

Automating Non-Taxable Lease Billing: An Oracle E-Business Suite R12 Case Study

Abstract

Background: Enterprise Resource Planning upgrades can introduce operational risks when functionality relied upon by business users is removed or reorganized. This case study examines a non-taxable lease billing gap encountered during migration from Oracle Lease Management 11i to Oracle Lease and Financial Management R12. The removal of stream-level tax applicability controls created potential exposure to incorrect invoice tax, customer disputes, receivable inaccuracies, and manual remediation. **Methods:** A business-focused tax determination framework combined Oracle E-Business Tax configuration, a dedicated zero-percent tax code, a custom user-defined fiscal classification, and automated Accounts Receivable interface processing. A custom concurrent program identified designated non-taxable lease streams using lease context and interface attributes, updated tax-related information before AutoInvoice, and preserved standard invoice processing. The paper describes the solution architecture, implementation decisions, validation considerations, and financial control implications using an enterprise implementation case study. **Results:** The implemented approach restored non-taxable billing treatment without changes to Oracle core application code. The project reported approximately 50 hours of avoided manual effort per month across finance and support activities. Separately, scheduled audit evidence reporting reduced a reported quarterly turnaround from approximately 40 hours to one hour, equivalent to a 97.5 percent reduction. The solution supported geographically distributed lease operations and improved the consistency of tax-related processing and reporting. **Conclusion:** The case illustrates how targeted interface automation and rule-based tax configuration can address an ERP upgrade functional gap while retaining a supported application processing path. Its lessons concern control design, traceability, maintainability, and careful validation rather than generalizable estimates of performance across organizations.

Keywords: Oracle Lease Management; Oracle E-Business Tax; ERP Transformation; Tax Automation; Lease Billing; Financial Systems Integration.

I. INTRODUCTION

Enterprise Resource Planning (ERP) upgrades are often undertaken to improve scalability, regulatory compliance, reporting capabilities, and operational efficiency. While such upgrades

provide significant long-term benefits, they can also introduce unexpected functional gaps that affect critical business processes. Organizations operating large-scale financial systems frequently encounter situations where legacy functionality relied upon by business users is modified, relocated, or eliminated during migration to newer software versions. Managing these transitions requires not only technical expertise but also a deep understanding of business operations and financial controls.

Oracle Lease Management represents a specialized component within the Oracle E-Business Suite ecosystem, supporting the lifecycle management of leased assets from contract origination through billing, accounting, asset disposition, and financial reporting. In equipment leasing organizations, lease billing accuracy is particularly important because invoice generation directly affects customer satisfaction, revenue recognition, receivable balances, tax reporting, and regulatory compliance.

The work presented in this paper originated during a large-scale Oracle upgrade initiative involving the migration of Oracle Lease Management 11i to Oracle Lease and Financial Management R12 (OLFM R12). The client organization used Oracle Lease Management 11i as the central platform for managing equipment lease operations across multiple business regions. The system supported lease origination, contract administration, customer billing, asset lease management, asset disposition, and accounting processes.

Although the R12 platform introduced significant architectural and functional improvements, the upgrade exposed a critical business challenge involving non-taxable lease charges. In Oracle Lease Management 11i, business users could explicitly designate individual billing streams as taxable or non-taxable. This capability disappeared in the upgraded environment, creating the possibility of incorrect tax calculations for several lease-related charges. Without an effective solution, the organization faced risks including overstated receivables, inaccurate tax reporting, revenue accounting discrepancies, customer disputes, audit concerns, and substantial manual correction efforts.

This paper describes the business problem, solution architecture, implementation strategy, leadership contribution, and measurable organizational benefits resulting from the development of a custom tax determination and automation framework designed specifically to restore non-taxable lease billing functionality within Oracle Lease and Financial Management R12.

Research objective and case-study boundaries

The research question is how an organization can restore a business-critical tax outcome when an ERP upgrade removes the original user-facing control. The unit of analysis is a lease-billing tax determination process, not the full Oracle R12 migration. The paper therefore focuses on the handoff between lease billing, receivables interface processing, E-Business Tax rules, and invoice creation. It describes the implemented design and the outcomes reported by the project; it does not claim that all possible lease tax jurisdictions, contract types, or Oracle configurations were evaluated.

A useful distinction is between restoring the prior user interface and restoring the prior business rule. Reintroducing an equivalent checkbox inside the upgraded lease application could create a fragile dependency on custom application changes. By contrast, transferring the tax decision to a supported classification and rule engine preserves the business intent while respecting the architecture of the target release. This distinction frames the design choices assessed in the following sections.

II. BUSINESS CHALLENGE AND PROBLEM STATEMENT

The challenge emerged during the Oracle Lease Management upgrade project, where significant changes were introduced to the application's taxation architecture. Under the Oracle Lease Management 11i environment, lease-related customer charges were managed through configurable billing streams. Each stream represented a scheduled financial event containing billing amounts, accounting details, and processing dates.

Examples of lease-related billing streams included security deposits, document fees, service charges, maintenance fees, delivery charges, and passthrough fees. Some of these charges were taxable, while others were explicitly designated as non-taxable depending on regulatory requirements and business policies.

In Oracle Lease Management 11i, business users could easily manage tax applicability through a stream-level taxable checkbox. When the checkbox was disabled, Oracle's integrated tax engine recognized the billing stream as non-taxable and prevented tax calculation during invoice generation. This functionality provided a simple yet highly effective mechanism for controlling taxation behavior directly within the lease management application.

The migration to Oracle Lease and Financial Management R12 fundamentally changed this process. Oracle centralized tax management within Oracle E-Business Tax (EBTax), introducing a more sophisticated tax architecture intended to provide consistent tax determination across multiple applications. While this modernization improved overall tax governance, it also eliminated the stream-level taxable option previously available within Oracle Lease Management.

The removal of this feature created a significant operational problem. Lease users could no longer designate individual streams as non-taxable within the lease application itself. Although Oracle E-Business Tax provided numerous tax attributes such as tax classification codes, product fiscal classifications, and user-defined fiscal attributes, none of these options directly replicated the stream-level tax applicability functionality previously available in Oracle 11i.

As a result, the upgraded system lacked an out-of-the-box mechanism for identifying specific lease streams as non-taxable. If left unresolved, the issue would have produced widespread operational and financial consequences. Customer invoices could contain incorrectly calculated taxes, resulting in overstated receivable balances and inaccurate revenue reporting. Tax liabilities could be misrepresented within the general ledger, potentially affecting compliance reporting and financial statement accuracy. Customers receiving incorrect invoices would likely raise disputes, creating additional workload for customer service and finance teams. Corrective actions would require credit memos, invoice reversals, rebilling activities, and extensive reconciliation efforts.

Business analysis estimated that the cumulative impact of these manual correction activities could exceed fifty hours per month across Finance, Accounting, Business Technology, and operational support teams. Given the scale of lease operations across the Americas, Europe, and Asia-Pacific regions, a sustainable and automated solution became essential.

Tax applicability and transaction context

A lease contract can generate several economically different streams even when those streams appear together in a customer billing cycle. Whether a charge is taxable depends on the relevant transaction facts and applicable policy or jurisdiction; a descriptive stream name alone is not a universal tax determination rule. The implementation therefore needed a reliable way to identify the specific streams that the organization had already designated for non-

taxable treatment. This identification had to survive the transition from the lease source application to the receivables interface and then to the tax engine.

The operational failure mode was not limited to an incorrect tax amount on a single invoice. An incorrect invoice can lead to customer inquiries, credit memo processing, rebilling, settlement differences, and extra reconciliations. Each corrective transaction creates additional processing and review work, while a poorly documented manual workaround can undermine consistency across business regions. These dependencies explain why a small missing control in an upstream application may create disproportionate downstream effects.

III. METHODS AND SOLUTION ARCHITECTURE

Case-study method and evidence boundaries

This paper uses a descriptive enterprise implementation case-study approach. Its evidence consists of the stated legacy-to-target functionality difference, the implemented configuration and interface-processing design, and the operational outcomes reported for the project. The design is analyzed as a sequence of decisions: define the business classification, map that classification to a tax result, identify eligible interface transactions, update those transactions before invoice import, and validate the resulting invoice and accounting behavior. No controlled experimental comparison, independently audited savings dataset, or jurisdiction-by-jurisdiction tax study is presented. Consequently, reported effort reductions should be interpreted as project outcomes rather than causal estimates that would automatically apply to other organizations.

Functional and nonfunctional requirements

The functional requirement was to produce the intended zero-tax outcome for specifically designated non-taxable streams while leaving taxable streams subject to the normal tax determination process. Related requirements included correct identification of originating lease streams, consistency between the source billing event and the receivables interface, and compatibility with AutoInvoice. Nonfunctional requirements included maintainability, repeatable batch execution, traceability of classification decisions, separation of configuration from custom logic, and minimal interference with delivered Oracle code. These criteria explain why a wrapper program and E-Business Tax configuration were used together rather than relying on either alone.

Design alternatives and rationale

Several conceptual alternatives can be distinguished. A manual correction approach would leave invoice creation exposed to inconsistent treatment and would shift work downstream. A direct customization of the delivered lease application could reproduce the old user experience but increase upgrade and support dependencies. A configuration-only approach would be attractive if all required transaction attributes reached the tax engine in usable form; the case indicates that the available standard options did not directly reproduce the prior stream-level control. The selected hybrid approach retained standard tax calculation while adding narrowly scoped logic to populate the classification needed for the intended result. These are design tradeoffs, not evidence that every alternative was formally prototyped during the project.

Architecture and data-flow rationale

The framework can be understood as a controlled translation between two different tax decision models. The legacy process relied on a taxability designation attached to a lease billing stream. The upgraded process required tax attributes that could be interpreted by Oracle E-Business Tax during receivables processing. The solution therefore separated the identification of a lease charge from the calculation of its tax result. The lease context was used to identify eligible charges; the user-defined fiscal classification expressed the business decision in a form recognizable by the tax engine; and the tax rule produced the configured zero-percent outcome. This separation was important because it allowed lease-specific identification logic to coexist with centralized tax calculation rather than bypassing it.

Sequence of automated processing

The proposed processing sequence begins after lease billing information has been prepared for transfer into Accounts Receivable and before the AutoInvoice import runs. First, the wrapper program examines the available interface records and identifies those associated with the relevant lease streams. Second, it evaluates lease-specific context and interface attributes to distinguish designated non-taxable charges from other invoice lines. Third, it assigns the NON-TAXABLE STREAM classification and the corresponding tax-related attributes to the identified records. Fourth, it addresses associated tax interface lines so that the intended zero-percent tax treatment is represented consistently. Finally, AutoInvoice processes the prepared interface data using the configured E-Business Tax rules. This sequence preserves the existing

standard invoice import step while inserting a focused decision layer at the point where lease information is translated into receivables transactions.

Design boundaries and maintainability

A significant architectural choice was to avoid modifying Oracle's delivered core programs. Customization was concentrated in a separate concurrent process and configuration objects. This boundary offers a practical maintenance advantage: changes to lease-specific eligibility rules can be assessed separately from the standard receivables import process. It also creates a clearer support model in which administrators can review tax configurations, functional analysts can review the classification rules, and technical teams can investigate interface processing. The design does not eliminate the need for regression testing after future upgrades; rather, it limits the custom surface area that must be examined. Changes to interface attributes, tax configuration, or the timing of standard programs could still affect the framework and should be assessed through controlled release management.

Rather than attempting to recreate Oracle 11i functionality through extensive customization, the solution was designed to leverage Oracle's new taxation architecture while restoring the business outcome previously achieved through stream-level tax control.

The first component of the solution involved configuring a dedicated 0% tax code within Oracle E-Business Tax. This tax code was designed exclusively for lease streams that should not incur taxation. By creating a formal tax configuration rather than bypassing the tax engine entirely, the solution remained fully aligned with Oracle's centralized tax strategy.

The second component involved the creation of a custom User Defined Fiscal Classification named NON-TAXABLE STREAM. This classification served as a business identifier capable of distinguishing non-taxable lease charges from standard taxable billing streams.

Tax determination rules were subsequently configured within Oracle E-Business Tax to associate the newly created fiscal classification with the 0% tax code. This design effectively transformed the fiscal classification into a tax decision driver, allowing Oracle's tax engine to calculate a zero-tax outcome whenever the designated classification was encountered.

The final and most innovative component involved the development of a custom concurrent program called Lease Update Tax Code for Non-Taxable Billing Streams. This program functioned as a wrapper process executed immediately before Oracle's AutoInvoice import program.

The custom process examined records within Oracle Accounts Receivable interface tables, particularly RA_INTERFACE_LINES_ALL. Using lease-specific context values and interface attributes, the program identified invoice lines originating from non-taxable lease streams. Once identified, the solution automatically populated the custom NON-TAXABLE STREAM fiscal classification and updated corresponding tax-related information.

The program also identified tax interface lines and applied the appropriate 0% tax rate. Consequently, when Oracle AutoInvoice executed, the standard tax engine calculated a tax amount of zero while still processing transactions through Oracle's supported taxation framework.

This approach restored the intended business functionality without modifying Oracle's core applications and without compromising upgrade compatibility.

Interface processing and data integrity considerations

The custom program operated before AutoInvoice, making the interface a practical intervention point. This placement matters because it separates transaction enrichment from final invoice generation: the program can classify eligible lines while the delivered invoice import and tax-processing components remain responsible for their normal functions. The processing sequence should preserve the relationship between each source lease stream and the resulting receivables line. A misidentified line could receive a zero rate when it should be taxable, whereas a missed eligible line could receive an unintended tax calculation. The identification logic is therefore as important as the tax rule itself.

For reproducibility, an implementation specification should define the source attributes used to identify the lease stream, the permitted values of the non-taxable classification, the mapping to the zero-percent tax treatment, and the point at which interface rows are updated. It should also specify how already-processed or rejected rows are handled and how processing outcomes are logged. The case establishes the use of RA_INTERFACE_LINES_ALL and tax-related interface information, but does not provide complete field mappings, program source code, or a database schema. Those implementation details should not be inferred from the narrative alone.

IV. IMPLEMENTATION AND VALIDATION

Requirements and control alignment

The implementation challenge was not solely to calculate a zero tax amount. The business requirement was to preserve a reliable and auditable distinction between charges that should and should not be taxed under the organization's applicable policies. That distinction requires functional agreement about eligible billing streams, consistent interpretation across regions, and a mechanism to prevent manual intervention from becoming the default operating model. The solution design therefore had to connect lease operations, tax determination, receivables, and downstream accounting. Finance and tax stakeholders were essential to defining the expected outcome, while technology teams were responsible for translating the rule into repeatable processing. The resulting division of responsibilities supported both operational continuity and accountability for tax treatment.

Validation methodology

A robust validation approach for this type of framework compares the expected tax treatment of representative lease billing streams against the actual invoice and accounting outcomes. Test cases should include charges designated as non-taxable, charges that remain taxable, combinations of taxable and non-taxable lines on the same invoice, and transactions processed in each relevant region. The validation should trace each case from the lease billing source through interface attributes, classification assignment, AutoInvoice results, and receivables accounting. It should also verify that eligible lines receive the configured zero-percent treatment without altering unrelated charges. Exception scenarios, including missing interface attributes, unexpected stream identifiers, repeated program execution, and rejected invoice imports, deserve specific attention. These are recommended validation dimensions for a similar implementation; the available case documentation does not provide a detailed test-execution log for every scenario.

Operational monitoring and exception handling

Once the framework is operating, controls should focus on whether each scheduled billing batch is processed completely and consistently. A support team can compare the number of eligible lease lines identified with the number updated successfully, review concurrent-program completion statuses, and investigate interface records rejected by AutoInvoice. Monitoring should distinguish genuine tax exceptions from routine zero-tax outcomes. A zero

calculated tax amount alone does not establish that the correct classification was applied, because multiple tax conditions may produce the same numerical result. Retaining processing logs, configuration change records, and evidence of exception resolution would strengthen traceability. These monitoring practices represent a governance model derived from the solution architecture rather than additional quantified outcomes claimed for the original project.

The initiative was executed while serving as the Financial Technology Lead responsible for the Oracle Lease Management upgrade project in Sydney, Australia. The role extended beyond system implementation and involved ownership of the broader transformation effort associated with migrating lease operations from Oracle 11i to Oracle R12.

Key responsibilities included coordinating finance stakeholders, leading functional design discussions, educating business users regarding architectural changes introduced in R12, supporting User Acceptance Testing, validating accounting outcomes, and assisting finance teams during post-go-live operations.

One of the most critical decisions involved recognizing that the issue could not be resolved solely through standard tax configuration. The challenge required a hybrid solution combining Oracle E-Business Tax capabilities with custom process automation. Balancing system integrity, business usability, maintainability, and compliance requirements required extensive collaboration across finance, accounting, and technology teams.

In addition to solution design responsibilities, significant effort was dedicated to organizational change management. Training sessions were conducted to explain R12 taxation behavior, lease accounting impacts, Subledger Accounting changes, and General Ledger integration enhancements. Business users were guided through the differences between the legacy and upgraded environments to ensure successful adoption.

The project ultimately became a showcase of how functional leadership and technical innovation can be combined to address complex ERP transformation challenges while minimizing disruption to business operations.

Proposed validation matrix and acceptance criteria

The following validation dimensions provide a transparent way to assess a solution of this type; they are methodological recommendations and are not represented as a retrospective inventory of executed test cases. First, test a designated non-taxable stream and verify that the

imported invoice reflects the expected zero-tax outcome. Second, test an ordinary taxable stream to ensure that the custom process does not override the standard tax result. Third, test invoices containing both types of streams to confirm line-level selectivity. Fourth, test missing or ambiguous classification attributes to establish that exceptions are visible and resolved rather than silently assigned an unsupported treatment.

Additional scenarios should include repeated batch execution, rejected AutoInvoice lines, credit and rebill activity, adjustments to source stream attributes, and reconciliation of invoice amounts to originating lease schedules. Where multiple business regions are involved, test data should represent the different configured tax regimes and local policies without assuming that the same tax result applies globally. The expected result for each scenario should be approved by the responsible tax and finance stakeholders before execution. A passed technical import is not sufficient evidence of correct tax determination; the invoice and accounting outcomes must also be inspected.

Operational controls and ownership

A sustainable process requires clear ownership of both the classification policy and the interface automation. Tax or finance owners should determine which charge categories qualify for the non-taxable designation under the applicable rules. Application administrators should control changes to E-Business Tax setup. Technical support should own program scheduling, error handling, and interface processing health. Reconciliations should compare eligible source streams, processed interface lines, and resulting invoices so that omitted, duplicated, or incorrectly classified items can be investigated. The appropriate frequency and thresholds depend on billing volumes and the organization's risk assessment.

Change management is particularly important because tax configuration can be modified without changing custom program code. A rule change may affect existing classifications, and a source-system change may alter the interface attributes used by the wrapper program. Regression testing should therefore cover both configuration and code changes, including tax exceptions and cross-region billing scenarios. Retained test evidence should identify the configuration version, transaction inputs, expected outcome, actual invoice result, and reviewer approval. These practices support auditability but are not claimed as additional quantified project results.

V. RESULTS

Interpreting the reported efficiency gains

The reported saving of approximately fifty hours per month reflects avoided manual activities associated with invoice corrections, tax adjustments, disputes, and reconciliations. Expressed as a simple annualized illustration, that monthly saving corresponds to roughly six hundred hours over twelve months if the same operating conditions persist. This annualized figure is an arithmetic extrapolation, not a separately measured project result. The audit preparation improvement is distinct: the paper reports a reduction from approximately forty hours to one hour per quarter, equivalent to a 97.5 percent decrease in preparation time. These two measures should not be added together without first establishing whether any activities overlap. Their value lies in showing that preventive automation can reduce both operational remediation and the effort required to assemble evidence for control reviews.

Financial integrity and governance implications

Tax treatment errors can propagate through a financial process even when the originating invoice line appears small. An incorrectly taxed charge may create an incorrect customer balance, require a credit memo or rebilling, and lead to reconciliation work between receivables, accounting, and tax reporting. The case illustrates the value of preventing such errors before invoice creation rather than relying on downstream adjustments. The user-defined classification also provides an explicit link between a business rule and the tax engine's determination, which can support review and explanation of the processing outcome. Nevertheless, a configured zero-percent rate must not be treated as a universal substitute for legal tax analysis. The eligibility of a particular lease charge depends on jurisdiction, transaction facts, and the organization's approved tax policy.

The implemented framework delivered substantial operational and financial benefits. Most notably, it restored accurate non-taxable billing functionality that had been lost during the Oracle upgrade. Lease invoices for security deposits, document fees, maintenance charges, service fees, and other designated non-taxable streams were generated correctly without requiring manual intervention.

The automation eliminated an estimated fifty hours of manual effort each month across Finance, Accounting, Business Technology, and operational support teams. These savings

were realized through the prevention of invoice corrections, dispute management, tax adjustments, reconciliation activities, and reporting efforts.

The solution also improved financial accuracy across multiple reporting domains. Customer invoices reflected appropriate tax treatment, Accounts Receivable balances remained accurate, revenue reports reflected correct values, and tax liabilities were calculated appropriately. By preventing errors at the source, the organization avoided downstream correction activities that would otherwise affect multiple reporting systems.

Adoption extended across global operations, with planned usage supporting more than fifty users spanning Finance, Accounting, Business Technology, internal SOX audit teams, and external auditors across multiple geographic regions. The framework became institutionalized within enterprise finance processes and supported routine operational activities.

An additional benefit emerged in audit and compliance operations. Automated reports and scheduled reporting capabilities significantly improved evidence collection for SOX reviews. Activities that previously required approximately forty hours per quarter were reduced to roughly one hour, representing a turnaround-time improvement of approximately 97.5 percent. Audit teams could access scheduled reports directly without relying on manual responses from operational teams, improving both efficiency and governance.

The solution also received recognition from finance leadership, business stakeholders, and technology management. Appreciation emails, internal recommendations, and executive acknowledgments highlighted the project's contribution to operational efficiency, compliance support, and business continuity.

Outcome interpretation

The case reports two distinct efficiency outcomes. The first is an estimated 50 hours of manual work avoided per month by preventing tax-related invoice correction and follow-up activity. The second concerns scheduled audit evidence reporting, for which quarterly turnaround reportedly declined from about 40 hours to one hour. These figures describe different workflows and should not be added together as though they measured the same task. Neither figure, by itself, establishes a monetary return on investment: such a calculation would require validated labor rates, implementation costs, maintenance costs, and a consistent observation period.

VI. DISCUSSION

Transferability and limitations

The broader lesson extends beyond Oracle Lease Management. Enterprise upgrades often move a business decision from a specialized application screen into a centralized service or rules engine. When this happens, a successful replacement may require an interface-level translation of business context rather than a literal recreation of the old screen or checkbox. The approach described here may inform similar projects involving billing, tax, or other shared enterprise services. However, the reported outcomes come from one implementation context and should not be generalized as guaranteed savings. The paper does not provide a controlled comparative experiment, detailed regional transaction volumes, or a complete breakdown of implementation costs. Future evaluations could measure error rates before and after deployment, quantify the cost of maintaining the custom process, and examine how the framework performs after subsequent application updates.

Implications for ERP modernization

This case illustrates a recurring modernization problem: platform architecture may become more centralized while business users lose a direct control that previously encoded an important policy decision. The appropriate response is not necessarily to recreate the old interface. Instead, the process owner can identify the underlying decision, determine where the new architecture expects that decision to be expressed, and bridge missing transaction context through a constrained integration. In this case, the bridge was the fiscal classification populated before AutoInvoice. The broader contribution is a design pattern for preserving business semantics across application boundaries.

The design also exposes a governance tradeoff. Centralized tax rules can improve consistency, but they require stronger definition of classification inputs and greater coordination among teams responsible for source transactions, tax policy, and invoice processing. An automated zero-rate outcome is only as reliable as the underlying designation. Accordingly, the framework should not be interpreted as a general rule that security deposits, fees, or other named streams are always non-taxable; the applicable determination remains dependent on jurisdiction and transaction facts.

Limitations and future evaluation

The paper reports a single enterprise case rather than a multi-organization comparative study. It does not provide invoice-volume distributions, a before-and-after error-rate series, independent audit measurements, or the complete implementation code. These limitations constrain causal claims about the amount of benefit attributable solely to the automation. Future evaluation could examine exception rates before and after deployment, the proportion of invoices requiring tax corrections, classification accuracy across regions, support ticket volumes, and the maintenance effort associated with subsequent Oracle updates. Such evidence would help distinguish the value of the architecture from local operating conditions.

A further limitation concerns the difference between a zero-percent tax rate and a legal determination of exemption or non-taxability. The implemented zero-rate configuration was a mechanism for producing the required billing outcome in the described environment; it should not be generalized as a universal compliance treatment. Organizations adapting the pattern should validate tax reporting, exemption documentation, invoice presentation, and audit requirements with qualified tax specialists. These checks are especially relevant when a global implementation spans different tax regimes.

VII. CONCLUSION

This paper presented a tax determination and automation framework developed to address a critical functional gap introduced during the migration from Oracle Lease Management 11i to Oracle Lease and Financial Management R12. By combining Oracle E-Business Tax configuration, custom fiscal classifications, automated interface processing, and standard Oracle invoicing functionality, the solution successfully restored non-taxable lease billing capabilities while preserving compliance with Oracle's upgraded tax architecture.

The implementation eliminated approximately fifty hours of manual effort per month, improved invoice accuracy, strengthened financial reporting integrity, enhanced audit readiness, and supported global lease operations. More importantly, it demonstrated how thoughtful system design and functional leadership can transform a potentially disruptive ERP upgrade challenge into an opportunity for operational improvement. The case highlights the importance of business-focused innovation within enterprise systems and illustrates how targeted solutions can deliver lasting organizational value across finance, accounting, compliance, and technology domains.

REFERENCES

- [1] T. H. Davenport, *Process Innovation: Reengineering Work Through Information Technology*. Boston, MA, USA: Harvard Business School Press, 1993.
- [2] T. H. Davenport, *Mission Critical: Realizing the Promise of Enterprise Systems*. Boston, MA, USA: Harvard Business School Press, 2000.
- [3] M. Hammer and J. Champy, *Reengineering the Corporation: A Manifesto for Business Revolution*. New York, NY, USA: Harper Business, 1993.
- [4] W. M. P. van der Aalst, *Process Mining: Data Science in Action*, 2nd ed. Berlin, Germany: Springer, 2016.
- [5] W. M. P. van der Aalst and K. M. van Hee, *Workflow Management: Models, Methods, and Systems*. Cambridge, MA, USA: MIT Press, 2004.
- [6] K. C. Laudon and J. P. Laudon, *Management Information Systems: Managing the Digital Firm*, 16th ed. Pearson, 2022.
- [7] E. Turban, R. Sharda, and D. Delen, *Business Intelligence, Analytics, and Data Science: A Managerial Perspective*. Pearson, 2021.
- [8] M. Lacity and L. Willcocks, "Robotic Process Automation and Risk Mitigation," *Journal of Information Technology Teaching Cases*, vol. 8, no. 2, pp. 70–79, 2018.
- [9] S. G. Sutton, "The Changing Face of Accounting in an Information Technology Dominated World," *International Journal of Accounting Information Systems*, vol. 1, no. 1, pp. 1–8, 2000.
- [10] V. Kumar and J. van Hillegersberg, *ERP Experiences and Evolution*. Hershey, PA, USA: IGI Global, 2000.
- [11] M. Hammer, "What Is Business Process Management?," in *Handbook on Business Process Management*, Springer, 2010.
- [12] J. G. Mooney, V. Gurbaxani, and K. L. Kraemer, "A Process-Oriented Framework for Assessing the Business Value of Information Technology," *The DATABASE for Advances in Information Systems*, vol. 27, no. 2, pp. 68–81, 1996.