Next-Generation SAP Testing with Cloud ALM - Evidence from Four Company Cases

Judit GOMBKÖTŐ[1] - Tímea KOZMA[2] - Zsigmond Gábor SZALAY[3]

DOI: 10.29180/978-615-6886-30-9_3

Abstract

Enterprise resource planning implementation projects remain high-risk undertakings, and failures frequently arise from insufficient test depth, weak defect management, and limited readiness assessment. Nowadays, the cloud environment is the platform where Enterprise Resource Planning systems have transitioned. The importance of structured test processes is a primary focus in Enterprise Resource Planning projects, as they simulate the future system. This study compares four cloud-based implementations (two with Excel for testing and two with cloud-based application lifecycle management). The study analyses how testing tools affect testing quality, defect management, and post-go-live stability. The comparison was based on quantitative project data, including test case logs, defect records, testing duration, cycle counts, and hypercare incidents. Descriptive statistics and non-parametric comparative methods were used for analysing the data. The research results show that projects with integrated lifecycle management tool support conducted deeper test cycles and detected more defects before go-live, and, as a result, systems were more stable after deployment. Excel-based testing was adequate only for smaller, less complex rollouts and led to higher defect density post-go-live due to limited traceability and fragmented workflows. The study highlights the strategic relevance of testing depth and demonstrates the benefits of integrated lifecycle management tools for implementation quality. These results contribute to Enterprise Resource Planning research by providing empirical evidence that test management tools influence project outcomes and long-term system stability.

Keywords: Systems, Applications, and Products in Data Processing (SAP), implementation, Cloud Application Lifecycle Management (CALM), Enterprise Resource Planning (ERP), testing, defect management, testing quality, defect, go-live, cloud, project governance, hypercare

Introduction

Enterprise Resource Planning (ERP) systems such as Systems, Applications, and Products in Data Processing (SAP) are fundamental to company operations, as they integrate financial, logistics, purchasing, and operations departments. The primary goal of ERP systems is to make operations more efficient and safer; however, their implementation remains difficult, costly, and risky. A study states that only 64% of ERP projects are successful, while the remaining 36% encounter problems such as budget increases, schedule delays, or scope changes (Panorama Consulting Group, 2023).

The Standish Group's CHAOS Report lists IT project failure factors and names lack of user involvement, insufficient executive ownership, unclear requirements, and ineffective

[1] PhD student, Doctoral School of Economic and Regional Sciences, Hungarian University of Agriculture and Life Sciences (MATE), Gödöllő, Hungary, email gombkoto.judit.klara@phd.uni-mate.hu
[2] Associate Professor, Budapest University of Economics and Business, Department of Process Management, Budapest, Hungary, e-mail: kozma.timea@uni-bge.hu
[3] Associate Professor, Department of Agridigitalization and Extension Activities, Institute of Rural Development and Sustainable Economy, Hungarian University of Agriculture and Life Sciences, Gödöllő, Hungary, e-mail: szalay.zsigmond.gabor@ui-mate.hu

project planning and delivery. ERP failures also highlight these issues and are heavily influenced by factors such as cross-functional complexity, poor data quality, and insufficient testing.

Testing is still one of the most critical process steps, but it is not properly standardised as part of ERP implementation. Many organizations still use Excel spreadsheets to manage their test cases, execution status, and defects, simply because they find this solution easier and more flexible. Although Excel spreadsheets offer limited collaboration, prior experience is not included, and no integrated defect tracker is provided. Defect resolution and traceability are key to go-live success.

Systems, Applications, and Products in Data Processing (SAP) Cloud Application Lifecycle Management (CALM) is SAP's new cloud-based lifecycle management platform for the future and is expected to overcome current limitations. The platform offers comprehensive process traceability, live dashboards, integrated defect workflows, and alignment with SAP Activate. CALM can provide centralised test-case design, future-state simulation, and readiness analytics throughout project phases.

This research explores the impact of testing tools (Excel and SAP Cloud ALM) on the success of SAP implementation projects. By comparing the real-life situations of four companies (two using Excel and two using CALM), the study examines variables such as the thoroughness and frequency of testing, defect leakage, and system stability after going live.

This research has two main focuses:

- To establish whether the use of structured and tool-supported testing leads to higher quality in implementation.
- To provide empirical research to the existing literature on critical success factors and ERP testing management.

Literature review

Introduction to ERP Implementation Research

Enterprise Resource Planning (ERP) systems have been the subject of the most in-depth research on information systems for over 20 years. Key studies by Esteves and Pastor (2001) introduced the concept that ERP implementations must be understood and evaluated across multiple dimensions of technology, organisation, people, and strategy. In the two decades since, ERP research has continued to show that the success or failure of ERP implementation depends a lot on the organisation, industry, and approach. One of the main conclusions from past research on ERP is that technology alone cannot guarantee success. As ERP systems allow organisations to integrate complex business processes across departments, the success of implementations is closely related to management and leadership, governance, user readiness, and data quality (Gattiker & Goodhue, 2005; Bradford & Florin, 2003). The latest research has not changed this and has established quite the opposite: ERP technology moves from traditional on-premises environments to cloud-based, agile delivery models (Wynn et al., 2024).

With these changes, a core element of ERP implementation success that has often been overlooked in research is the scope of testing, test management, and the maturity of defect resolution, which this study focuses on.

Critical Success Factors in ERP Implementations

Critical success factors (CSFs) in ERP implementations have been the subject of extensive research. Al-Fawaz, Al-Salti, and Eldabi (2008) conducted a broad-based study that showed the most often mentioned CSFs to be top management support, a strong project management team, user involvement, well-defined project requirements, training, and communication. Maditinos et al. (2012) also found that organizational culture, project governance, and user readiness

significantly impacted the success of ERP projects in Greek companies. Kouriati et al. (2022), who studied the implementation of ERP systems in companies in the food and agricultural processing sector, identified CSFs including change management, process clarity, and user engagement and provided quantitative data to support the view that these factors play a significant role in successful implementation. Other regional studies (Shafi et al., 2019 in Pakistan; Idilbi & Abu-Shanab, 2022 in Qatar) also support these findings and emphasize that employees' experiences with support, communication, and training directly affect whether ERP is accepted and successful.

As ERP projects move towards cloud-based solutions, the role of CSFs is likely to become stronger. According to Wynn et al. (2024), ERP implementations in the digital era follow different patterns: they occur in shorter, more frequent release cycles, are more complex, involve a greater number of systems and tools, and rely more heavily on lifecycle management tools for governance. These changes put testing under even more pressure.

ERP Value Realisation and Long-Term Performance

Beyond success in the project phase, some researchers have examined the positive impact of ERP systems on long-term organisational performance. Hitt, Wu, and Zhou (2002) showed that productivity improvements following ERP adoption largely depend on organisational readiness, process standardisation, and data governance. Their results point to the interplay between technology and managerial practices: when internal capabilities are lacking, firms struggle to translate ERP capabilities into measurable value. More recent evidence supports this view. Bradford and Florin (2003) found that innovation diffusion factors, such as user learning, communication, and perceived usefulness, strongly influence the benefits observed after implementation. Gattiker and Goodhue (2005) studied outcomes at the plant level and concluded that employee interdependence and process harmonisation have a significant impact on ERP success. Together, these studies make it clear that technology by itself does not generate value: it is the organisation's ability to implement, manage, and stabilise the system that drives long-term performance. This clearly links value realisation to testing governance: if test planning, defect management, and go-live readiness are not carried out meticulously, organisations will experience issues that not only affect the implementation but will also hinder the benefits of the ERP system in the future.

The Role of Testing Quality in ERP Implementations

Initially, in the ERP literature, testing was considered a project activity, but more recent research regards it as a strategic factor for successful implementation. Shatat and Dana (2016) showed that insufficient testing leads to operational instability, increased defects, and user dissatisfaction. Shafi et al. (2019) concluded that testing quality and user training were among the most significant factors for user satisfaction in ERP systems implemented in the manufacturing sector. Testing is now seen as a business readiness check. Immidi and Mane (2025), in their study of the confluence of Artificial Intelligence and SAP IBP, noted that strong, well-documented testing frameworks are indispensable for ensuring consistency across modules and reliable performance. Their research highlighted a key point: only by covering end-to-end scenarios can we be sure that the forecasting system will behave as expected in a real-world setting rather than merely testing individual components.

ERP implementations that do not allocate sufficient resources to testing often encounter many defects during go-live and hypercare. This, in its turn, delays the stabilisation process and erodes user confidence. Mahraz, Benabbou and Berrado (2019) mentioned the lack of testing depth as one of the main risk factors for ERP projects in their literature review. Moreover, research on implementation methodologies (Viljakainen, 2024) shows that SAP projects must undergo multiple well-structured test cycles, particularly when they are multi-country or multi-

phase. In such cases, business processes need to be checked and re-checked as they evolve from one release to the next.

Testing Tools and Lifecycle Management

Modern ERP systems have become sophisticated and valuable, but many companies still rely on Excel to manage test cases. If we are to believe Aires and Abrantes (2022), such reliance on spreadsheets for testing will become increasingly problematic as project size increases, because Excel cannot provide version control, team collaboration, and traceability up to the desired standards. The transition to cloud ERP systems has only accelerated the adoption of integrated Application Lifecycle Management (ALM) tools. According to Gupta (2025), who studied over 80 SAP implementations globally, the key to managing large-scale ERP deployments is maintaining consistency, visibility, and traceability across the project stages. Madathala and Yeturi (2025) also observed that a well-defined defect workflow and early issue identification were crucial to the success of SAP ERP projects in Indian companies.

SAP Cloud ALM (CALM) is considered to be the future of SAP lifecycle management. Its main features are as follows:

- the ability to trace a business process all the way to the test case,
- a seamless integration with the defect management system,
- real-time dashboards and key performance indicators for readiness,
- an implementation that follows the SAP Activate methodology,
- support for distributed teams through shared workspaces.

There is not much research on CALM in academic circles, but it is worth noting that the characteristics of CALM identified so far are very much in line with general findings on ALM systems. Such systems are effective in ensuring thorough testing and fewer defects slip through, and they are also adequate for supporting project delivery in an auditable manner (Vaid, Reddy & Prabhagaran, 2024; Wynn et al., 2024). Apparently, testing is about managing risk and ensuring that the business is ready for change. In this context, a fully integrated ALM system offers capabilities that Excel cannot provide. In SAP projects that involve frequent releases and dependencies, and where multiple modules might interact, these properties are fundamental.

Defect Management and Post-Go-Live Stabilisation

Research consistently finds that the later defects are found in a project, the higher the cost of fixing them and the greater the disruption caused. Studying SAP MM/WMS implementations in manufacturing environments Banța (2020) identified a pattern of insufficient unit and integration testing, which ultimately led to a large number of defects being discovered after go-live and a significant hypercare period. Likewise, Salas (2023) contended that modern ERP implementations using agile methods must identify defects iteratively and maintain continuous feedback loops to achieve stability.

Idilbi and Abu-Shanab (2022) described how organisations with robust defect management processes and clear accountability structures achieved smooth go-lives and high user satisfaction. Their observations are supported by broader studies, which conclude that adopting a structured approach to logging defects, identifying them promptly, and being transparent about them reduces the risk of operational failure. One of the ways SAP Cloud ALM meets this need is by connecting defect workflows to test cases and business process models, so that anyone can follow the thread of a defect right through a project.

Summary of Gaps in the Existing Literature

Despite the large number of studies dealing with ERP CSFs, value realisation and testing principles, there are some open questions:

- Empirical research is largely lacking with respect to a comparison of Excel-based vs. CALM-based testing approaches in SAP environments.
- The literature rarely examines test depth and testing cadence as measurable predictors of go-live readiness and stabilisation speed.
- Testing is usually considered a technical activity rather than a strategic lever for value realisation.
- Empirical studies rarely examine defect leakage rate, scenario coverage metrics, and incident resolution outcomes across different testing tools.

This research is filling these gaps by directly comparing four SAP projects, empirically showing how testing tools influence project outcomes, defect patterns, and organisational readiness.

Research Methodology

Research Design

The present research uses a comparative multiple-case study to analyse how different test management tools (Excel spreadsheets and SAP Cloud ALM (CALM)) affect the depth of testing, defect discovery, and the project's post-go-live phase in SAP implementations. A multiple-case study method is suitable for ERP research, as implementation occurs in real-world organisations and involves a complex interplay among tools, processes, and project management. The study is explanatory and comparative in nature: it does not only aim to describe differences but also seeks to understand how and why the use of a structured ALM platform may lead to more thorough testing and more stable stabilisation phases than spreadsheet-based approaches.

Case Selection and Sampling

Four SAP projects were selected based on the availability of complete test datasets and the clarity of the testing approach. The cases were selected using theoretical purposive sampling to ensure they contrast and are comparable across key dimensions.

Sampling criteria:

- Two projects where test management was done using Excel
- Two projects where test management was done using CALM
- Availability of quantitative data (test-case logs, defect logs, completion rates, duration, hypercare data) without gaps
- Similar implementation scope (SAP Public/Private Cloud rollouts)

Regarding case characteristics, the four companies are the representative mix of:

- deployment types (2 Public Cloud, 2 Private Cloud),
- testing approaches (2 Excel, 2 CALM),
- sizes and complexities (ranging from a small template rollout to an extensive multi-country programme).

The sampling is based on the following replication logic:

- Excel projects are used as literal replications (expected similar low-depth testing behaviour),
- CALM projects are used as theoretical replications (expected to provide deeper and more structured testing due to the tool capabilities).

Data Sources

Quantitative Testing Artefacts Extracted from uploaded project files:

- Test-case logs (Excel sheets and CALM exports)
- Execution status (passed, failed)
- Testing duration (months)
- Number of test cycles (mixed in all types of phases and modules)
- Defect logs (closure rates)
- Hypercare defect tickets (first 90 days).

The study is entirely data-driven.

Key Variables and Measurement

Four main variable groups were operationalised.

1. Testing Depth Indicators: total number of executed test cases, end-to-end business process coverage, number of testing cycles performed, test cases per month (testing intensity)
2. Defect Behaviour Indicators: total defects detected per phase, defect leakage, rework volume
3. Stabilisation Performance Indicators: hypercare defects (0–90 days), recurring issues
4. Tool Capability (Excel vs CALM) features: test-case traceability, real-time dashboards, integrated defect lifecycle, collaboration model

Table 1 below shows the structured way of the Key variables and Measurement used as the scope of the research.

Table 1: *Key Variables Used in the Study and Their Measurement Indicators*

Key Variables and Measurement	**Indicators**
Testing Depth	Total number of executed test cases End-to-end business process coverage Test cases per month
Defect Behaviour	Total defects detected per phase Defect severity distribution Defect leakage Rework volume
Stabilisation performance	Hypercare defects Recurring issues
Tool Capability (Excel vs CALM)	Test case traceability Real-time dashboards Integrated defect lifecycle Collaboration model

These definitions correspond to SAP Activate's testing governance model and to CALM's default configuration.

The research employs a mixed quantitative analytical approach, combining statistical comparison.

1. Descriptive Statistical Analysis: mean, median, range of test-case volumes, summarised defect patterns, testing duration, frequency, and hypercare issue distributions.
2. Derived Metrics Analysis: two calculated indicators allow comparability.

- Testing intensity = total test cases ÷ testing months.
- Defect density = hypercare defects ÷ 100 test cases.

3. Group Comparison Methods: non-parametric tests were chosen; the Mann-Whitney U test, the calculated testing intensity and defect density were used As stated in the manuscript, these did not reach significance but showed apparent directional differences favouring CALM due to low statistical power.

Validity, Reliability, and Limitations

Data are from similar SAP projects with comparable testing structures. It depends on the project how much the given test case is broken down into steps. In CALM-based testing, the test cases are more fragmented. One of the samples with a large number of items is high because not only was the implementation complex (several modules at once), but all testing was included for every period, and the project duration itself was extended due to the company's size and the complexity of processes.

The research limitation is the small sample size. Differences in scope and complexity partly influence results. However, since the study focuses solely on data-based tool comparison, a quantitative design is suitable and methodologically sound.

Results and Analysis

Overview of the Analysed Cases

The four SAP implementation projects considered in the study are summarised in Table 2. The cases refer to a balanced mix of deployment models and testing approaches: two Public Cloud and two Private Cloud implementations; two Excel-based and two CALM-based testing projects. This allows for a meaningful comparison while maintaining a certain level of project diversity in terms of scope and complexity. The cases represent both smaller, template-driven, and larger, multi-module implementations, thus providing a basis for the results to be relevant to a wide range of testing workloads and quality requirements.

Table 2: *Key characteristics of the four implementation projects*

Company	Deployment Type	Testing Tool	Customisation Level	Region / Industry
Company 1	SAP Private Cloud	CALM	Medium–High	Hungary
Company 2	SAP Public Cloud	Excel	Low	Hungary
Company 3	SAP Public Cloud	CALM	Medium	Hungary
Company 4	SAP Private Cloud	Excel	Medium	Hungary

Descriptively, the distinction between the two groups of cases in terms of testing approach is quite pronounced: projects with CALM support were generally more customised and had a broader geographical or functional scope, whereas projects that used Excel for testing were usually less complex and only involved single-country rollouts. This issue should be considered when reviewing the more detailed quantitative indicators below.

Descriptive Comparison of Testing Activities

Table 3 shows the main testing KPIs for the four projects, along with their original descriptive values. The difference in testing volumes is striking. The two CALM projects executed 1,879 and 74 test cases in total, whereas the two Excel projects only managed 27 and 82, respectively. This large gap also explains the difference in testing duration: the two CALM projects typically ran their tests for 6–18 months, whereas the two Excel projects ran their tests for only 2–4 months.

The percentage of test cases closed at go-live is relatively similar across all projects (76–91%), which indicates that, regardless of methodology, teams completed test execution as planned. However, the scope of the testing differs greatly. The two Excel-based projects managed to close a high % of their test cases, mainly because the total number of test cases was very low to begin with. Again, the number of defects logged during Hypercare (0–90 days) follows a similar trend: the two CALM projects also have a higher absolute number of defects (208 and 6) than the two Excel projects (5 and 17). At face value, this could suggest quality issues; however, the subsequent KPIs show that the higher defect numbers are driven by thorough, deep defect discovery activities conducted before go-live.

Table 3: *Key Testing KPIs of analysed implementation cases*

Company	Testing Duration (months)	Total Test Cases	Test Case Closure Rate (%)	Number of Test Cycles	Hypercare Defects	Deployment Type	Testing Tool
Company 1	18	1,879	76%	3 cycles	208	Private	CALM
Company 2	2	27	91%	1 cycle	5	Public	Excel
Company 3	6	74	85%	2 cycles	6	Public	CALM
Company 4	4	82	85%	1–2 cycles	17	Private	Excel

Derived Indicators: Testing Intensity and Defect Density

Table 4 shows two normalized indicators, testing intensity and defect density, to provide a more meaningful comparison across projects of different sizes.

Testing Intensity:

The testing intensity of the two CALM projects was more than double:

- CALM average: 58.35 test cases per month
- Excel average: 17.00 test cases per month

Table 4: *Testing Performance Metrics for the Four SAP Projects*

Company	Testing Tool	Testing Intensity (Test Cases / Month)	Hypercare Defect Density (Defects / 100 Test Cases)	Test Completion at Go-live (%)
Company 1	CALM	104.40	11.10	76%
Company 2	Excel	13.50	18.50	91%
Company 3	CALM	12.30	8.10	85%
Company 4	Excel	20.50	20.70	85%

The central part of the difference is attributed to a single country project (1,879 test cases). In contrast, the two CALM projects demonstrate more thorough and continuous testing, even when the large project is excluded. Testing in Excel was limited to brief periods in both projects, indicating not only the constraints of the methodology but also the projects' narrower testing scope.

Defect Density:

The results are reversed when we look at the number of defects found after the go-live. Lower values are to be expected in this case because they indicate better quality assurance before the release:

- Average defect density for CALM projects: 9.59 defects per 100 test cases
- Average defect density for Excel projects: 19.63 defects per 100 test cases

This supports one of the study's main conclusions: projects that use CALM for testing find a larger share of their defects before going live and therefore have fewer issues during Hypercare. In other words, testing supported by CALM moves defect detection earlier in the project timeline, making it cheaper and less disruptive to fix defects.

Statistical Comparison Between Excel and CALM Projects

Given the limited sample size (n=4), the statistical power of this analysis is restricted. However, Table 4 presents a formalised comparison with Mann–Whitney U tests to complement the descriptive analysis.

While none of the tests are statistically significant, all the effects are consistent with the advantages of CALM-based testing:

- Testing intensity: a strong effect in favour of CALM
- Defect density: an effect in favour of CALM (lower density)
- Closure rate: mixed effect, with a slight advantage for Excel due to a smaller scope

Based on the Table 4 data, we can calculate the Mann-Whitney test, although the sample sizes are small.

Table 5: *Mann–Whitney U Test Results for Key Testing Indicators*

Indicator	**CALM Mean**	**Excel Mean**	**p-value (Mann-Whitney)**	**Direction**
Testing intensity	58.36	17.00	1.000	CALM > Excel
Defect density	9.59	19.63	0.333	CALM < Excel
Closure rate	80.5%	88%	0.414	Mixed

These findings support the view that CALM's functionalities (traceability, workflow integration, analytics) contribute to more comprehensive test planning and deeper defect detection than spreadsheet-based testing.

The result of the Mann–Whitney U tests was from a minimal sample size (n = 4). Therefore, results should be considered exploratory rather than confirmatory. The p-values with these tests are not indicative of statistical significance. However, the descriptive trends suggest that projects with CALM had broader testing and lower post-go-live defect rates. The non-significant p-values indicate limited statistical power rather than the absence of substantial differences. Therefore, the statistical tests serve as a supplement to the descriptive and process-based interpretation of the results and they cannot be substituted.

Summary of findings

Looking across all five tables, the findings are consistent:

- CALM enables more comprehensive and structured testing than Excel.
- Projects that use CALM find more defects earlier, which thereby lowers the risk of production escapes.
- Excel testing may seem efficient only because the testing is shallow, not because of higher quality.
- The relative post-go-live defect density is significantly lower in CALM projects, which indicates better readiness.
- CALM's process traceability, dashboards, and workflows lead to measurable quality improvements.

Taken together, these findings suggest that testing tools matter and that organisations planning any mid- to large-scale SAP implementation should consider using SAP Cloud ALM or a similar lifecycle management platform.

Discussion / Conclusion

Discussion

Table 6 below combines qualitative observations and quantitative data to provide a broader perspective on how testing tools influence project behaviour and outcomes.

We have collected the outcomes of the research as follows:

- Traceability and Progress: Excel does not have automated traceability, and it is challenging to link defects to specific business processes. CALM's process mapping facilitated accountability, defect tracking, and readiness reporting throughout the project.
- Scalability for complex implementation: The two CALM-supported projects were able to significantly increase their testing volumes while at the same time maintaining coordination and visibility across the team. The two Excel-based projects, on the other hand, were not only limited in size but also in complexity. Test cases can be missed because CALM uses build scenarios based on previous standardized project experiences.
- Defect Management: Defect workflows in Excel were manual, and errors could be tracked manually. These ineffective ways result in delayed resolution or incomplete documentation. CALM's integrated defect management system streamlined the process, reduced duplicate efforts, improved defect closure cycles, and enabled earlier resolution.
- Hypercare stability: The combined effect of these efficiencies is most noticeable in hypercare. The two CALM projects had lower relative defect density and faster stabilisation. In comparison, the two Excel projects had fewer defects identified during testing, but more than expected after go-live.

Table 6: *Comparative Assessment of Testing Practices: Excel vs. CALM*

Aspect	**Excel-Based Testing**	**CALM-Based Testing**	**Observed Impact**
Traceability	Manual, fragmented	End-to-end automated	Earlier defect discovery in CALM projects
Test coverage	Limited scalability	High scalability	CALM projects executed 10–20× more test cases
Collaboration	Static sheets	Shared workspace	Better cross-team coordination in CALM
Defect management	Manual & error-prone	Integrated lifecycle	Faster closure cycles in CALM
Hypercare stability	Higher defect density	Lower defect density	Faster stabilisation in CALM projects

This aligns with broader theoretical arguments in the ERP literature that testing depth and governance are key factors in implementation success and long-term system stability.

Conclusion

This paper studied four SAP implementation projects to check if using different test management tools, Excel spreadsheets, or SAP Cloud ALM can affect testing efficiency and quality via thoroughness of testing, the number of defects, and stability after go-live. For all descriptive, derived, and comparative indicators, a clear pattern emerged: Testing supported by CALM is more thorough, organized, and transparent than testing with spreadsheets. Projects with CALM performed much more testing and maintained a much higher testing intensity

throughout the project months. Implementations with CALM found more defects in absolute terms; however, this was due to more rigorous, thorough testing rather than lower quality. Once accounting for testing volume, CALM projects experienced a lower defect density after go-live, indicating better risk mitigation before cutover.

Testing in Excel was only adequate for smaller, less complex rollouts. The lack of traceability, fragmented defect management, and limited scalability prevented testing from being deep enough, which led to a higher relative number of defects during hypercare. These results are consistent with the general ERP literature, which argues that the success of an implementation depends not only on the quality of the process design but also on the rigor, cadence, and governance of testing activities.

In short, the paper shows that the test tool is not only a matter of administration but also a key success factor in SAP projects. By enabling end-to-end traceability, integrated defect lifecycle management, and real-time readiness analytics, SAP Cloud ALM can make SAP implementations more predictable, transparent, and stable.

Future Research and Practical Next Steps

The outcomes of this study are indicative but highlight several opportunities for subsequent research and practical application.

Potential Research in the Future

- Broader multi-sector sample: Subsequent research could investigate a wider range of SAP implementations with various industries and regions to confirm the applicability of the testing intensity and defect resolution trends.
- Longitudinal study: A Longitudinal study (covering all phases of the project) could identify changes in test coverage, defect leakage, and stabilisation speed more clearly than a retrospective study.
- Further development of the testing model: Some aspects can be added to the model, like risk-based testing weights or process criticality scores. The further development of this model would enable organisations to predict testing sufficiency and readiness with greater precision.
- The role of AI and predictive analytics: Currently, there is embedded AI in CALM. This can be used for automated test generation, defect prediction, and readiness scoring. This would link ERP quality assurance to advanced decision-support techniques.

Suggestions for Practice

- Moving from Excel-based test management to integrated ALM platforms: In the case of companies still using Excel for test management. If they implement a broader scope with many modules and test cases, CALM can be the right solution from a testing perspective.
- Readiness for the design phase: The Project team can invest earlier in traceability mapping, defect categorization, test data creation, and end-to-end design.
- Risk-based testing and test KPI dashboards: Test KPIs, including defect density, leakage, and testing intensity, should be tracked regularly. CALM dashboards lay a strong foundation for such tracking.
- Define governance roles: Although some of the implementation team have these project-related roles, there are still many who do not focus on the right capabilities. CALM requires a clear role for each: the test manager, the defect coordinator, and the process owner.

- Using the hypercare period to estimate quality: The number of defects in the hypercare period is a solid indicator of deep testing. Companies should use it to demonstrate the maturity of project testing during the execution phase.

References

Aires, M., & Abrantes, R. (2022). Requirements elicitation in ERP implementation process. *Procedia Computer Science, 204*, 794–802. https://doi.org/10.1016/j.procs.2022.08.096

Al-Fawaz, K., Al-Salti, Z., & Eldabi, T. (2008). Critical success factors in ERP implementation: A review. In *Proceedings of the European and Mediterranean Conference on Information Systems 2008 (EMCIS 2008)*, Dubai, United Arab Emirates, May 25–26.

Banța, V.-C. (2020). SAP MM/WMS – Implementation of an ERP system – Analysis and recommendations in the IT unit testing phase: A case study (III). *Annals of the Constantin Brâncuşi University of Târgu Jiu, Economy Series* (*1)*, 16–21.

Bradford, M., & Florin, J. (2003). Examining the role of innovation diffusion factors on the implementation success of enterprise resource planning systems. *International Journal of Accounting Information Systems, 4*(3), 205–225. https://doi.org/10.1016/S1467-0895(03)00026-5

Esteves, J., & Pastor, J. (2001). Enterprise Resource Planning systems research: An annotated bibliography. *Communications of the Association for Information Systems, 7*(1), 1–52. https://doi.org/10.17705/1CAIS.00708

Gattiker, T. F., & Goodhue, D. L. (2005). What happens after ERP implementation: Understanding the impact of interdependence and differentiation on plant-level outcomes. *MIS Quarterly, 29*(3), 559–585. https://doi.org/10.2307/25148695

Gupta, V. (2025). Driving global SAP transformation: A case study of 80+ enterprise implementations. *American Journal of Multidisciplinary AI & Technology, 1*(2).

Hitt, L. M., Wu, D. J., & Zhou, X. (2002). Investment in enterprise resource planning: Business impact and productivity measures. *Journal of Management Information Systems, 19*(1), 71–98. https://doi.org/10.1080/07421222.2002.11045716

Idilbi, M., & Abu-Shanab, E. A. (2022). Critical success factors for ERP implementation: Two directions focusing on employee perceptions in Qatar. *International Journal of Technology and Human Interaction, 18*(1), 1–22. https://doi.org/10.4018/IJTHI.297613

Immidi, K., & Mane, S. (2025). Integration of artificial intelligence with SAP integrated business planning (IBP): A technical analysis of implementation success and performance metrics across industries. *International Journal of Engineering Research & Technology, 14*(2).

Kouriati, A., Moulogianni, C., Kountios, G., Bournaris, T., Dimitriadou, E., & Papadavid, G. (2022). Evaluation of critical success factors for enterprise resource planning implementation using quantitative methods in agricultural processing companies. *Sustainability, 14*(11), 6606. https://doi.org/10.3390/su14116606

Madathala, H., & Yeturi, G. (2025). Navigating SAP ERP implementation: Identifying success drivers and pitfalls. In *Proceedings of the Third International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT-2025)* (pp. 75–83). IEEE. https://doi.org/10.1109/IDCIOT64235.2025.10914890

Maditinos, D., Chatzoudes, D., & Tsairidis, C. (2012). Factors affecting ERP system implementation effectiveness. *Journal of Enterprise Information Management, 25*(1), 60–78. https://doi.org/10.1108/17410391211192161

Mahraz, M.-I., Benabbou, L., & Berrado, A. (2019). *Success factors for ERP implementation: A systematic literature review*. In *Proceedings of the 9th Annual International Conference on Industrial Engineering and Operations Management*. IEOM Society International, Bangkok, Thailand, March 5–7.

Salas, W. H. (2023). Model to improve an ERP implementation based on agile best practice: A Delphi study. *Procedia Computer Science, 219*, 1785–1792.https://doi.org/10.1016/j.procs.2023.01.474

Shafi, R., Ahmad, N., Nawab, M., Bhatti, K. A., Shad, M. K., Hameed, W. U., Asif, M., & Shoaib, M. (2019). Measuring performance through enterprise resource planning system implementation. *IEEE Access, 7*, 6696-6702. https://doi.org/10.1109/ACCESS.2018.2884900

Shatat, A. S., & Dana, N. (2016). Critical success factors across the stages of ERP system implementation in Sohar University: A case study. *International Journal of Management and Applied Research, 3*(1), 30–47. https://doi.org/10.18646/2056.31.16-003

Vaid, A., Reddy, C., & Prabhagaran, S. (2024). A hybrid framework for dynamic clustering and anomaly detection in SAP ERP systems. *International Journal of Computer Science and Mobile Computing, 13*(12), 23–34. https://doi.org/10.47760/ijcsmc.2024.v13i12.003

Viljakainen, M. (2024). *Utilization of project implementation methodology in ERP implementation projects* (Master's Thesis). Lappeenranta–Lahti University of Technology. LUT]. LUTPub. https://lutpub.lut.fi/handle/10024/167312

Wynn, M., Ilyas, J., Isleyen, O. F., Brüntrup, H., & Metin, B. (2024). Reassessing critical success factors for ERP implementation in the digital era. *Digital Technologies Research and Application, 3*(2), 117-131 https://doi.org/10.54963/dtra.v3i2.297