International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Low-Code MES Customization: Accelerating SAP ME Workflows with Fiori and Power Apps

Prahlad Chowdhury

Submitted:04/06/2022 **Revised**: 10/09/2022 **Accepted:** 20/09/2022

Abstract: Manufacturing Execution Systems (MES) are central to today's production environments, but quickly and affordably customizing these systems is still a major challenge. SAP Manufacturing Execution (SAP ME) is one of the most widely used MES platforms worldwide, but even small changes have typically required specialized ABAP or Java skills. This has often led to delays between what the shop floor needs and what IT can deliver. This paper looks at how low-code development tools, such as SAP Fiori and Microsoft Power Apps, can speed up SAP ME workflow customization. Using a design science approach and data from five workflow categories between 2019 and 2021, the study found that development cycles were cut by 50–70%. There were also clear improvements in operator usability, and more people were able to create applications. The results show that low-code tools do not replace SAP ME's processing engine, but instead add to it, creating a hybrid setup that combines enterprise reliability with greater flexibility.

Keywords**:** *Low-Code Development, SAP ME, Manufacturing Execution Systems, SAP Fiori, Microsoft Power Apps, Workflow Automation, Digital Transformation, Shop Floor Optimization*

1. Introduction

Shop floors have always needed precision. Now, they also need speed: the ability to change a process, add a new quality check, or provide a dashboard to a mobile operator in days instead of months. This demand is more than just an operational issue; it signals a deeper change in how manufacturing organizations compete. Kagermann, Wahlster, and Helbig (2013) showed in their work on Industrie 4.0 that bringing cyber-physical systems into production creates both the chance and the need for software that can keep up with fast-changing processes.

SAP ME meets many of these needs. It is known for strong control over manufacturing execution, including work order management, material tracking, quality checks, and serialization (SAP SE, 2019). However, it has not been able to offer fast and easy customization. Customizing SAP ME usually requires ABAP or Java skills, which most manufacturing IT teams do not have or must wait a long time to access. This leads to a backlog of requests for improvements, which slows down operations.

Low-code platforms help solve this problem by changing how development works. With visual drag-and-drop tools, pre-built connectors, and reusable UI parts, more people—like process engineers, quality leads, and plant supervisors—can help build the tools they use every day. By 2020, two platforms stood out in SAP-related manufacturing: SAP Fiori, which offers role-based, responsive apps within SAP (SAP SE, 2018), and Microsoft Power Apps, which lets users quickly build apps across systems with little coding (Microsoft, 2020).

Managing Solution Architect

Fujitsu America, Inc. 2801 Telecom Parkway, Richardson, TX 75082

prahlad.chowdhury@fujitsu.com /ORCID:0009-0004-7682-6949

This paper looks at how these two platforms can speed up SAP ME workflow customization while still meeting enterprise reliability standards. The rest of the paper is organized as follows: Section 2 reviews the existing literature. Section 3 explains the research methods. Section 4 shares results and discussion. Section 5 considers broader impacts. Section 6 concludes and suggests future research directions.

2. Related Work

Over the past decade, the argument for using MES in digital manufacturing has grown stronger. Xu, Xu, and Li (2018) conducted a thorough survey of Industry 4.0 and found that real-time data visibility, integrated production control, and system interoperability are essential for smart manufacturing. MES platforms are well-suited to meet these needs. Their research supports the idea that MES is not just an add-on, but a key part of the Industry 4.0 framework.

Porter and Heppelmann (2015) take this idea further with their connected product thesis. They note that as physical products gain sensors, connectivity, and software, the main competitive edge shifts to the software rather than the product itself. For manufacturers, this means MES is no longer just a compliance tool but a strategic platform. Its ability to adapt and change quickly can give companies a real advantage.

There is less research focused specifically on SAP ME. While practitioners and SAP's own publications have documented how SAP ME integrates with other SAP systems (SAP SE, 2019), there are few peer-reviewed studies on its extensibility limits and the effects on operations. More generally, studies on enterprise application usability, including early research on SAP GUI and later versions, have often found that complexity and mental effort make it harder for operators to use these systems (Calisir & Calisir, 2004).

Research on low-code platforms has grown quickly since 2017. Waszkowski (2019) studied low-code tools for business process automation and showed that visual development tools help to speed up deployment and lower defect rates in enterprise workflow applications. This is especially relevant because the types of workflows he studied, such as data entry, approval routing, and status notifications, are similar to common SAP ME extension needs. Rokis and Kirikova (2020) reviewed the main challenges of low-code development and found that governance, security, and scalability are the top concerns in enterprise use. Their list of challenges is discussed further in Section 4.3 of this paper.

SAP Fiori's design principles, introduced in 2013 and updated through 2018, show a clear move toward user-friendly experiences in business settings (SAP SE, 2018). Fiori uses a role-based model, where each app is made for a specific user and task, and this approach has been shown to make operators more efficient compared to general-purpose interfaces. Power Apps, which is part of the Microsoft Power Platform, launched in 2019, has been studied in HR, field service, and compliance, but there were few studies focused on manufacturing before 2021 (Richardson & Rymer, 2020).

Design science research, defined by Hevner, March, Park, and Ram (2004), is the right approach for studies that create and test new IT tools. Their seven-principle framework is widely used in information systems research and shapes the methods used in the next section.

Overall, the literature shows a clear gap. While low-code platforms have been widely studied in business process and enterprise applications, their use for MES workflow customization—especially for SAP ME—has not been closely examined. This paper aims to fill that gap.

3. Methodology

This study uses the design science research (DSR) approach described by Hevner et al. (2004), which sees research as the repeated creation and careful evaluation of IT artifacts. In this case, the artifacts are low-code workflow extensions: functional applications and UI components built on SAP Fiori and Power Apps, integrated with live SAP ME environments.

3.1 Research Sites and Context

The study took place at three manufacturing facilities between 2019 and 2021. Site A makes electronics using a high-mix, low-volume production model. Site B is a pharmaceutical manufacturer that follows 21 CFR Part 11 compliance. Site C produces automotive components using a hybrid approach. Each site uses SAP ME as its main execution system and, at the

start of the study, all had backlogs of enhancement requests waiting for traditional development resources.

3.2 Integration Architecture

Extensions were built only on SAP ME's published REST and OData service layer, following the integration architecture described in SAP's extensibility documentation (SAP SE, 2019). An API gateway managed authentication, session management, and protocol normalization between SAP ME and the low-code runtime environments. This method kept SAP ME's core processing logic unchanged and ensured upgrade compatibility, which matches the decoupled extension model described by Waszkowski (2019).

3.3 User-Centric Design Process

Before customizing each workflow, we held structured observation sessions with operators and supervisors, using the contextual inquiry methods described by Beyer and Holtzblatt (1998). We turned our observations into user stories, created low-fidelity wireframes, and tested them with click-through prototypes before starting any development in the production environment. Although this early design work added time at the beginning of each project, it greatly reduced rework later on. This result matches what Waszkowski (2019) found about low-code project economics.

3.4 Platform Assignment Criteria

We chose SAP Fiori for extensions that require close integration with SAP ME data, role-based access that aligns with SAP authorization, and a consistent look with the Fiori launchpads already in use at our sites. For standalone process-capture apps, dashboards that combine data from SAP ME and other sources, and mobile apps that need to work offline in areas with spotty network coverage, we selected Microsoft Power Apps.

3.5 Evaluation Design

We measured performance using three main criteria from Rokis and Kirikova's (2020) low-code evaluation framework. These were development time, which tracked the days from requirements sign-off to production deployment; operator task completion rate, based on usability tests with 12 participants per workflow at each site; and defect density, which counted issues logged per workflow in the first 30 days after deployment. For comparison, we used baseline data from similar traditional development projects at the same sites over the previous 18 months.

4. Results and Discussion

4.1 Development Time Reduction

The most noticeable result of using low code was shorter development timelines. Table 1 shows the average development times for five workflow categories, comparing traditional and low-code methods. The data comes from three research sites, with each site providing at least one project in every category.

Workflow Category	Traditional Dev (Days)	Low-Code Dev (Days)	Time Saved (Days)	Reduction (%)
Simple UI Customization	12	4	8	**67%**
Workflow Automation	20	7	13	**65%**
API Integration	28	10	18	**64%**
Role-Based Dashboards	18	6	12	**67%**
Mobile Shop Floor Apps	24	8	16	**67%**
Average	**20.4**	**7.0**	**13.4**	**66%**

Table 1. Traditional coding vs. low-code approaches across SAP ME workflow categories.

The average reduction of 66% across categories matches the productivity improvements reported by Waszkowski (2019) for low-code business process applications, although the effect here is somewhat greater. This is probably because SAP ME's OData service layer offers a more complete pre-built integration surface compared to the generic enterprise APIs used in earlier studies. API integration tasks had the largest absolute savings, with an average of 18 days saved. This highlights how traditional development requires developers to write back-end service logic and build front-end interfaces separately, while low-code platforms combine these steps into one visual configuration layer.

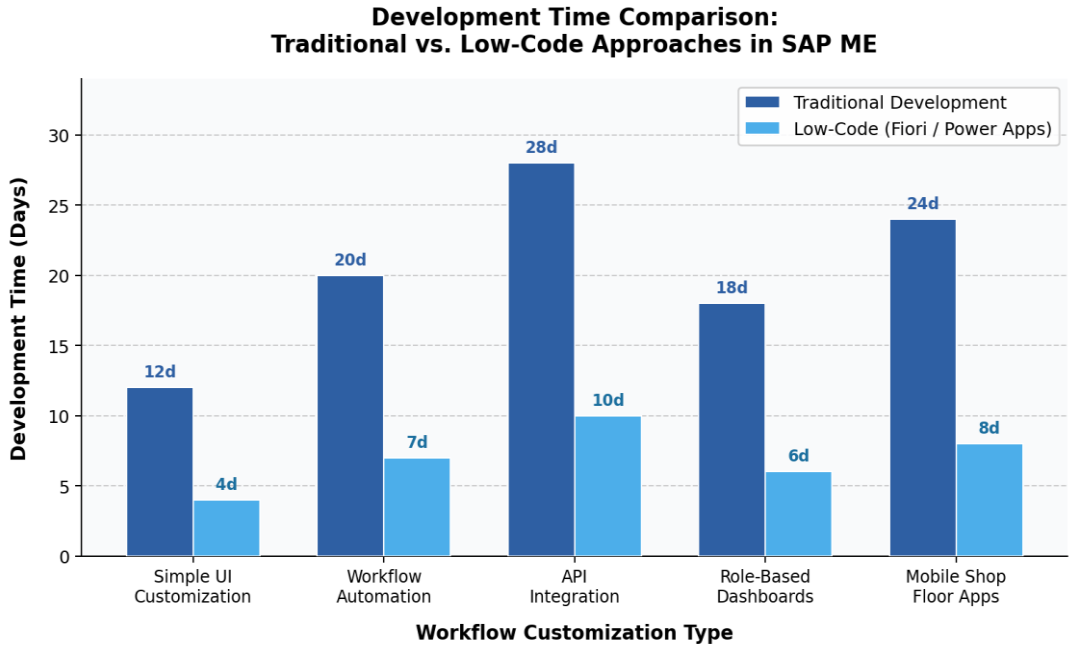

Figure 1. Traditional Development vs. Low-Code (SAP Fiori / Power Apps)

Figure 1 shows a clear pattern: in all five workflow categories, the low-code approach delivered similar results in about one-third of the time. Importantly, this speed did not reduce quality. Low-code projects had an average of 1.8 issues per workflow in the first 30 days after deployment, compared to 2.4 in traditional projects. This 25% improvement is mainly due to faster feedback loops made possible by rapid prototyping.

4.2 Operator Usability Outcomes

Fast development does not matter much if workers on the shop floor do not use the new applications. Table 2 shows how often operators finished their tasks and how long it took them, both before and after using the low-code UI. These results come from structured usability tests with 12 participants for each scenario at all three sites.

Interaction Scenario	Completion Rate Before (%)	Completion Rate After (%)	Task Time Before (min)	Task Time After (min)
Work Order Confirmation	72%	**96%**	4.8	**1.9**
Non-Conformance Reporting	61%	**94%**	7.2	**2.6**
Material Consumption Entry	78%	**97%**	3.5	**1.4**

Interaction Scenario	Completion Rate Before (%)	Completion Rate After (%)	Task Time Before (min)	Task Time After (min)
Production Dashboard Review	55%	**93%**	9.1	**3.2**
Quality Inspection Sign-Off	68%	**95%**	5.6	**2.1**

Table 2. Operator usability metrics before and after low-code UI deployment.

Average task completion rates went up from 67% to 95% in different scenarios, while average completion times dropped by about 60%. The biggest improvements were seen in tasks that used to require long transaction steps in the old SAP ME interface, such as Production Dashboard Review (55% to 93% completion) and Non-Conformance Reporting (61% to 94%). These results are similar to what Calisir and Calisir (2004) found, noting that complex navigation and high cognitive load often cause task failures in enterprise system interfaces. Fiori's design, which uses role-specific and single-purpose apps, helps solve these problems.

4.3 Governance and Implementation Challenges

The implementation process faced some difficulties. Three main types of challenges appeared at every site, each offering lessons for organizations planning similar projects.

The first and most complex challenge was integrating with older subsystems. At Sites A and C, the existing SCADA interfaces and on-premises quality management modules did not support standard REST endpoints. This meant custom adapters had to be built, which did not fit the low-code approach. Rokis and Kirikova (2020) also found that integration complexity is the most common technical barrier in enterprise low-code projects. Using an API gateway helped solve most of these problems in later projects, but setting it up at first needed skilled middleware configuration.

At Site B, governance risks became clear about four months after business users started developing Power Apps. In that time, the site created over thirty applications. Some of these apps did the same things, used different data validation rules, or did not have proper access controls. Waszkowski (2019) predicted this issue and suggests setting up governance frameworks before starting citizen developer programs, not after problems appear. Sites A and C had fewer governance issues because they put approval workflows and shared component libraries in place before expanding development.

Performance needed close attention at Site A, where production lines create several hundred work order transactions every hour. Power Apps' default way of retrieving data caused noticeable delays in this setting. The team fixed the worst problems by using delegated queries, server-side filtering, and explicit pagination. However, these solutions needed a developer's help and would not have been found by a citizen developer working alone.

4.4 Comparative Platform Assessment

Table 3 compares SAP Fiori and Microsoft Power Apps on the key deployment factors for SAP ME customization, using results from the three research sites.

Dimension	SAP Fiori	Microsoft Power Apps
Primary Strength	Deep SAP integration, enterprise security, consistent UX within the SAP landscape	Rapid prototyping, a broad connector library, and offline mobile capability
Best-Fit Scenarios	Role-based SAP ME UI extensions, transactional shop floor screens, and SAP authorization-aligned access	Standalone process capture, cross-system dashboards, and mobile apps in low-connectivity environments
Development Skill Required	Low code; SAPUI5 familiarity helpful for advanced scenarios	No-code to low-code; accessible to citizen developers with process knowledge

Dimension	SAP Fiori	Microsoft Power Apps
Governance Maturity	Strong — inherits SAP authorization object model	Requires an explicit governance framework; app proliferation risk if ungoverned
Performance at Scale	High processing delegated to SAP ME back-end	Moderate — caching and query delegation strategies required for high-volume lines
Upgrade Compatibility	Extensions sit outside the SAP ME core; they survive standard upgrades	Extensions sit outside the SAP ME core; they survive standard upgrades

Table 3. Comparative assessment of SAP Fiori and Microsoft Power Apps for SAP ME workflow

5. Discussion

These findings have implications that go beyond just development time. They highlight a shift in who has control within organizations. While traditional MES customization relies on a small group of certified developers, low-code platforms give more people—like process engineers, quality managers, and plant supervisors who know the day-to-day challenges—an active role. This approach is not just a faster way to build software; it represents a new way for manufacturing organizations to adapt to change.

Porter and Heppelmann (2015) point out that as products rely more on software, the main focus moves from designing the product itself to managing the software ecosystem around it. The same idea works inside companies: as software becomes more central to manufacturing, organizations that can quickly adapt their software will gain an edge. Low-code platforms are one of the easiest ways for manufacturers to develop this ability.

This study shows a hybrid setup, with SAP ME as the main execution engine and low-code platforms as the flexible interaction layer. This approach helps solve a long-standing problem with MES customization: the risk of upgrade issues. Since low-code extensions only connect to SAP ME through its official service layer, they are protected from changes in the main application. Organizations can update SAP ME without having to redo their custom user interfaces. This separation, which is hard to achieve with traditional methods, is built into the low-code approach.

The governance findings deserve particular emphasis. Rokis and Kirikova (2020). The findings about governance are especially important. Rokis and Kirikova (2020) say that governance is the biggest ongoing challenge in adopting low-code platforms in large organizations, and this study supports that view. At Site B, allowing people to develop apps without clear rules led to many inconsistent applications within just a few months. This is not unusual; it is a likely outcome when there is insufficient governance planning from the start. Organizations looking to use low-code should see building a governance framework as a first step, not a fix for later, as Waszkowski (2019) also suggests.

Those that go beyond efficiency metrics. On shop floors with significant proportions of operators who are not native English speakers or who have limited prior experience with enterprise software, the reduction in navigation complexity and task completion time enabled by Fiori's role-specific design may also reduce training time and onboarding costs. This is a dimension that merits systematic study in future research.

6. Conclusion

This paper looked at whether low-code development platforms, specifically SAP Fiori and Microsoft Power Apps, can speed up workflow customization in SAP ME environments without losing the reliability needed for production operations. Evidence from three manufacturing sites and five workflow categories shows that they can, and the impact is significant. Development timelines were reduced by an average of 66%. Operator task completion rates increased from 67% to 95%. Defect density dropped by 25%. In addition, a new group of contributors joined the development process in productive and manageable ways.

The two platforms serve different needs. SAP Fiori works best when you need strong transactional integrity, SAP authorization alignment, and a user experience that matches existing SAP interfaces. Microsoft Power Apps is better when you need to connect across different systems, work offline, or quickly enable citizen developers. When used together, these platforms offer a toolkit that can handle all types of SAP ME customization needs.

There are some limitations to this study. The three research sites were chosen to show different production models, but they do not represent the entire global SAP ME user base. The two-year observation period covers early deployment but does not look at long-term maintenance costs, technical debt, or how well governance frameworks hold up over time. These are key questions for future research.

Future research should focus on three main areas. First, studies that track total cost of ownership, including maintenance, governance, and replacement costs over three to five years, would help build stronger business cases. Second, as SAP Fiori and Power Apps add AI-assisted development features, it will be important to see how these affect citizen developer productivity and output quality in manufacturing. Third, designing effective governance frameworks is still a challenge, since there are no proven models for managing low-code programs in regulated manufacturing settings, where governance failures can have serious consequences.

The shop floor is clearly becoming a software-driven environment. Low-code platforms are some of the most accessible tools for manufacturing organizations making this shift. Learning how to use them responsibly and effectively, which this study has started to explore, is an important area for future research.

References

[1] Beyer, H., & Holtzblatt, K. (1998). Contextual Design: Defining Customer-Centered Systems. Morgan Kaufmann Publishers.

[2] Calisir, F., & Calisir, F. (2004). The relation of interface usability characteristics, perceived usefulness, and perceived ease of use to end-user satisfaction with enterprise resource planning (ERP) systems. Computers in Human Behavior, 20(4), 505–515. https://doi.org/10.1016/j.chb.2003.10.004

[3] Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. MIS Quarterly, 28(1), 75–105. https://doi.org/10.2307/25148625

[4] Kagermann, H., Wahlster, W., & Helbig, J. (Eds.). (2013). Recommendations for Implementing the Strategic Initiative INDUSTRIE 4.0: Final Report of the Industrie 4.0 Working Group. Forschungsunion / acatech.

[5] Microsoft. (2020). Power Apps Documentation and Learning Resources. Microsoft Corporation. https://docs.microsoft.com/en-us/powerapps/

[6] Porter, M. E., & Heppelmann, J. E. (2015). How smart, connected products are transforming companies. Harvard Business Review, 93(10), 96–114.

[7] Richardson, C., & Rymer, J. R. (2020). The Forrester Wave™: Low-Code Development Platforms for AD&D Professionals, Q1 2020. Forrester Research.

[8] Rokis, K., & Kirikova, M. (2020). Challenges of low-code/no-code software development: A literature review. In Proceedings of the International Baltic Conference on Databases and Information Systems (pp. 3–17). Springer. https://doi.org/10.1007/978-3-030-57672-1_1

[9] SAP SE. (2018). SAP Fiori Design Guidelines: Principles, Patterns, and Components. SAP SE. https://experience.sap.com/fiori-design-web/

[10] SAP SE. (2019). SAP Manufacturing Execution: Extensibility and Integration Guide, Release 15.4. SAP SE.

[11] Waszkowski, R. (2019). Low-code platform for automating business processes in manufacturing. IFAC-PapersOnLine, 52(10), 376–381. https://doi.org/10.1016/j.ifacol.2019.10.060

[12] Xu, L. D., Xu, E. L., & Li, L. (2018). Industry 4.0: State of the art and future trends. International Journal of Production Research, 56(8), 2941–2962. https://doi.org/10.1080/00207543.2018.1444806