full-stack development, bringing the logic of business processes to the fore in application development (Sahay et al., 2020). Successful coordination between IT and business can lead to citizen developers concentrating on the realization of technically less demanding work such as the automation of business processes - i.e., harvesting the so-called "low hanging fruit" - while IT is thereby relieved and can once again act more strongly as an innovation driver within the company (Woo, 2020).

Through their integrated functions, LCPs facilitate the use of proven development approaches of the agile project management approach such as Scrum, Kanban, etc. (Sahay et al., 2020) and make a significant contribution to the success of the company through this promotion of agility (Jesse, 2019). The significantly higher degree of flexibility and agility in the development project is also cited in practice. Thus, with the rapid development of prototypes and the rapid adaptation of applications, demand on the market can be tested quickly, and new needs of external stakeholders can be responded to in an agile manner. This can be essential, especially in niche markets, in order to stand out from competitors who may still be using standard software and are thus limited in their agility on the market. When it comes to the challenges, one of the most striking is their dependence on the platform provider. In most cases, applications developed using LCPs are only compatible with the platforms on which they were originally developed, which means that these applications can no longer be used if the provider ceases operations (Bock & Frank, 2021). This danger is also seen in practice. In most cases, the lack of automatic code generation and the inability to use the generated source code outside the provider's environment would result in dependency on the provider in terms of the aforementioned cessation of operations and price increases.

The opportunity offered by LCPs to enable employees in the business units to develop their own applications also brings negative consequences, such as the formation of shadow IT (Iho et al., 2021). This can ultimately lead to the violation of data protection and compliance regulations (Haan, 2020), as it can bypass IT as the controlling authority with regard to the testing and approval of the software (Sanchis et al., 2019). While in practice, the danger of shadow IT is also seen as a significant challenge, LCPs can contribute to the identification of enterprise software through their cloud-based structure in a way that would not be the case without LCPs - there was often talk of in-house solutions developed using Excel in the business units.

Nevertheless, the interviewed practitioners were able to identify technical deficiencies that result from programming with LCPs and subsequently lead to considerable additional effort for maintenance work. The lack of possibility to create superordinate object classes in the low-code area and the more difficult possibility of changing the code can also lead to the fact that new isolated solutions are always built, and no overall solution with good performance is developed. Thus, in addition to the technical complexity, the lack of possibility to apply conventional practices for documenting the code is an additional factor that increases the maintenance effort (Lethbridge, 2021). The lack of ability to test applications in a species-appropriate manner is mentioned in both theory (Jacinto et al., 2020) and practice. However, this is put into perspective insofar as it is emphasized that the components used in LCPs have already been tested in advance and are thus more error-resistant. Difficulties regarding the beautiful and independent design of the user interface as well as the development of a good user

experience could be attributed in practice to the special logic of how LCPs work.

In addition, the expert interviews identified further challenges that have not yet been examined in the literature. For example, the possible resistance of management due to a lack of openness to the necessary change, the resistance of IT specialists due to expected underperformance, and the resistance of citizen developers due to perceived initial excessive demands were mentioned. Thus, we formulate the following recommendations.

Recommendation 1: Do the work of persuasion.

Resistance can come from all sides within a company. In this context, it is important to convince the works council, IT staff, and citizen developers of the potential opportunities of low code before implementing LCPs. This can be achieved by actively involving them in the planning process and by creating initial points of contact with the topic.

Recommendation 2: Formulation of a plan for the functional area of LCPs.

LCPs are merely a tool for digitizing corporate processes. They alone cannot bring about the transformation of the digital age. Thus, a suitable plan is needed to solve specific problems of a company. While LCPs should always be used in the core processes of a company to promote the agility and competitiveness of the company, a criteria catalog can be a suitable measure for identifying relevant use cases for low code in a standardized manner.

Recommendation 3: Establish a Center of Excellence for LCPs.

The further development of enterprise software should be as planned and structured as possible. In order to counteract the uncontrolled growth of, in the worst case, incompatible stand-alone solutions, a central authority is required to make the citizen developers aware of their rights and obligations before they use the platform and thus hold them accountable. As the platform administrator, it can define the rights of different developers and thus counteract problems such as possible violations in the area of data privacy.

It should not see itself as a restrictive authority but, at the same time, promote the work of the citizen developers. Initial workshops can find new citizen developers to work with low-code and, at the same time, establish a low-code community within the company that can give developers the opportunity to network. Once a networking platform of this kind has been established, citizen developers can also exchange information and support each other in the event of ambiguities. In this context, these platforms should also offer training courses to familiarize citizen developers with the basics.

The aim is to familiarize citizens with programming and the use of the low-code platform. Depending on the size and resource allocation of the company, concepts such as a buddy system can be offered in which citizen developers are supported in the development process by an IT specialist in weekly meetings, for example.

6. Conclusion, Limitations, and Future Research

The purpose of this paper is to demonstrate the opportunities and challenges of low-code platforms in companies and to derive possible recommendations for action in dealing with these platforms. The results show that, under certain conditions, LCPs are certainly suitable for increasing the agility and innovative power of companies through the rapid automation of business processes with faster development times and even partially counteracting the shortage of IT specialists due to the lower requirements for previous IT knowledge, since employees from specialist departments can make significant contributions to IT development work. Challenges such as the increased dependency on platform operators or the dangers of the formation of shadow IT areas or even real damage to the IT structure, as well as the non-observance of legal requirements and governance aspects due to the low level of IT knowledge of LCP users, can very well be mastered by taking appropriate steps. The implementation of a center of excellence for LCPs as a central authority in the company can be recommended for training, further education, the creation of framework agreements and guidelines for the further development and testing of the IT systems concerned, but also for the networking of employees. The formulation of a plan for the specific areas of application of LCPs counteracts the feared proliferation of isolated solutions. Furthermore, it seems highly advisable to get management and, if necessary, the works council as well as interested employees on board in good time to be able to counteract in good time the aspects not yet described in the relevant literature, such as possible resistance on the part of management to necessary changes in connection with the use of LCPs and perceived initial excessive demands on the part of citizen developers.

In this paper, we conducted interviews to analyze opportunities and challenges in the use of LCP. The main results yield a wide variety of application fields and scenarios where LCP plays a significant role in practice. Yet, we only referred to practical insights. Future research could analyze the literature on LCP and compare it with our results.

Since the low code paradigm is a rather new phenomenon, practitioners rather view low code as a key solution for business problems, e.g., for tackling

new requirements during digital transformation processes. We derived three recommendations that can be tested in experimental settings or case studies. Future research could look at the possible effects of using LCP on the employees and the organization as well as on customer satisfaction.

References

Alt, R., Leimeister, J. M., Priemuth, T., Sachse, S., Urbach, N., & Wunderlich, N. (2020). Software-defined business. *BISE*, *62*(6), 609–621.

Andreessen, M. (2011). Why Software Is Eating the World. *Wall Street Journal*.

Babbie, E. R. (2015). The practice of social research: Nelson Education. *Boston, MA Cengage Learning. USA*.

Baran, B. E., & Woznyj, H. M. (2020). Managing VUCA: The human dynamics of agility. *Organizational Dynamics*.

Bexiga, M., Garbatov, S., & Seco, J. C. (2020). Closing the gap between designers and developers in a low code ecosystem. *Proceedings of the 23rd ACM/IEEE ICMDELS: Companion Proceedings*, 1–10.

Bock, A. C., & Frank, U. (2021). Low-Code Platform. *BISE*, *63*(6), 733–740.

Brinker, S. (2018, May 29). *Democratizing martech: Distributing power from IT to marketing technologists to everyone*. Chief Marketing Technologist.https://chiefmartec.com/2018/05/democratizing-martech-marketing-technologists/

Charette, R. N. (2005). Why software fails [software failure]. *IEEE Spectrum*, *42*(9), 42–49.

Fokaefs, M., Barna, C., & Litoiu, M. (2017). From DevOps to BizOps: Economic sustainability for scalable cloud applications. *ACM TAAS*, *12*(4), 1–29.

Gartner Inc. (2022). *Enterprise LCAP (Low-Code Application Platforms) Reviews 2022 | Gartner Peer Insights*. Gartner.

Glaser, B. G., & Strauss, A. L. (2017). *Discovery of grounded theory: Strategies for qualitative research*. Routledge.

Haan, J. D. (2020). *Council Post: Bring Shadow IT Into The Light And Capitalize On Citizen Developers*. Forbes.

Hintsch, J., Staegemann, D., Volk, M., & Turowski, K. (2021). *Low-code Development Platform Usage: Towards Bringing Citizen Development and Enterprise IT into Harmony*.

Hurlburt, G. F. (2021). Low-Code, No-Code, What's Under the Hood? *IT Professional*, *23*(6), 4–7.

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

Jacinto, A., Lourenço, M., & Ferreira, C. (2020). Test mocks for low-code applications built with OutSystems. *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*, 1–5.

Jesse, N. (2019). *Agility eats Legacy–the Long Good-bye*.

Koksal, I. (2020). *The Rise Of Low-Code App Development*. Forbes.

Lethbridge, T. C. (2021). Low-code is often high-code, so we must design low-code platforms to enable proper software engineering. *International Symposium on Leveraging Applications of Formal Methods*, 202–212.

Longhurst, R. (2003). Semi-structured interviews and focus groups. *Key Methods in Geography*, *3*(2), 143–156.

Lugovsky, V. (2021). *Council Post: A Guide To Low-Code/No-Code Development Platforms In 2021*. Forbes.

Mayring, P. (2015). *Qualitative Inhaltsanalyse: Grundlagen und Techniken*. Beltz Verlag.

Mendix. (2022). *Pandemie sorgt für gravierend beschleunigte Digitalisierung und großen Software-Bedarf—Low-CodeTechnologie ermöglicht schnelle Umsetzung*. Mendix.

OutSystems. (2022). *Accelerate Your Customer Experience Transformation*. https://www.outsystems.com/use-cases/customer-digital-experiences/

Richardson, C., Rymer, J. R., Mines, C., Cullen, A., & Whittaker, D. (2014). New development platforms emerge for customer-facing applications. *Forrester: Cambridge, MA, USA*, *15*.

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

Sanchis, R., García-Perales, Ó., Fraile, F., & Poler, R. (2019). Low-code as enabler of digital transformation in manufacturing industry. *Applied Sciences*, *10*(1), 12.

Tariq, H. (2021). *Council Post: Low-Code Versus No-Code And The Future Of Application Development*. Forbes.

Troise, C., Corvello, V., Ghobadian, A., & O'Regan, N. (2022). How can SMEs successfully navigate VUCA environment: The role of agility in the digital transformation era. *Technological Forecasting and Social Change*, *174*, 121227.

Vincent, P., Iijima, K., Driver, M., Wong, J., & Natis, Y. (2019). Magic quadrant for enterprise low-code application platforms. *Gartner Report*.

Wang, Y., Feng, Y., Zhang, M., & Sun, P. (2021). The Necessity of Low-code Engineering for Industrial Software Development: A Case Study and Reflections. *2021 IEEE ISSREW*, 415–420.

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

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