August 2026

The Missing Link in ERP Customization: Integrating Low-Code Platforms and Business Process Libraries

Adrian Abendroth
University of Potsdam, adrian.abendroth@wi.uni-potsdam.de

Helena Schulze Neuhoff
University of Potsdam, helena.sn@protonmail.com

Christopher Schilling
University of Potsdam, christopher.schilling@uni-potsdam.de

The Missing Link in ERP Customization: Integrating Low-Code Platforms and Business Process Libraries

Full Paper

Adrian Abendroth
University of Potsdam
adrian.abendroth@wi.uni-potsdam.de

Helena Schulze Neuhoff
University of Potsdam
helena.schulze-neuhoff@uni-potsdam.de

Christopher Schilling
University of Potsdam
christopher.schilling@uni-potsdam.de

Abstract

Enterprise Resource Planning (ERP) systems provide organizational stability but often lack the flexibility needed for specific process adaptations. While Low-Code platforms (LCP) enable domain experts to design extensions, the integration of business process libraries (BPLs) with Enterprise Systems (ES) remains under-researched. This paper investigates the link between ERP customization, LCP, and BPL through 20 qualitative interviews with process experts and consultants. The findings reveal a dual capability core where visual modeling lowers entry barriers for business users, while conventional code remains necessary for complex logic. The study identifies integration challenges regarding API stability and transparency, alongside a five-role socio-technical governance model for maintainable, decentralized adaptations. By conceptualizing BPL as lifecycle-oriented artifacts bridging modeling and execution, this research contributes to ES and LCP literature with actionable guidance for scalable, governable process customization. Future research should explore prototypical implementations, AI-driven process creation, and long-term governance impacts.

Keywords

Enterprise Resource Planning (ERP) Systems, Low-Code Platforms, Business Process Libraries, Integration

Introduction

Enterprise systems (ES), particularly Enterprise Resource Planning (ERP) systems, form the backbone of digitally managed organizations, as they control core business processes (BP) such as lead management, procurement, and project planning while standardizing information flows (Barna et al., 2021). These BPs are often based on industry-wide standards that come preconfigured within the ES, allowing only limited adaptation to company-specific requirements (Abendroth et al., 2024). Such adaptations typically require advanced programming skills, making them costly and time-consuming (Haddara et al., 2022). As the demand for more flexible BPs increases and digital transformation accelerates, these limitations have become a critical concern for maintaining competitiveness (Gronau, 2026). Organizations are under pressure to adapt processes quickly and efficiently without incurring high development costs. In addition to challenges such as skill shortages and adaptation expenses, the recurring need to reimplement customizations after system updates represents a major issue (Bender & Korjahn, 2023).

To address these challenges, many organizations aim to empower internal domain experts with process knowledge to participate more actively in shaping and customizing BPs in ES. This approach promises

faster, more cost-effective, and needs-oriented process evolution. In this context, low-code platforms (LCPs) are gaining greater significance. They can increase the flexibility of ERP customization and reduce dependency on traditional programming (Abendroth & Bender, 2025). Using visual and declarative tools, business users can develop applications and extensions for specific BPs, ranging from simple dashboards to complex workflows (Alamin et al., 2023; Bies et al., 2022), without deep coding knowledge (Lourenço et al., 2023). Especially for legacy ERP systems with limited flexibility, external LCPs such as Mendix or Appian offer new integration possibilities (Gode et al., 2023). Through API interfaces, available connectors, or metamodels, ERP functionalities can be extended or supplemented (Picek, 2023).

The same principle can be applied to BPs: business users can remove, adjust, or add process steps to create customized process variants (Picek, 2023). While some modern ERP systems, such as SAP in combination with Signavio (SAP SE, 2026b), already provide integrated workflow management systems (WfMS) that support editable processes from business process library (BPL), such capabilities remain the exception rather than the rule. A structured BPL constitutes a key success factor, as it provides predefined, reusable building blocks for BPs. Through standardized modeling, PBLs support consistent implementations, reduce isolated solutions, and enable more effective governance of process customization across the organization, thereby lowering effort and costs (Picek, 2023; Tang, 2022).Whereas BP customizations historically had to be implemented deep within the system core (Abendroth et al., 2024), the combination of LCPs and BPLs may enables flexible, user-driven adjustments. In particular, low-code based BPLs can simplify integration with ES, provide clearer architectural separation, and make process customization more accessible to citizen developers while maintaining control and transparency.

Despite the significant potential of combining LCP and BPLs for scalable integration, improved process governance, and user-friendly customization, this approach remains largely unexplored in research and practice. Technical requirements, organizational implications, and viable integration options are still insufficiently understood. Accordingly, this study investigates how BPLs, can be more effectively integrated into ES like ERP to enable BP customization through LCP. This study addresses the following research questions (RQ):

- **RQ1:** What are functions, integration options, challenges in integration, entry barriers and implementation prerequisites when integrating BPLs with ES like ERP systems using LCP?
- **RQ2:** How can a low-code-based BPL be designed for ES integration, and what technical implementation use cases result from it?

To explore these questions, we employ a qualitative research approach, conducting interviews with process experts, LCP users, and LCP consulting firms specializing in ES integration. By empirically examining the requirements, challenges, and design options for integrating BP libraries through LCP, this study aims to contribute by clarifying of how process modularization can enhance the flexibility and adaptability of ES.

Background

BPs vary significantly in strategic value. While support and administrative tasks such as payroll allow for high standardization, value-adding processes require extreme flexibility to secure market advantages (Balint, 2017). Standard ES, however, are typically built on industry "best practices" that fail to capture these competitive nuances. When organizations attempt to bridge this gap through traditional customization, they encounter prohibitive costs (Rothenberger & Srite, 2009), long development cycles (Benders et al., 2006) and vendor-imposed constrains (Parthasarathy & Sharma, 2017). This technical rigidity often forces a retreat into either fragmented ES landscapes or manual workarounds (Irnawati & Rahayu, 2025) because employees must constantly switch between tasks and systems, which in turn increases cognitive switching costs and undermines efficiency (Suija-Markova et al., 2020).

This limitation is best understood through the lens of Model-Driven Engineering (MDE), a paradigm where models are treated as primary, machine-processable artifacts that directly drive the engineering process rather than merely supporting it (Cabot, 2020) . In an MDE approach, high-level models (e.g. diagrams) are automatically transformed into executable software or system configurations (Hailpern & Tarr, 2006). While modeling frameworks provide a structured foundation for organizational analysis, they primarily operate as descriptive instruments rather than for operational change.

The Open Group Architectural Framework (TOGAF) for Enterprise Architecture Management (EAM) provides a robust industry-standard methodology for aligning IT strategy with business goals through its Architecture Development Method (The Open Group, 2018). While it is the foundation for SAP's Enterprise Architecture Framework (SAP SE, 2026a), its best practice approach is designed for holistic landscape management, rather than rapid technical changes to business logic. While TOGAF ensures a structural foundation, it is often too complex to provide the technical agility needed for fast, model-driven updates.

The Architecture of Integrated Information Systems (ARIS), is a process-oriented management approach designed to reduce enterprise complexity through a holistic view of design and execution (Scheer & Nüttgens, 2000). ARIS deconstructs the enterprise into five distinct perspectives: organization, data, process (control), function, and product/service, which are combined as a matrix across three lifecycle layers: the business concept layer, design specification layer, and the implementation layer (Palleduhn & Neuendorf, 2013; Pesme, 2023). However, ARIS is fundamentally static. While it provides a robust PBL of linked models and supports MDE approach, its implementation layer remains largely disconnected from the actual runtime environment of (legacy) ES. Furthermore, its high cost and complexity make it impractical for most small and medium companies (SME). Consequently, ARIS excels at documentation and process mining, but it lacks a technical "bridge" to the execution layer. It can define what a process should look like, but it cannot facilitate the automated or low-code adaptation of the underlying ERP system based on those models. This creates a **modeling-to-execution** gap that prevents domain experts from translating process insights directly into functional system updates.

Current tools using frameworks like ARIS and TOGAF attempts to bridge the gap between business processes and services, seeking to translate conceptual models into executable logic (Bizagi, 2026; SAG Aris GmbH, 2026; Sparx Systems, 2026; Visual Paradigm, 2026). However, even within advanced tools, which incorporate Service-Oriented Architecture (SOA) and provide APIs for microservices, a significant technical chasm remains. While these technologies offer the necessary building blocks for modularity, their orchestration and integration typically require professional developers. Consequently, when a business process requires modification, the programming effort needed to adapt the ERP core or synchronize the microservice layer remains prohibitive for non-technical staff. What is still missing is a direct technical bridge that allows process models themselves to act as executable interfaces for system behavior.

Such a bridge could be realized through the combination of microservice based LCPs and structured BPLs, enabling process models to serve simultaneously as conceptual abstractions and technical triggers for system adaptation (Abendroth et al., 2024; Abendroth & Bender, 2025). By leveraging BPLs as reusable and governed building blocks, this approach promises not only improved integration and flexibility, but stronger governance and traceability of process changes. At the same time, it lowers the barrier for citizen developers allowing process adaptations to be implemented directly from modeling constructs, while ensuring that system changes remain documented by design and processes are technically actionable.

Methodology

To address the research questions concerning functions, integration options, challenges in integration, entry barriers and implementation prerequisites, and use cases, we conducted a qualitative, empirical interview study, applying content analysis techniques in accordance with Mayring (2020). Structured content analysis was used to systematize findings, identify patterns and derive categories from the interview material. The next section describes data collection and category building.

Data Collection and Sample: To ensure comprehensive insights, two key stakeholder groups were selected: LCP users with process knowledge, and developers. We developed slightly adapted questionnaire guidelines for each group to capture their unique perspectives. These guidelines were designed using the methodologies of Faulbaum et al. (2009) and Mayring (2020), and cover five high-level categories: (i) functions; (ii) integration options; (iii) challenges in integration; (iv) entry barriers; and (v) implementation prerequisites. Using a semi-structured interview approach (Gläser & Laudel, 2010), we allowed interviewers flexibility to explore emerging themes while maintaining a structured framework (Döring & Bortz, 2016). Given the lack of prior empirical research in this area, 20 in-depth interviews were conducted in mid 2025, providing initial insights into the RQs. Interviews lasted an average of 44 minutes and included participants from various industries and organizational roles (see Table 1).

ID	Type	Industry	Position	ID	Type	Industry	Position
I1	Dev	Packaging	CEO/Developer	I11	User	Metal industry	Inventory management
I2	User	HR/Consulting	Data analyst	I12	Dev	IT-Consulting	Solution architect
I3	User	University	Lecturer	I13	User	IT-Consulting	CTO
I4	User	Trade association	Project management	I14	Dev	Automobile	IT-architect
I5	User	University	Lecturer	I15	Dev	Energy	LCP developer
I6	User	Media	Business analyst	I16	User	Energy	Business analyst
I7	User	B2B-Software	Business analyst	I17	User	IT	CEO
I8	User	Job placement	User	I18	Dev	Industry	LCP engineer
I9	Dev	IT-Consulting	Developer	I19	User	Health	Security operations center specialist
I10	User	IT-Sales	Business analyst	I20	User	IT-Consulting	Business analyst

Table 1. Overview of interview partners

Category building: Following Mayring's (2020) approach to qualitative content analysis, we employed structuring as the primary analytical technique. This method extracts a framework from transcribed interview material to categorize and synthesize key themes. A coding guide was developed to ensure consistency, containing definitions, anchor examples, and coding rules (Mayring, 2020). Initially, the transcribed content was coded based on the five predefined categories. As the analysis progressed, new subcategories emerged inductively, refining the coding framework through an iterative process to enhance intercoder reliability. An example of the developed coding guidelines is provided in Table 2.

Subcategory	Description	Coding rule	Anchor example
Integration	It encompasses the connection and interaction between LCP, ERP systems, and other internal or external applications. This includes data exchange, process orchestration, interface design, and the technical or organizational effort required to ensure seamless interoperability across systems. Integration is relevant because it enables end to end processes, avoids data silos, and supports the coordinated use of heterogeneous system landscapes.	Statements are coded when interviewees explicitly refer to integrating systems, applications, data sources, or processes, including mentions of interfaces, APIs, middleware, or synchronization mechanisms. General statements about system usage or functionality that do not address cross system interaction or interoperability are not coded.	"Yes, I think that integration capabilities, i.e., the connectors, the pre-built integration modules that come with such a platform, I think that things stand or fall with that, i.e., the capabilities, the capabilities in the area of integration, I believe, are pretty crucial" [I5]. (Translated from German)
Documentation	It refers to the creation, maintenance, and use of written or digital materials that describe	Statements are coded when interviewees explicitly mention the need for	"I think I would put documentation at the top of the list. I

	processes, configurations, functionalities, and changes within LCP and related systems. This includes technical documentation, user guides, process descriptions, and change logs that support transparency, knowledge transfer, and sustainable system use over time. Documentation is considered essential because it enables understanding across different stakeholder groups and supports maintenance, onboarding, and further development.	documenting processes, applications, configurations, or changes, or when they refer to documentation as a support mechanism for understanding, communication, training, or maintenance. General references to knowledge sharing, informal communication, or personal experience without a clear link to documented artifacts are not coded.	believe that when you look at the entire IT world as a whole, documentation is always a very important factor, which various platforms sometimes manage well or very poorly“ [I8]. (Translated from German)

Table 2. Exemplary coding guidelines, category “Technical challenges in integration”

Findings

Our analysis of the **functions category** highlights several core clusters necessary for an integrated BPL. **Integration** and **connectors** emerged as the primary capability, specifically regarding prebuilt building blocks for accessing ERP and legacy data [I5-I8, I10-I12, I14, I17-I20]. Experts identified these as critical success factors, noting that specialized integration skills remain "crucial" for efficient use case realization [I5]. A second cluster concerns **usability** of the platform and the BPL’s modeling environment [I5-I11, I20]. Experts emphasize that an intuitive and visually appealing interface lowers entry barriers and enables non-IT users to build something up in a short time, accepting that not every solution will be completely customized [I10]. Despite the low-code paradigm, the possibility to embed **conventional code** (e.g., Javascript, Java, Python) is regarded as an important functional extension [I1-2, I5, I7-10]. Interviewees report that simple cases are efficiently handled via low-code, but complex scenarios are often “faster “and “cleaner” solved by integrating custom scripts to overcome platform limitations [I7]. Furthermore, **documentation and governance functions**, including logging and debugging, are essential for transparency and maintainability [I1-I2, I5, I7-I10, I16]. In addition, governance-oriented functions such as platform management features and role concepts support centralized control over process lifecycle and platform usage [I3, I5, I9-I11, I18]. Finally, **predefined components** and **templates** are seen as key for fostering standardization and reducing implementation effort through reuse [I4-I7, I10-I12, I15]. Interviewees describe that once a base app or process structure exists, it can be reused across further applications, significantly reducing implementation effort [I10].

Regarding the **category integration** options, **REST APIs** are the most frequently named integration mechanism for linking the BPL to ERP and surrounding systems [I1-I2, I12-I13, I15, I17-I19]. SOAP interfaces remain relevant, especially in established ERP and legacy environments where service contracts are already standardized [I12-I13, I18]. **CSV and XML** exchange are still common options, for example in batch-oriented integrations or where APIs are not available or economically feasible [I4, I13-I15, I20]. These formats serve as pragmatic fallbacks, particularly when connecting older ERP modules or third-party applications without modern service interfaces. Several interviewees refer to dedicated **connectors**, including those for major ERP vendors and for the Microsoft suite [I3-I6, I9, I11]. Such connectors encapsulate technical complexity but are sometimes criticized because they either only partially work in practice or require additional “Premium” licenses, especially for SAP or other specialized systems [I6]. This duality makes connectors both an enabling option and a potential cost driver for ERP-related integrations.

Regarding the **challenges in integration,** a dominant theme is the lack of knowledge and the big learning curve, both for LCP and for the underlying business processes [I4-I10, I13-I17, I19-I20]. Interviewees observe that users often start automating without first clarifying the actual process to be implemented, which leads to suboptimal designs and rework [I6]. From a technical standpoint, integration is challenged by incomplete or **unstable connectors**, missing standard interfaces in ERP systems, and the need to bridge heterogeneous data models [I4, I6, I9-I13, I15-I16, I19-I20]. While “there are connectors

for everything" in theory, in practice they frequently work only "halfway" or require additional manual configuration and troubleshooting [I6]. **Entry barriers** include license costs of LCPs, which are perceived fundamentally different from classical development and can become problematic when priced per user and month [I5-I7, I9, I13-I16, I18-I20]. Interviewees point out that low-code makes economic sense when many use cases are realized and costs can be distributed, but is expensive if adoption remains limited [I9, I13].

Within the **prerequisites category**, requirements are closely intertwined with governance and security protocols, particularly in scenarios where external stakeholders, e.g. customers or suppliers, interact with processes that interface directly with sensitive ERP data. Organizations must ensure data security compliant processing and require providers to demonstrate adequate data security and privacy certifications, with the exact requirements depending on the type and sensitivity of the processed data [I2-I4, I6, I9-I12, I14]. This necessitates a robust role- and rights concept within the BPL, including clear responsibilities for approving and maintaining integrated processes [I4-I5, I10-I12, I18, I20].

Design of a low-code based BPL: The library is expected to offer processes "outside the ERP standard", such as supplier portals, invoice processing or onboarding, exposed via an easy-to-use user interface. It should allow workflows to be fully modeled and executed within the low-code environment, particularly for approval processes, web-based interactions and customer-specific variants, as well as scenarios "when interfaces to the outside are needed" and data must be pulled from third-party systems [I1-I2, I5, I8, I11, I13, I15, I17]. At the same time, interviewees consistently emphasize that such flexibility must be tightly coupled with **governance** mechanisms to avoid uncontrolled process proliferation and inconsistent ERP adaptations. Accordingly, the design of the BPL is explicitly embedded within a governance model that defines ownership, responsibilities, and decision rights for ERP related processes and low-code based adaptations. A clear roles and rights concept is required, covering internal and external users as well as platform-level responsibilities MDE-based metamodels enable platform-independent process governance and adaptation. Functional responsibility for the library and its processes should be assigned to a product owner or business unit, supported by defined approval workflows for process changes.

Based on the interviews, an **organizational usage framework** in the form of use case diagram was derived with the following main roles and functions (Figure 1): (i) business department (product owner) orchestrating the lifecycle by requesting and adapting processes using predefined building blocks and commission process changes; (ii) citizen developer acting as the technical anchor, responsible for creating, monitoring, and maintaining the underlying processes and its documentation; (iii) internal user executing processes and providing process feedback for optimization; (iv) approving authorities serving as a vital regulatory gatekeeper, reviewing and approving process changes before productive use; and (v) external user, initiating processes (e.g. via portals) and viewing the status of their external processes.

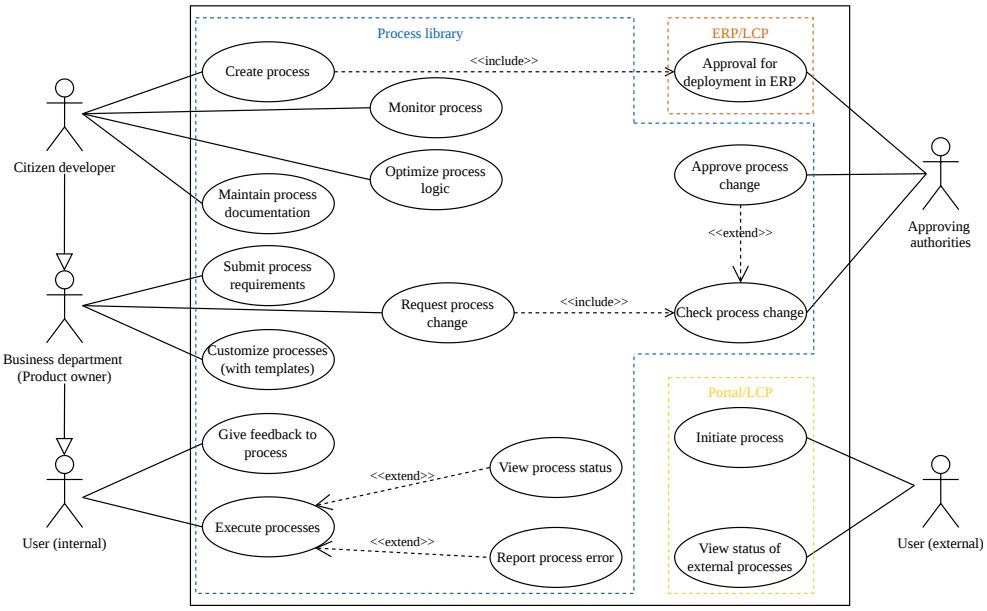

Figure 1. Use cases of BPL based on interviews

At **process level**, the interviews highlight several design requirements for workflows in the library. Reusability of processes and process fragments is a central requirement, enabling the use of templates for similar use cases across departments or customers [I1, I3, I10-I11, I15]. Processes must be adaptable, with the possibility to adjust logic and parameters over time [I2-I3, I6-I8, I11-I12]. At the same time, some experts stress that low-code/no-code should still work with as little custom code as possible and only allow code access where necessary [I5, I9-I10]. Advanced features discussed by the experts include an optional process marketplace and AI support. A marketplace could facilitate sharing and reuse of process templates, although some organizations are reluctant to publish their "best" business processes [I3-I6, I8, I12]. AI is seen to prompt and create processes as well as make adaptations to processes, adaptation and recommendation of optimization options [I4-5, I12]. After the expert interviews, a prototype using Mendix (LCP) and ERPNext(ERP) was built to validate the framework's core logic, confirming the technical feasibility of the integration approach, while a full evaluation will follow in a future study.

Discussion

This research investigated the Requirement Engineering (RE) landscape for integrated BPLs within LCPs, specifically targeting processes that reside outside the ERP standard, such as supplier portals and onboarding workflows. The findings reveal that a successful BPL is built on a dual-capability functional core: While usability and visual modeling lower the entry barrier for non-IT users, the ability to embed conventional code (e.g., JavaScript or Java) remains a non-negotiable requirement for handling complex business logic that exceeds platform limitations. Furthermore, we found that while REST and SOAP are the preferred integration standards for connecting to ERP systems, pragmatic fallbacks like CSV and XML exchange remain essential for legacy environments or where modern APIs are not economically feasible. Beyond technical functions, our results highlight a critical socio-technical dimension to BPL design. We identified a five-role ecosystem, consisting of the product owner, citizen developer, internal user, approving authority, and external user, that necessitates a robust governance model. A central finding is that integration is often the "Achilles' heel" of LCP adoption: experts noted that while "connectors for everything" exist in theory, they frequently work only "halfway" in practice or introduce significant cost drivers through premium licensing. Consequently, the design requirements for a BPL must prioritize documentation and "Compliance by Design" (e.g., LDAP integration and DSGVO-compliant processing) to ensure that decentralized process adaptation does not compromise system integrity or security.

Comparison with Commercial Solutions: The proposed framework fills a gap between expert-level architecture tools and BP suites. Classical modeling tools like Enterprise Architect (Sparx Systems, 2026) allow for thorough documentation but lack the collaboration features needed for modern teams. Similarly, Visual Paradigm (2026) serves primarily as an expert-focused tool for modeling frameworks like ARIS or TOGAF. In the broader BP suite market, high-end solutions like ARIS Process Intelligence (Software AG, 2024) and SAP Signavio (SAP SE, 2026b) provide robust governance, lifecycle management, automation solutions and ERP connectors, yet their high cost and complexity often exclude SMEs. Meanwhile, BP automation-focused tools like Camunda (2026) excel in RPA, and Bizagi (2026) offers a strong low-code approach for rapid development (Thalhofer & Krumm, 2024).

Implications for Practice: *For companies* and users of LCP, the creation of robust governance structures is essential to maximize LCP success (Viljoen et al., 2024). Managers could implement an Integration Platform as a Service (IPaaS)-driven data hub strategy to serve as a stable foundation for the BPL, ensuring that all necessary data is accessible for process optimization without bypassing security protocols. Also, decision-makers must critically evaluate how a LCPs underlying metamodel architecturally supports process logic, whether through microflows, rule-based engines, or BPM, while remaining cognizant of the high risk of vendor lock-in resulting from the limited export and import functionalities currently offered by many LCP vendors (Alfonso et al., 2025). Governance must be granular, clearly defining roles such as citizen developers who serve as technical anchors, and Product Owner who validate logic within departmental silos. Furthermore, organizations should utilize the BPL to establish internal benchmarks and marketplaces. By sharing non-competitive, industry-standard process templates (e.g., via a marketplace), firms could foster standardization and reduce implementation efforts (Abendroth & Bender, 2024, 2025). Licensing and security must be addressed via "Compliance by Design" using role concepts to manage access for external stakeholders like suppliers without compromising the ERP core.

For *LCP vendors*, there is a significant opportunity to gain competitive advantage by aligning internal metamodels with industry standards like BPMN 2.0 to facilitate professional documentation and direct system integration. Vendors should prioritize the development of external user portals and native SOA/microservices support, allowing communication to flow directly into the application. To lower the steep learning curve, vendors must enhance usability through AI-driven visual prompting and provide multi-tiered training resources (e.g., interactive manuals and online certifications). Finally, improving the transparency and stability of connectors is vital. Vendors who provide reliable, easily configurable integration building blocks will see higher adoption rates compared to those offering harder to implement solutions that require extensive manual troubleshooting.

Theoretical Contributions: First, our research establishes that because LCP development mirrors the paradigm of MDE, the structural requirements for building integrated BPLs can be directly derived from MDE principles (Di Ruscio et al., 2022). We contribute to the literature by demonstrating that the structure of these libraries is decisive for their success, especially for the software development lifecycle. This theoretical shift moves the focus from short-term app development to a long-term, lifecycle-oriented design approach. Building on this MDE perspective, our work highlights a critical theoretical gap concerning the interoperability of process modeling standards. While professional process management relies on standards like BPMN 2.0, most LCPs utilize proprietary visual notations that often lack robust import/export capabilities (Alfonso et al., 2025). Our findings suggest that future research must prioritize the development of transformation logic between these layers to prevent the long-term risk of vendor lock-in. By highlighting this disconnect, we provide a foundation for researchers to explore cross-platform integrity, effectively bridging the divide between formal BPM theory and the practical realities of citizen development. Finally, our research challenges the traditional binary view of IT against business in RE by providing a taxonomy of governance roles. By identifying specialized functions, such as citizen developers as “Power Users” who serve as technical anchors and coordinators who harmonize departmental needs, we offer a framework for studying the social dynamics of LCP ecosystems. Finally, our research extends RE and BPM literature by proposing a specialized taxonomy of roles for decentralized environments. Traditional BPM frameworks often rely on a rigid separation between the Process Owner (business) and the IT Architect (technical implementation). Our model challenges this binary by introducing the citizen Developer as a "technical anchor" and the approving authority as a necessary regulatory gatekeepers.

Limitations: This study is based on qualitative data from 20 interviews across diverse organizations. However, it is subject to single-respondent bias, as each company was represented by only one expert. Consequently, the findings reflect individual professional perspectives rather than a multi-perspective organizational consensus. Furthermore, the sample is primarily concentrated in the DACH region, which may reflect specific local regulatory environments and BPM maturity levels, limiting broader generalizability. Additionally, the rapid evolution of LCP ecosystems may shift functional requirements faster than academic documentation can track. Finally, the research is conceptual and lacks a technical validation of the proposed metamodel structures.

Future research should focus on the prototypical implementation of these requirements, utilizing standards like the OMG’s Meta-Object Facility or BPMN 2.0 to bridge the gap between theory and practice. Quantitative studies could further assess the impact of integrated BPLs on efficiency and long-term maintainability. Another vital avenue is the analysis of AI-supported prompting, investigating how automated process creation changes the RE process for citizen developers. Longitudinal case studies are needed to evaluate the technical debt of decentralized BPLs and to refine governance frameworks for external stakeholder integration. Lastly, the mentioned prototype will be further extended and evaluated.

Conclusion: This study highlights the transformative role of integrated BPLs in LCPs, addressing the gap for processes existing "outside the ERP standard." By defining the library's requirements through a MDE lens, we identified the essential functions, integration fallbacks, and governance roles, from Product Owners to External Users, needed for success. While LCPs foster innovation and reduce IT dependency, they necessitate a structured lifecycle approach to ensure long-term stability. Our findings contribute to MDE theory and decentralized RE research, offering a foundation for future studies on quantifying the impacts of LCP agility. As enterprise software becomes increasingly modular, these BPLs will be crucial in shaping the next generation of adaptable business architectures.

Acknowledgements

This research is part of the research project OpenData4KMU (01IF24639N) funded by the German Federal Ministry for Economic Affairs and Energy.

Use of generative artificial intelligence (AI): Generative AI tools were used to support language refinement, including grammar correction and minor sentence improvements.

REFERENCES

Abendroth, A., & Bender, B. (2024). Ein Werkzeug zur Analyse von Komplexität von Low-Code und No-Code Add-ons: Ein Design Science Ansatz. HMD Praxis der Wirtschaftsinformatik, 61(5/2024), 1180–1212.

Abendroth, A., & Bender, B. (2025). Bridging the Gap: Low-Code Platforms and the Future of ERP Customization. AMCIS 2025 Proceedings, 28. https://aisel.aisnet.org/amcis2025/intelfuture/intelfuture/28/

Abendroth, A., Bender, B., & Gronau, N. (2024). The Evolution of Original ERP Customization: A Systematic Literature Review of Technical Possibilities. Proceedings of the 26th International Conference on Enterprise Information Systems (ICEIS 2024), 1, 17–27.

Alamin, M. A. A., Uddin, G., Malakar, S., Afroz, S., Haider, T., & Iqbal, A. (2023). Developer discussion topics on the adoption and barriers of low code software development platforms. Empirical Software Engineering, 28(1), 4.

Alfonso, I., Conrardy, A., & Cabot, J. (2025). Towards the Interoperability of Low-Code Platforms. In L. Pufahl, K. Rosenthal, S. España, & S. Nurcan (Eds.), Intelligent Information Systems (Vol. 557, pp. 3–11). Springer Nature Switzerland.

Balint, B. (2017). Maximizing the Value of Packaged Software Customization: International Journal of Enterprise Information Systems, 13(1), Article 1.

Barna, L.-E.-L., Ionescu, B.-Ştefan, & Ionescu-Feleagă, L. (2021). The Relationship between the Implementation of ERP Systems and the Financial and Non-Financial Reporting of Organizations. Sustainability, 13(21), Article 21.

Bender, B., & Korjahn, N. (2023). ERP-Updatestudie 2022/2023 (ERP-Updatestudie, pp. 1–26). Institut für Wirtschaftsinformatik und Digitale Gesellschaft e.V. https://lswi.de/lehrstuhl/blog/veroeffentlichung-der-erp-updatestudie-2022-23

Benders, J., Batenburg, R., & van der Blonk, H. (2006). Sticking to standards; Technical and other isomorphic pressures in deploying ERP-systems. Information & Management, 43, 194–203.

Bies, L., Weber, M., Greff, T., & Werth, D. (2022). A mixed-methods study of low-code development platforms: Drivers of digital innovation in SMEs. Proc. of the International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME), 1–6.

Bizagi. (2026). Business Orchestration for AI Impact | Bizagi. https://www.bizagi.com/en

Cabot, J. (2020). Positioning of the low-code movement within the field of model-driven engineering. Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings, 1–3.

Camunda. (2026). The Universal Process Orchestrator. Camunda. https://camunda.com/

Di Ruscio, D., Kolovos, D., De Lara, J., Pierantonio, A., Tisi, M., & Wimmer, M. (2022). Low-code development and model-driven engineering: Two sides of the same coin? Software and Systems Modeling, 21(2), 437–446.

Döring, N., & Bortz, J. (2016). Forschungsmethoden und Evaluation (5.th). Springerverlag.

Faulbaum, F., Prüfer, P., & Rexroth, M. (2009). Was ist eine gute Frage? VS Verlag für Sozialwissenschaften.

Gläser, J., & Laudel, G. (2010). Experteninterviews und Qualitative Inhaltsanalyse. VS Verlag.

Gode, A., Roesner, D., Bingler, D., Naujoks, F., Sontow, K., & Finkler, M. (2023). Programmieren für Dummies: Bedeutet Low-Code das Ende von ERP? (p. 7) [Diskussionspapier aus dem Bitkom Arbeitskreis ERP]. Bitkom e.V.

Gronau, N. (2026). ERP-Systeme: Architektur, Management und Funktionen des Enterprise Resource Planning: 5.th edition. De Gruyter Oldenbourg.

Haddara, M., Gøthesen, S., & Langseth, M. (2022). Challenges of Cloud-ERP Adoptions in SMEs. Procedia Computer Science, International Conference on ENTERprise Information Systems / ProjMAN - International Conference on Project MANagement / HCist - International Conference on Health and Social Care Information Systems and Technologies 2021, 196, 973–981.

Hailpern, B., & Tarr, P. (2006). Model-driven development: The good, the bad, and the ugly. IBM Systems Journal, 45(3), 451–461.

Irnawati, I., & Rahayu, R. (2025). The Role of Digital Transformation through Modular ERP in Enhancing Corporate Adaptability to Global Market Changes. Jurnal Ar Ro'is Mandalika (Armada), 5(2), 468–479.

Krouwel, M. R., Op 't Land, M., & Proper, H. A. (2022). Generating Low-Code Applications from Enterprise Ontology. In B. S. Barn & K. Sandkuhl (Eds.), The Practice of Enterprise Modeling (pp. 18–32). Springer International Publishing.

Lourenço, M., Gasiba, T. E., & Pinto-Albuquerque, M. (2023). You are doing it wrong: On vulnerabilities in low code development platforms. CYBER 2023 : The Eighth International Conference on Cyber-Technologies and Cyber-Systems. https://repositorio.iscte-iul.pt/handle/10071/29454

Mayring, P. (2020). Qualitative Inhaltsanalyse. In Handbuch qualitative Forschung in der Psychologie (pp. 495–511). Springer.

Palleduhn, D. U., & Neuendorf, H. (2013). Architektur integrierter Informationssysteme (ARIS). In Geschäftsprozessmanagement und integrierte Informationsverarbeitung. Walter de Gruyter.

Parthasarathy, S., & Sharma, S. (2017). Impact of customization over software quality in ERP projects: An empirical study. Software Quality Journal, 25(2), 581–598.

Pesme, J.-O. (2023). Tracing and tracking wine bottles: Protecting consumers and producers. BIO Web of Conferences, 68, 03028. https://www.bio-conferences.org/articles/bioconf/abs/2023/13/bioconf_oiv2023_03028/bioconf_oiv2023_03028.html

Picek, R. (2023). Low-code/No-code Platforms and Modern ERP Systems. 2023 International Conference on Information Management (ICIM), 44–49.

Rothenberger, M. A., & Srite, M. (2009). An Investigation of Customization in ERP System Implementations. IEEE Transactions on Engineering Management, 56(4), 663–676.

SAG Aris GmbH. (2026). Business Process Optimization for the AI Era—ARIS. https://aris.com/

SAP SE. (2026a). Relating the SAP Enterprise Architecture Framework to TOGAF®. https://learning.sap.com/learning-journeys/exploring-the-sap-enterprise-architecture-framework-foundation/relating-the-sap-enterprise-architecture-framework-to-togaf-

SAP SE. (2026b). SAP Signavio | Business Process Transformation Suite. SAP Signavio | Business Process Transformation Suite. https://www.signavio.com/

Scheer, A.-W., & Nüttgens, M. (2000). ARIS Architecture and Reference Models for Business Process Management. In W. M. P. van der Aalst, J. Desel, & A. Oberweis (Eds.), Business Process Management, Models, Techniques, and Empirical Studies (Vol. 1806, pp. 376–389). Springer.

Software AG. (2024). Methodenhandbuch ARIS - VERSION 10.0—SERVICE RELEASE 27 AND HIGHER.

Sparx Systems. (2026). Enterprise Architect. https://sparxsystems.com/platforms/mda_tool.html

Suija-Markova, I., Briede, L., Gaile-Sarkane, E., & Ozolina-Ozola, I. (2020). Multitasking in Knowledge Intensive Business Services. Emerging Science Journal, 4, 305–318.

Tang, L. (2022). ERP Low-Code Cloud Development. 2022 IEEE 13th International Conference on Software Engineering and Service Science (ICSESS), 319–323.

Thalhofer, C., & Krumm, Y. (2024). 16 Prozessmodellierungstools im Test. ERP Management, 20(3), 21–39.

The Open Group. (2018). The TOGAF® Standard, Version 9.2. The Open Group.

Viljoen, A., Radić, M., Hein, A., Nguyen, J., & Krcmar, H. (2024). Governing Citizen Development to Address Low-Code Platform Challenges. MIS Quarterly Executive, 23(3). https://aisel.aisnet.org/misqe/vol23/iss3/6

Visual Paradigm. (2026). Visual Paradigm—AI-Powered Visual Modeling. https://www.visual-paradigm.com/