Multi-Agent System for Dynamic Business KPI Selection, Evaluation and Quantification Based on Oracle EBS

Geno Stefanov * and Valentin Kisimov

Posted Date: 2 May 2026

doi: 10.20944/preprints202604.2193.v1

Keywords: ERP analytics; Model Context Protocol; multi-agent systems; KPI generation; Oracle EBS; LLM

 doi:10.20944/preprints202604.2193.v1

Article

Multi-Agent System for Dynamic Business KPI Selection, Evaluation and Quantification Based on Oracle EBS

Geno Stefanov * and Valentin Kisimov

Information Technologies and Communications Department, University of National and World Economy, Sofia Bulgaria

* Correspondence: genostefanov@unwe.bg; Tel.: +359883350762

Abstract

The growing complexity of enterprise resource planning (ERP) systems necessitates intelligent approaches for dynamically identifying and evaluating key performance indicators (KPIs) that accurately reflect organizational performance. This paper proposes a multi-agent architecture for dynamic KPI management over Oracle E-Business Suite (EBS). The core design combines a dynamic multi-agent analytics layer, an extendable dedicated EBS KPI Model Context Protocol (MCP) server layer, and a data layer. The dynamic multi-agent analytics layer defines a set of independent large language model (LLM) agents, each responsible for a specific task determined by the business requirements of a particular company. The EBS KPI MCP server layer defines the tools required to access and transform Oracle EBS data and exposes them to the AI agents in the upper layer. Above these layers is the user layer, where the user actively participates in the process through a human-in-the-loop approach. Based on this general architecture, we proposed and implemented, as a proof of concept (PoC), a multi-agent system for dynamic business KPI selection, evaluation, and quantification, in which three distinct agents for KPI selection, KPI quantification, and KPI forecasting were instantiated within the multi-agent analytics layer. This demonstrates the practical applicability of the proposed general architecture. The study contributes to intelligent business analytics by showing how coordinated LLM agents can automate KPI lifecycle activities within ERP ecosystems, enabling adaptive, data-driven performance management aligned with evolving organizational needs.

Keywords: ERP analytics; Model Context Protocol; multi-agent systems; KPI generation; Oracle EBS; LLM

1. Introduction

Enterprise resource planning (ERP) systems have evolved from transactional back-office platforms into strategic digital infrastructures that support decision-making across finance, supply chains, procurement, inventory, sales, and customer service. In particular, Oracle E-Business Suite (EBS) provides organizations with large volumes of structured operational data that can be used for performance monitoring and business analysis. In parallel, recent progress in artificial intelligence, especially in large language models (LLMs), has created new opportunities for automating analytical reasoning, natural-language interpretation, and decision support [1,2]. Together, these developments open important possibilities for intelligent KPI management, but they also raise new questions about how to design reliable, adaptable, and controllable AI-driven analytical systems.

A particularly important challenge in this context is the dynamic management of key performance indicators (KPIs). KPIs play a central role in evaluating organizational performance by translating raw enterprise data into actionable managerial insight [3]. Traditionally, KPI design and performance measurement have been associated with structured managerial frameworks such as the

 doi:10.20944/preprints202604.2193.v1

Balanced Scorecard introduced by Kaplan and Norton [4]. This framework links strategic objectives to measurable outcomes across multiple perspectives and remains one of the most influential approaches to performance management [5]. However, despite their conceptual value, such approaches are often implemented in a static and expert-driven way, relying on predefined indicators that may no longer correspond to current business priorities, process changes, or available enterprise data [5,6]. As ERP ecosystems become more complex and data-intensive, the limitations of static KPI systems become increasingly visible [6].

Historically, performance measurement in ERP environments has depended on manual analytical work. Analysts and domain experts must interpret business requirements, identify relevant data domains, verify data availability, define formulas, and explain the results to decision-makers. This workflow is difficult to scale in large enterprise systems, especially when multiple modules, tables, and reporting needs are involved. In practice, organizations may continue to monitor indicators that are technically computable but strategically outdated, or strategically meaningful indicators that are poorly supported by the available data. This creates a need for more adaptive approaches to KPI lifecycle management, including KPI identification, validation, quantification, and interpretation [6].

Recent advances in LLMs suggest a promising direction for addressing this complexity. LLMs have demonstrated strong capabilities in semantic interpretation, structured reasoning, and natural-language generation [1,2,7]. These properties make them suitable for analytical tasks such as interpreting business goals, proposing candidate KPIs, generating analytical logic, and explaining results. At the same time, applying LLMs directly to enterprise systems remains problematic. ERP data structures are highly specialized, semantically dense, and governed by strict access constraints. Without a controlled interface to the underlying system, LLM-based analytics may generate hallucinated schema elements, invalid query logic, or analytically unreliable outputs [8–11]. As a result, effective AI support for KPI management requires more than a single assistant model; it requires an architecture that combines semantic reasoning, controlled tool access, and explicit user oversight.

Multi-agent systems provide a natural framework for such an architecture. Rather than assigning all analytical responsibilities to a single model, multi-agent approaches decompose the workflow into coordinated tasks performed by specialized agents [12]. Recent research has shown that LLM-based multi-agent systems can improve reasoning quality, robustness, and factual reliability through role specialization, redundancy, and crossvalidation [13,14]. In enterprise analytics, this is especially relevant because KPI lifecycle management is inherently multi-stage: business requirements must be interpreted, relevant data domains identified, candidate indicators proposed, computational feasibility validated, values quantified, and future tendencies forecast. These are distinct analytical tasks that may benefit from different agent roles and different configurations depending on the company context.

A further challenge lies in connecting these reasoning agents to enterprise data in a reliable and controlled way. Oracle EBS environments are characterized by tightly interrelated schemas and business objects, while practical KPI computation often requires metadata inspection, validation of SQL logic, and controlled execution of read-only queries. To address this, the present study adopts the Model Context Protocol (MCP) as a structured interface between AI agents and Oracle EBS data. MCP enables standardized tool invocation and controlled access to external resources, thereby grounding LLM reasoning in actual enterprise structures and reducing the risk of unsupported analytical outputs [15–17]. In this setting, the protocol layer does not merely transport data; it functions as a semantic and operational boundary between AI reasoning and ERP execution.

To address the above challenges, we introduce a multi-agent architecture for dynamic KPI management over Oracle E-Business Suite. The core idea is to separate the analytical workflow into coordinated layers. At the center of the architecture is a dynamic multi-agent analytics layer, in which the number and function of independent LLM agents are defined according to the business requirements, process complexity, and available employee expertise of a particular company. Rather

 doi:10.20944/preprints202604.2193.v1

than fixing a universal agent structure in advance, the architecture allows the system to be configured differently for different organizational contexts. This is complemented by an extendable EBS KPI MCP server layer, which defines the tools required to access, validate, and transform Oracle EBS data and exposes them to the agents through a controlled interface. Below this layer is the enterprise data layer, while above it stands the user layer, where human participants remain actively involved through a human-in-the-loop process.

This architectural perspective is important because it reframes KPI management as a configurable, enterprise-specific analytical workflow rather than as a static reporting artifact. Some organizations may require only a limited number of agents focused on KPI discovery and measurement. Others may require additional agents for review, validation, explanation, forecasting, or governance support. Similarly, the MCP server layer may initially expose only basic data-access and validation tools, but in real deployments it may need to be extended with additional company-specific capabilities. In this sense, the proposed architecture is not merely a technical implementation, but a general design pattern for adaptive KPI management in ERP ecosystems.

Based on this general design, we further propose and implement, as a proof of concept, a multi-agent system for dynamic business KPI selection, evaluation, and quantification. In this instantiation, three specialized agents are defined within the multi-agent analytics layer: a KPI Selector, a KPI Quantifier, and a KPI Forecaster. Together, these agents demonstrate how the general architecture can be translated into a concrete Oracle EBS analytical solution. The proof of concept shows that the proposed architectural model is not only conceptually flexible, but also practically applicable in a realistic ERP setting.

The main contributions of this paper are as follows:

- A general multi-agent architecture for dynamic KPI management.
- A dynamic multi-agent analytics layer in which the number and role of agents are determined by company-specific business requirements and employee expertise.
- An extendable EBS KPI MCP server layer as a controlled interface between AI agents and Oracle EBS data.
- A proof-of-concept implementation with three specialized agents for KPI selection, KPI quantification, and KPI forecasting.
- A demonstration of how coordinated LLM agents can support KPI lifecycle automation in ERP ecosystems while preserving user participation and architectural extensibility.

The remainder of this paper is organized as follows. Section 2 reviews related work in KPI management, LLM-based enterprise analytics, multi-agent systems, and protocol-based enterprise integration. Section 3 presents the proposed methodology and architecture. Section 4 describes the proof-of-concept implementation over Oracle EBS. Section 5 discusses the analytical workflow and the role of the specialized agents. Section 6 examines the implications, limitations, and future extensions of the approach. Finally, Section 7 concludes the paper.

2. Related Work

This work lies at the intersection of four research areas: KPI selection and performance measurement, LLM-based access to enterprise data, multi-agent LLM systems for analytical workflows, and protocol-based tool integration for enterprise AI. While related studies address parts of this problem, the integrated perspective proposed here remains insufficiently explored.

2.1. KPI Selection and Performance Measurement

The KPI literature recognizes that identifying meaningful performance indicators is a complex task involving large candidate spaces, multiple criteria, and strong dependence on expert knowledge [18]. Traditional approaches rely on structured frameworks such as the Balanced Scorecard, which translates strategic objectives into measurable indicators across multiple dimensions [4]. More recent work has focused on systematic KPI selection, prioritization, and adaptation to organizational goals.

 doi:10.20944/preprints202604.2193.v1

These studies emphasize that effective KPI systems must balance measurability, relevance, comparability, and business interpretability [3,19].

At the same time, literature reviews have highlighted important limitations of current performance measurement systems, including lack of adaptability, insufficient validation, and strong dependence on subjective interpretation [6]. These limitations are particularly significant in ERP environments, where data structures, process logic, and reporting needs evolve continuously. In such settings, fixed KPI definitions may become outdated or poorly aligned with actual operational conditions. These observations support the need for more dynamic and data-driven approaches to KPI lifecycle management.

2.2. LLMs over Enterprise Data and ERP Systems

The application of LLMs to enterprise analytics introduces significant opportunities, but also important challenges. LLMs can interpret natural-language business requests, generate analytical descriptions, and support decision-making through contextual reasoning [1,2]. However, enterprise data environments differ substantially from the open-domain textual settings in which many of these models are trained. Research has shown that LLMs often struggle with structured enterprise data because of schema complexity, domain-specific semantics, and the need for precise data grounding [8,20,21].

Recent studies on enterprise text-to-SQL and structured-data reasoning further confirm these limitations. Models may produce syntactically plausible but semantically invalid queries, misinterpret schema relationships, or hallucinate missing attributes when no explicit metadata grounding is provided [8,9,20,22]. Research in enterprise data engineering has also emphasized the importance of contextual metadata, historical query patterns, and domain knowledge in making LLM-based analytical systems reliable [23,24]. Although early ERP-specific LLM applications demonstrate the feasibility of conversational querying and workflow support, they focus primarily on data access or process automation rather than on dynamic KPI generation, quantification, and evaluation [25,26].

2.3. Multi-Agent LLM Systems for Analytical Reasoning

Multi-agent systems offer a natural framework for decomposing complex analytical tasks into coordinated subtasks [12]. In recent work, LLM-based multi-agent architectures have shown that reasoning quality can improve when multiple agents specialize in different responsibilities and when outputs are reviewed, challenged, or refined collaboratively [13,14]. This approach is especially useful when tasks involve multiple stages of reasoning, validation, and explanation rather than a single prompt-response interaction.

Surveys of LLM-based multi-agent systems emphasize advantages such as role specialization, modularity, robustness, and scalability [27]. These properties are highly relevant for business analytics, where KPI lifecycle management includes requirement interpretation, data-domain identification, validation, quantification, and forecasting. Recent applications in finance and enterprise analytics also suggest that multi-agent architectures can improve reliability when extracting, validating, or standardizing KPI-related knowledge across heterogeneous sources [26,28]. However, existing research typically assumes a predefined set of agent roles and does not sufficiently address the need for enterprise-specific configuration of agent number and function according to company requirements and available expertise.

2.4. MCP for Enterprise Data Access

A critical challenge in enterprise AI is enabling reliable and controlled interaction between LLM-based agents and external systems. Traditional direct API or database access mechanisms are often insufficient for complex analytical workflows because they do not provide enough semantic context,

 doi:10.20944/preprints202604.2193.v1

structured validation, or governance control. This is particularly problematic in ERP environments, where access to operational data must be carefully constrained and auditable.

Recent work emphasizes the value of protocol-based integration layers for agent systems. MCP has been proposed as a standardized interface for tool invocation and structured data exchange, enabling secure and interoperable communication between models and external tools [29–31]. Related research on tool-augmented LLMs similarly highlights the importance of external tool interfaces for improving reasoning reliability and operational usefulness [15–17]. In enterprise settings, such structured interfaces can serve as both semantic bridges and governance boundaries, allowing agents to operate over real data while preserving control over how those data are accessed and transformed.

For Oracle EBS-based KPI management, this type of protocol layer is particularly important. KPI computation often requires metadata inspection, SQL validation, transformation logic, and bounded read-only execution over complex relational structures. An MCP server layer can encapsulate these operations as explicit tools, exposing only the required functionality to the agent layer and allowing the tool set to be extended when company-specific requirements demand additional capabilities.

2.5. Research Gap

The literature confirms the importance of KPI engineering, the potential of LLMs for analytical reasoning, the advantages of multi-agent architectures, and the need for structured tool access in enterprise AI. However, these areas are still largely treated separately. KPI studies mainly address methodological relevance and prioritization [18,19], LLM studies focus on reasoning and text generation over enterprise data [8,20,23,24], multi-agent studies emphasize specialization and collaboration [13,14,27], and MCP-related approaches focus on protocol-mediated tool access [29]. What remains insufficiently explored is the integration of these dimensions into a unified architecture for dynamic KPI lifecycle management in ERP environments.

To the best of our knowledge, prior work does not provide a general architecture that simultaneously supports:

- dynamic configuration of the number and role of analytical agents according to company-specific business requirements;
- controlled protocol-based access to Oracle EBS data through an extendable MCP tool layer;
- human-in-the-loop validation within the KPI lifecycle;
- KPI selection, evaluation, quantification, forecasting, etc. within one coordinated analytical framework.

This paper addresses that gap by proposing a general multi-agent architecture for dynamic KPI management over Oracle E-Business Suite and by demonstrating its practical applicability through a proof-of-concept implementation with three specialized agents for KPI selection, KPI quantification, and KPI forecasting.

3. Methodology and Architecture

This study follows a design-oriented research approach aimed at defining and validating an architectural model for dynamic KPI management in Oracle E-Business Suite environments. Rather than focusing on a single predictive model or isolated automation task, the objective is to formulate a general framework that integrates business requirement interpretation, controlled enterprise data access, configurable multi-agent reasoning, and human-in-the-loop decision support. The methodological focus is therefore placed on architectural decomposition, functional layering, and the practical realization of the KPI lifecycle within ERP ecosystems.

The starting point of the methodology is the observation that KPI management in enterprise systems is not a purely technical problem. It is simultaneously a business, analytical, and governance problem. A KPI is useful only when it reflects a relevant business objective, can be grounded in actual enterprise data, is interpretable by decision-makers, and can be maintained under evolving process

 doi:10.20944/preprints202604.2193.v1

conditions. For that reason, the proposed methodology does not assume a fixed set of indicators or a fixed analytical workflow. Instead, it adopts a layered architecture in which the analytical logic can be configured according to the needs of the target organization.

3.1. General Architectural Logic

The proposed architecture is organized into four layers: a user layer, a dynamic multi-agent analytics layer, an extendable EBS KPI MCP server layer, and a data layer (Figure 1). These layers form a structured analytical pipeline in which each layer has a distinct role, while the interactions between layers remain explicit and controlled.

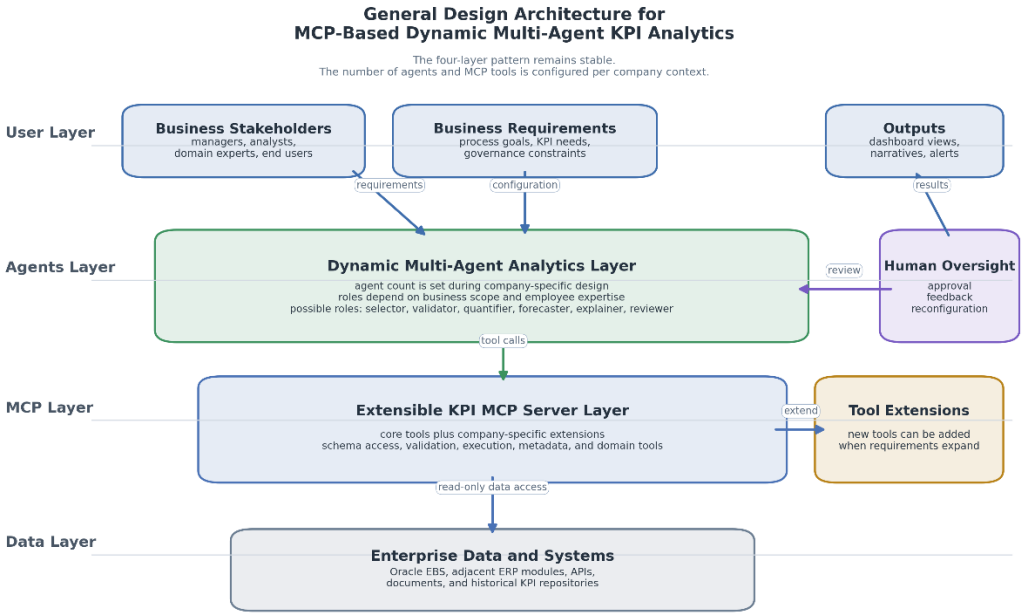


Figure 1. General Design Architecture for MCP-Based Dynamic Multi-Agent KPI analytics.

The user layer represents the business-facing and analyst-facing entry point of the system. At this level, users express business needs, validate intermediate results, and consume KPI analyses. This layer is intentionally not limited to simple input and output. Instead, it is designed around human-in-the-loop participation, meaning that users may intervene at critical points of the process, for example when confirming business areas, approving KPI candidates, or validating analytical logic before execution.

The dynamic multi-agent analytics layer is the central analytical component of the architecture. Its defining property is configurability. The number of agents and their responsibilities are not fixed in advance, but are determined during system design according to the business requirements, process complexity, and available employee expertise in the target company. In one organization, the analytical process may require only a small number of agents; in another, it may require additional agents for validation, review, explanation, or forecasting. In this sense, the architecture does not impose a universal agent structure, but instead provides a framework within which enterprise-specific analytical teams of agents can be instantiated.

The extendable EBS KPI MCP server layer acts as an intermediary between the analytical agents and Oracle EBS data. It exposes the tools required for controlled data access, validation, and transformation. This layer ensures that analytical agents do not interact directly with the ERP database or system interfaces, but only through explicitly defined tools that can enforce access policies, validation rules, and read-only constraints. The tool set is also extendable, meaning that different company deployments can enrich the server with additional domain-specific capabilities when required.

 doi:10.20944/preprints202604.2193.v1

The data layer contains Oracle EBS and related enterprise data sources. This includes transactional tables, metadata structures, and any additional supporting resources relevant to KPI management, such as historical KPI repositories, adjacent ERP modules, APIs, or documents. The role of this layer is not to expose data directly to the analytical workflow, but to serve as the underlying source that is accessed through the MCP server layer.

3.2. Dynamic Multi-Agent Analytics Layer

The dynamic multi-agent analytics layer is the key methodological contribution of the architecture. Existing multi-agent systems often define agent roles in advance, assuming a stable workflow that can be reused across contexts [8,9,23]. In contrast, the present design assumes that the structure of the agent layer must depend on the needs of the specific company. This assumption follows from the practical reality that KPI management requirements differ across organizations in terms of process scope, data maturity, control requirements, and analytical expertise.

For example, a company with relatively straightforward reporting needs may require agents only for KPI selection and KPI quantification. A more complex company environment may require additional agents for query validation, explanation generation, business-rule checking, or forecasting. Similarly, organizations with less internal analytical expertise may benefit from a stronger explanation-oriented layer, while organizations with advanced BI capabilities may prefer agents specialized in formal validation and SQL generation. Therefore, the architecture does not define the dynamic multi-agent analytics layer as a fixed set of roles, but as a configurable analytical environment capable of hosting multiple independent LLM agents, each assigned to a task derived from the company's requirements.

Methodologically, this means that the analytical process begins with system design rather than with immediate implementation. Before instantiating a concrete multi-agent workflow, the organization must determine which analytical functions are needed, which decisions require human participation, and which tasks can be reliably delegated to specialized agents. This design-first logic makes the architecture adaptable and prevents premature restriction of the system to one predetermined workflow.

3.3. Extendable EBS KPI MCP Server Layer

The MCP server layer is introduced to ensure that the analytical flexibility of the agent layer is matched by a controlled and reliable data-access mechanism. This is particularly important in Oracle EBS environments, where enterprise data structures are complex and operational constraints are significant [21]. In the proposed architecture, the EBS KPI MCP server layer is not merely a communication interface, but a semantic and operational control boundary.

At the proof-of-concept level, the server exposes a minimal core set of tools needed for KPI-oriented analysis. These include tools for retrieving schema metadata, validating SQL queries, and executing approved read-only KPI queries. This baseline tool set is sufficient to demonstrate how analytical agents can interpret business requirements, verify data structures, transform KPI definitions into executable analytical logic, and retrieve measurable results from Oracle EBS.

However, one of the key methodological assumptions of the architecture is that this tool set must remain extendable. Different company deployments may require additional tools for domain-specific metadata access, KPI template retrieval, business-calendar normalization, organizational-unit mapping, document lookup, or integration with adjacent enterprise systems. Therefore, the server layer is defined as extensible by design. It provides a stable architectural role, while allowing the actual tools exposed through it to evolve according to enterprise-specific analytical requirements.

This design is consistent with recent work on protocol-based tool integration, which emphasizes that structured external tools can improve the reliability, traceability, and governance of LLM-based systems [13–15,25]. In the present architecture, the MCP server layer fulfills exactly this role by grounding the analytical reasoning of the agent layer in validated enterprise structures.

 doi:10.20944/preprints202604.2193.v1

3.4. Human-in-the-Loop as a Methodological Principle

A further methodological principle of the proposed architecture is active user participation. The system is not designed as a fully autonomous analytical engine, but as a human-in-the-loop architecture in which users remain involved at key stages of the KPI lifecycle. This is particularly important because KPI management involves semantic and organizational judgments that cannot always be reduced to data availability or technical computability.

For example, even when a candidate KPI is measurable from Oracle EBS data, it may still be unsuitable if it does not properly reflect the intended business objective. Likewise, an automatically generated query may be syntactically valid but misaligned with managerial meaning. Human participation therefore serves both as a quality-control mechanism and as a governance safeguard. It allows business experts and analysts to validate whether the outputs of the system are not only technically feasible, but also organizationally meaningful.

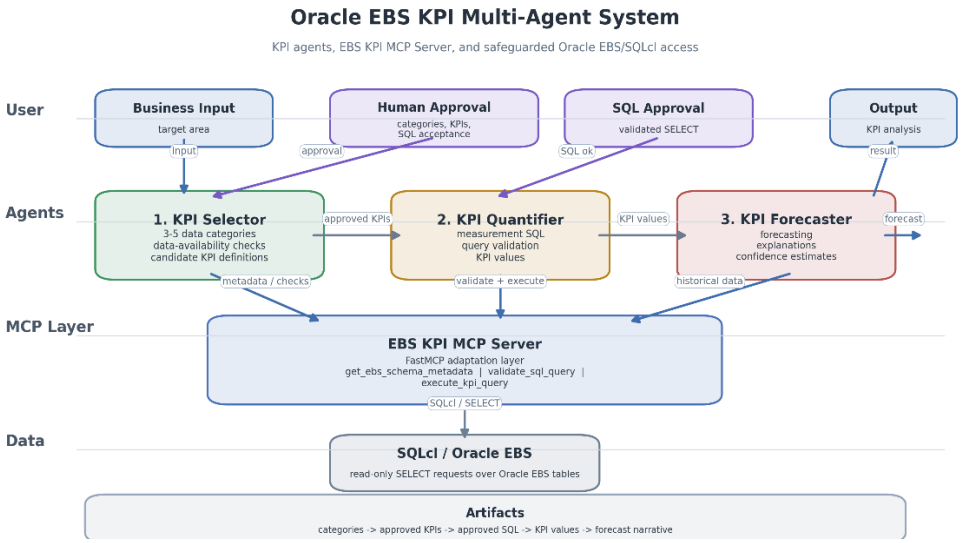
From a methodological perspective, the human-in-the-loop principle also strengthens the architecture's adaptability. It allows the system to evolve iteratively as users provide corrections, confirmations, and refinements. This creates a feedback loop in which the analytical workflow becomes progressively better aligned with the enterprise context.

3.5. Proof-of-Concept Instantiation

To demonstrate the practical applicability of the general architecture, the study implements a proof of concept system in which the dynamic multi-agent analytics layer is instantiated with three specialized agents: a KPI Selector, a KPI Quantifier, and a KPI Forecaster. This proof-of-concept implementation should be understood not as the only possible realization of the architecture, but as one concrete specialization derived from the broader design principles described above.

The KPI Selector is responsible for identifying relevant KPI categories and candidate indicators based on the business area specified by the user. The KPI Quantifier transforms confirmed KPI definitions into measurable analytical logic and interacts with the MCP server layer to validate and execute the required operations over Oracle EBS data. The KPI Forecaster operates on measured KPI values and historical information to generate projected tendencies and explanatory interpretations. Together, these three agents instantiate a minimal but functionally complete KPI lifecycle workflow and serve as evidence that the proposed architecture can be translated into an operational analytical system.

The proof-of-concept configuration also illustrates the central methodological claim of the paper: that a general architecture with a dynamic agent layer and an extendable MCP layer can support concrete and useful KPI management solutions in ERP environments. The specific three-agent implementation is therefore not the goal in itself, but a validation of the broader architectural model.



 doi:10.20944/preprints202604.2193.v1

Figure 2. MAS for dynamic business KPI selection, evaluation and quantification based on Oracle EBS architecture.

3.6. Methodological Outcome

The outcome of the proposed methodology is a layered, configurable architecture for dynamic KPI management in Oracle EBS environments. It combines business-oriented interaction, configurable multi-agent reasoning, protocol-based enterprise data access, and active human participation within a single integrated framework. In this way, the methodology addresses the limitations of static KPI systems and provides a foundation for adaptive KPI lifecycle automation in ERP ecosystems.

4. Implementation

To validate the proposed architecture, we implemented a prototype system as web-based application, which implementation follows the layered design introduced in the previous section, but here the focus is placed on the technical realization of the orchestration logic, the MCP-based tool layer, and the interaction with Oracle EBS. In contrast to a purely conceptual description, this section presents the system as an executable workflow.

The prototype is organized into three operational layers. The first is the interaction layer, where the user provides the target business area, reviews intermediate outputs, and approves critical decisions. The second is the LangGraph-based multi-agent layer, which manages the analytical state and routes execution between specialized agents. The third is the enterprise access layer, implemented through the EBS KPI MCP Server, which mediates between the agents and Oracle EBS. This separation provides a clear boundary between reasoning, orchestration, and data access, and prevents the LLM agents from interacting directly with the ERP system.

4.1. Web application

At the presentation layer, the prototype includes a user interface implemented with Streamlit. The choice of Streamlit is motivated by the need for a lightweight but functional analytical front end that can rapidly expose the workflow to business users and researchers without requiring a separate full-scale web application framework. Through this interface, the user specifies the business area, reviews the categories and KPI proposals generated by the agents, confirms SQL statements before execution, and receives the resulting KPI values and forecast-oriented explanations. In this way, Streamlit serves as the interaction point between the human participant and the multi-agent analytical backend, while also supporting the human-in-the-loop control model embedded in the architecture.

4.2. Multi-Agent System Orchestration

The core execution logic of the implemented system is realized through LangGraph, which coordinates the analytical process as a stateful graph rather than as a simple sequential chain. This choice is important because the KPI lifecycle contains multiple approval points, iterative refinement loops, and transitions between heterogeneous tasks such as category discovery, KPI definition, SQL generation, validation, execution, and forecasting. In the present implementation, LangGraph is responsible for maintaining the workflow state, routing control between specialized agents, and ensuring that rejected outputs are returned to the relevant earlier stage for revision. The graph state stores the main analytical artifacts generated during execution. These include the user-specified business area, candidate data categories, metadata validation results, proposed KPI definitions, generated SQL statements, SQL validation status, approved KPI values, and forecast outputs. Each graph node receives the current state, updates the fields associated with its responsibility, and passes the modified state to the next stage. In this way, LangGraph serves both as an orchestration engine and as a shared memory structure for the entire multi-agent process.

 doi:10.20944/preprints202604.2193.v1

The LLMs used in the prototype are implemented through the OpenAI API rather than through locally hosted models. In the current system, OpenAI ChatGPT-family models are used as the common reasoning backbone for the KPI Selector, KPI Quantifier, and KPI Forecaster agents.

Figure 3 represents the agent's logic as follows:

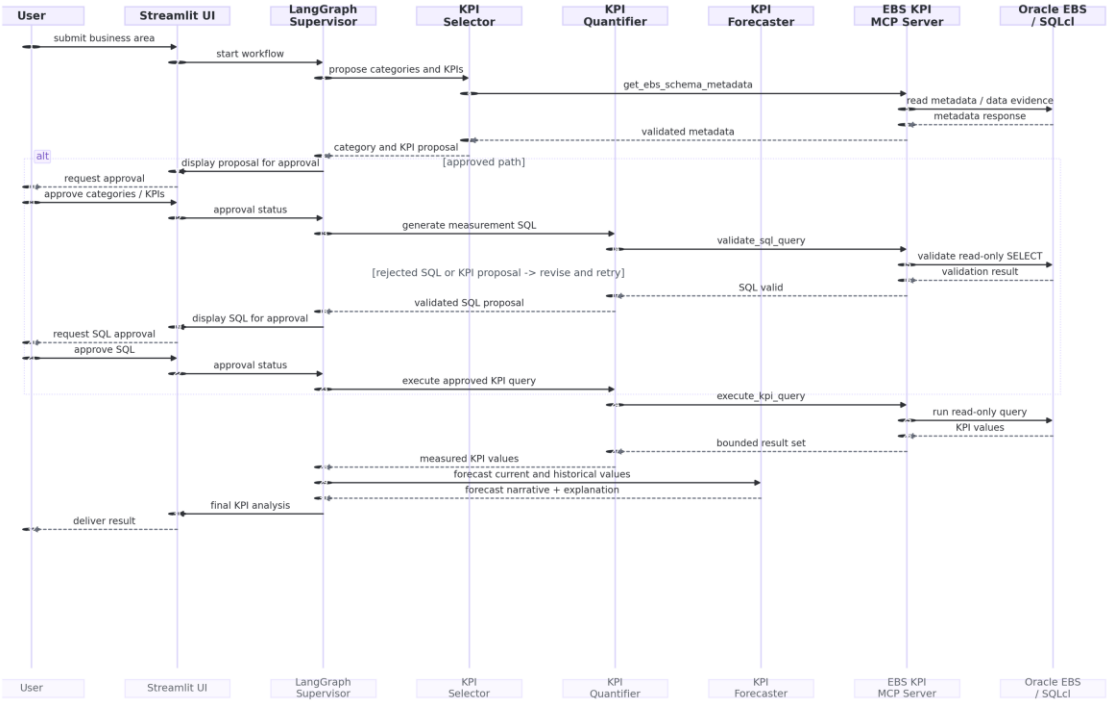


Figure 3. Multi-agent system sequence diagram.

The KPI Selector is the entry point of the analytical workflow. After receiving the business area from the user, it proposes between three and five candidate data categories relevant to the selected business scope. These categories are not accepted immediately. Instead, the agent invokes the MCP tool layer to check whether Oracle EBS contains actual data corresponding to the proposed categories. The result is a structured validation artifact containing the category, the executed check, and a flag indicating whether relevant data exists. Only after this step, and after user confirmation, does the agent produce a list of KPI candidates with names, descriptions, and associated categories.

The KPI Quantifier receives the approved KPI definitions and transforms them into executable measurement logic. In practice, this means generating SQL statements grounded in Oracle EBS schema metadata and the previously validated categories. The generated SQL is then passed to the MCP validation tool before it can be executed. This introduces a technical control point between SQL generation and database access. After validation, the SQL is shown for user confirmation, and only then is it executed against Oracle EBS in read-only mode. The output of this stage is a set of measured KPI values together with the validated SQL logic that produced them.

The KPI Forecaster consumes the measured KPI values and the available historical data. Its task is to derive forecast-oriented output and generate an explanatory interpretation of the observed tendencies. In implementation terms, this stage extends the workflow from descriptive analytics toward predictive support. The output is not limited to a numeric forecast. It also includes an explanation of the historical basis, the identified trend, and the resulting interpretation.

4.3. Prompting Strategy

An important implementation aspect of the proposed system is the prompting strategy used to coordinate the specialized agents. Although the analytical workflow is controlled by LangGraph, the quality of the generated outputs depends strongly on how each agent is instructed. In the current

 doi:10.20944/preprints202604.2193.v1

prototype, the agents use specialization is achieved primarily through role-specific prompting rather than through different model architectures. Each prompt combines four elements: role definition, workflow context, allowed actions, and expected output format. The prompting strategy is designed to reduce ambiguity and improve traceability. Figure 4 presents an example of the prompt used for the KPI Selector agent.

```
prompt = (
    "You are the KPI Selector agent. Based on confirmed Oracle EBS data categories with available data, "
    "propose KPI definitions. Do not invent tables or columns. Return only a JSON array. Each object must contain: "
    "kpi_name, description, data_categories.\n"
    "data_categories must be an array of category_name values from the confirmed categories.\n\n"
    f"Business area: {business_area}\n"
    f"Confirmed categories: {json.dumps(available_categories, ensure_ascii=False)}\n"
    f"User feedback or requested changes: {feedback or 'None'}"
)
```

Figure 4. Example of prompt for the KPI Selector.

4.4. EBS KPI MCP Server

The enterprise integration layer is implemented as EBS KPI MCP Server, built with FastMCP. This server acts as an intermediate service between the LangGraph agents and Oracle EBS. Its primary role is to encapsulate all ERP-facing operations behind explicit tools, thereby ensuring that the agents never issue uncontrolled database actions.

In the current implementation, the server exposes three core tools:

- get_ebs_schema_metadata
- validate_sql_query
- execute_kpi_query

The get_ebs_schema_metadata tool is used to retrieve structural information from Oracle EBS, including table and column metadata derived from the Oracle data dictionary. This tool reduces schema hallucination by grounding agent reasoning in real ERP structures. It is used primarily by the KPI Selector and KPI Quantifier when selecting data categories and constructing SQL-based KPI measurements.

The validate_sql_query tool checks whether a generated statement satisfies the system's execution constraints. In the current implementation, the main requirement is that the statement must be a valid read-only SELECT expression. This validation step is essential because it introduces a formal gate between SQL generation and execution.

The execute_kpi_query tool is responsible for running approved KPI queries and returning a bounded result set with column names and data values. Before execution, the implementation performs an additional safety check to ensure that the statement begins with SELECT. This tool is therefore the final execution endpoint in the measurement stage.

4.5. Oracle EBS Access and Execution Mode

The MCP server uses connection parameters supplied through environment configuration, via .env. At runtime, the implementation supports direct Oracle access through python-oracledb, while also allowing SQLcl fallback in scenarios where a SQLcl-based execution layer is required. This dual-access strategy makes the prototype more robust because it decouples agent behavior from the specific low-level mechanism used to reach Oracle EBS.

A key implementation principle is that all database access remains read-only. The system is designed for analytical KPI workflows, not for transactional modification of ERP records. Consequently, the allowed SQL subset is intentionally restricted. This aligns the implementation with enterprise governance requirements and reduces operational risk.

 doi:10.20944/preprints202604.2193.v1

4.6. Human-in-the-Loop Control Points

The implementation includes explicit human-in-the-loop checkpoints embedded into the LangGraph process. These checkpoints are not auxiliary interface features, but execution-level control nodes. User confirmation is required at least at the following stages: after the proposal of data categories, after the proposal of KPI definitions, and after the generation of SQL statements for KPI quantification.

When a confirmation is not granted, the graph does not terminate. Instead, it routes execution back to the relevant agent so that a revised proposal can be generated. This mechanism is particularly important in ERP analytics because semantic correctness cannot always be inferred from schema validity alone. A KPI may be technically measurable yet still be unsuitable from a business perspective. The implementation therefore combines automated reasoning with controlled expert validation.

4.7. Execution Example and Resulting Analytical Output

To illustrate the operation of the implemented system, this subsection presents an example execution for the business area Order management. The example covers the main stages of the workflow, from category discovery and KPI definition to SQL-based quantification and forecast-oriented output.

Figure 5 presents the KPI Selector stage. After the user specifies the business area, the system proposes relevant Oracle EBS data categories and evaluates their suitability through generated data-check SQL. In the example, the proposed categories include Sales Order Headers, Sales Order Lines, Shipping Delivery Details, Order Holds, and Returns and RMA Lines. The interface also shows the corresponding Oracle EBS tables and judge scores, after which the categories with available data are confirmed. This step grounds the workflow in data that are both semantically relevant and available in the ERP environment.

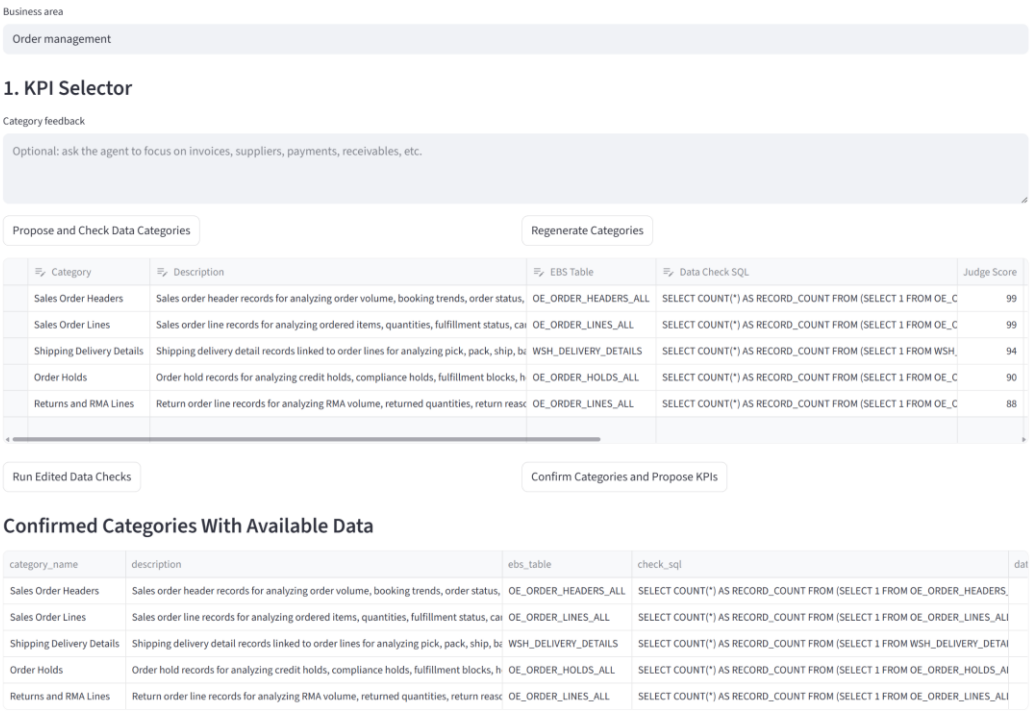


Figure 5. Data Categories proposal and evaluation.

Figure 6 shows the next stage, in which candidate KPI definitions are generated from the confirmed categories. The proposed indicators include operational and financial measures such as

 doi:10.20944/preprints202604.2193.v1

Line Fill Rate, Line Cancellation Rate, Open Order Backlog Value, On-Time Shipment Rate, and Order-to-Fulfillment Cycle Time. Each KPI is presented together with a short description and its associated data category. In this way, the system transforms the validated ERP categories into a candidate KPI portfolio suitable for further review and quantification.

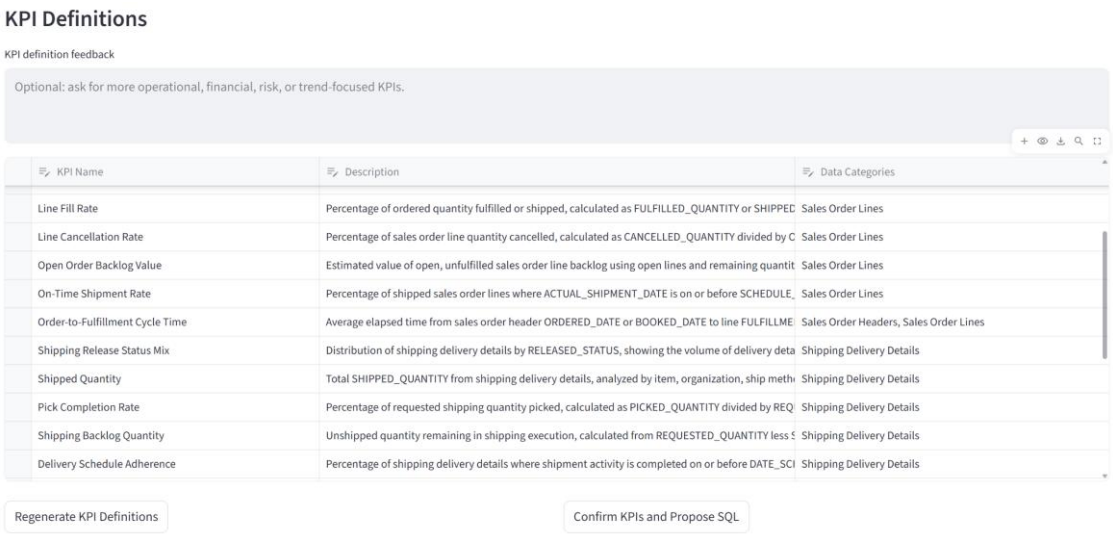


Figure 6. KPI Definitions.

Figure 7 illustrates the KPI Quantifier stage. At this point, the system generates SQL statements for the approved KPIs and presents them for user approval before execution. This stage is technically important because it introduces an explicit control point between KPI definition and Oracle EBS access. Rather than allowing automatic execution of all generated SQL, the system requires the user to approve the selected statements, after which they are executed through the MCP layer in read-only mode.

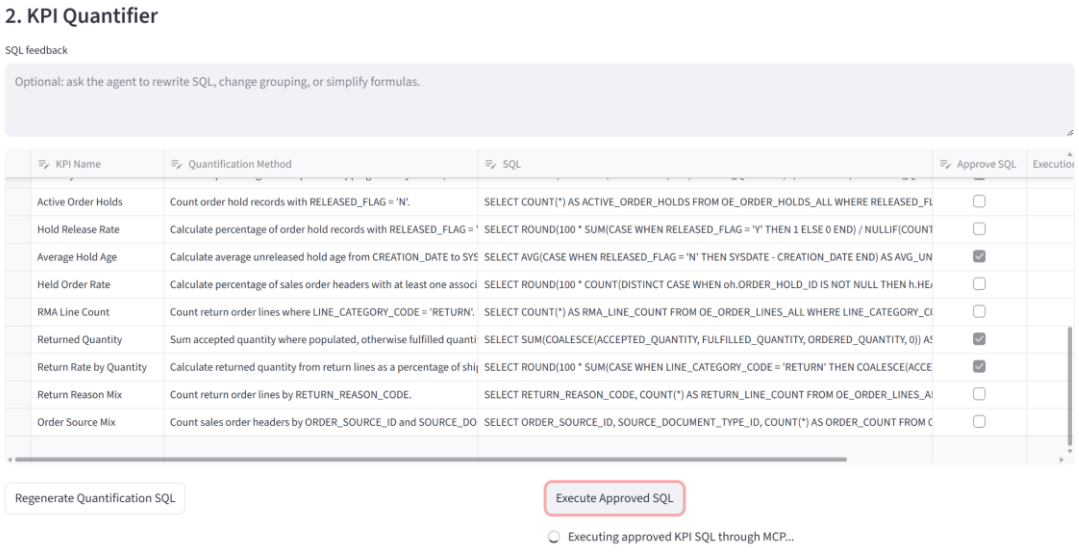


Figure 7. KPI Quantifier.

Figure 8 presents the quantification results returned from Oracle EBS after execution of the approved queries. In the example, the system produces measured values such as Booked Order Count = 36, Cancelled Order Rate = 4.65, and Order Line Revenue = 421146.5. These results confirm that the previously proposed KPIs are not only conceptually meaningful, but also computable from the available Oracle EBS data.

 doi:10.20944/preprints202604.2193.v1

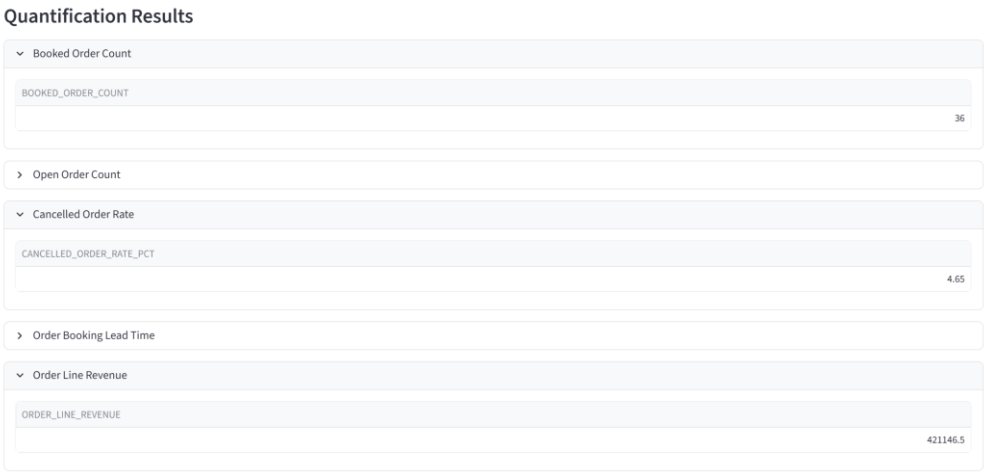


Figure 8. Quantification Results.

Finally, Figure 9 shows the KPI Forecaster stage. Based on the measured KPI values, the system generates forecast values and short explanatory outputs. The lower part of the interface presents dashboard-style KPI indicators that combine the current measured value with the forecasted one. For example, the prototype reports Booked Order Count with current value 36 and forecast 38, Cancelled Order Rate with current value 4.65 and forecast 4.8, and Order Line Revenue with current value 421146.5 and forecast 433781. This stage demonstrates that the workflow extends beyond descriptive KPI measurement and supports forward-looking analytical interpretation.

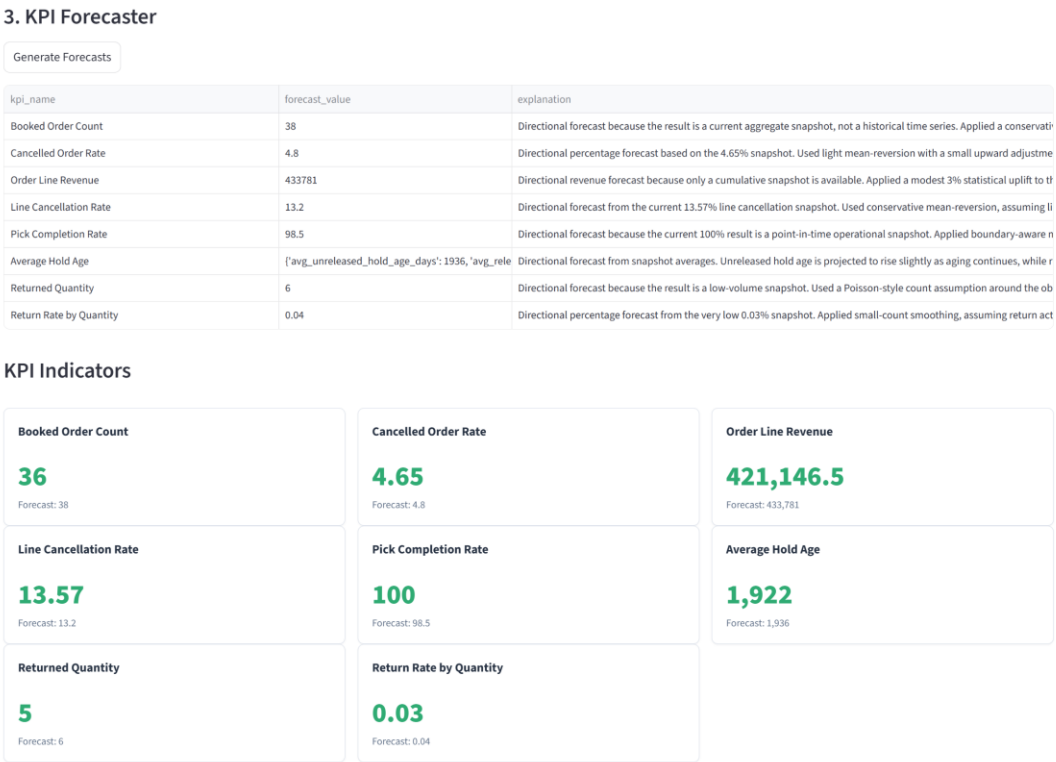


Figure 9. KPI Forecaster.

Taken together, Figures 5–9 illustrate the full execution logic of the implemented prototype. Starting from a user-defined business area, the system verifies relevant data categories, proposes KPI definitions, generates and validates measurement SQL, retrieves KPI values from Oracle EBS, and produces forecast-oriented analytical output. The example confirms the practical applicability of the proposed architecture as an executable enterprise KPI analytics workflow.

 doi:10.20944/preprints202604.2193.v1

5. Discussion

The proposed system contributes to enterprise analytics in three interconnected ways. First, it reframes KPI engineering as a lifecycle rather than as a one-time dashboard configuration exercise. This lifecycle is dynamic because every stage can revisit earlier decisions when data availability, validation status, or analyst feedback changes. Second, it demonstrates how protocol-mediated tool access can operationalize trust boundaries in LLM-based enterprise systems. Third, it shows that multi-agent decomposition is especially valuable when analytical correctness depends on intermediate approvals and semantically different reasoning tasks.

Several limitations should also be acknowledged. The current manuscript presents detailed architecture and an illustrative Oracle EBS scenario, but not a large-scale benchmark over multiple real installations. In practice, KPI performance thresholds, acceptable confidence ranges, and business-area taxonomies would need to be localized for each organization. Likewise, the forecasting stage is intentionally modular and can host methods of varying sophistication; future empirical work should compare alternative ensemble designs under real historical workloads.

Another limitation concerns metadata quality. The architecture assumes that schema metadata and table semantics are available in a sufficiently usable form. In real Oracle EBS environments, customizations, local naming conventions, and legacy extensions may complicate this assumption. This is precisely why the MCP server is valuable: it concentrates schema translation and policy control in one layer. Nevertheless, enterprise deployment would benefit from an additional semantic registry that stores curated business meanings for critical fields and joins.

From a governance perspective, the paper argues for a strong human-in-the-loop model. This choice may appear conservative, but it is appropriate for enterprise analytics where a mathematically plausible indicator can still be organizationally misleading. A mature deployment can gradually automate more steps, yet the architecture should preserve the possibility of human intervention at high-risk transitions.

6. Conclusions and Future Work

This paper proposed a general multi-agent architecture for dynamic business KPI management over Oracle E-Business Suite (EBS). The study addressed the limitations of static KPI engineering by introducing a configurable analytical framework in which specialized LLM-based agents operate over enterprise data through a controlled Model Context Protocol (MCP) layer. The proposed design separates the system into a user layer, a dynamic multi-agent analytics layer, an extendable EBS KPI MCP server layer, and a data layer, thereby establishing a clear boundary between semantic reasoning, tool invocation, and enterprise data access.

A key contribution of the paper is the introduction of the EBS KPI MCP server as a semantic and operational control layer between the agents and Oracle EBS. Instead of allowing direct and potentially unsafe interaction with ERP data, the system exposes a bounded set of tools for schema inspection, SQL validation, and KPI query execution. This architectural decision improves traceability, reduces the risk of hallucinated or invalid analytical outputs, and aligns the analytical workflow with enterprise governance requirements. In parallel, the use of LangGraph enables explicit workflow state management, conditional branching, and iterative return loops, which are essential for human-in-the-loop validation in KPI lifecycle management.

The proof-of-concept implementation further demonstrated that the general architecture can be instantiated in a concrete Oracle EBS setting. Through the coordinated work of the KPI Selector, KPI Quantifier, and KPI Forecaster, the prototype showed how business-area requests can be transformed into validated KPI proposals, measurable SQL logic, quantified KPI values, and forecast-oriented interpretations. In this way, the paper contributes not only a conceptual model, but also a technically grounded implementation pattern for enterprise KPI analytics.

Overall, the results support the view that coordinated LLM agents, when combined with protocol-mediated enterprise access and explicit human oversight, can form a practical foundation

 doi:10.20944/preprints202604.2193.v1

for adaptive KPI management in ERP ecosystems. The proposed approach is therefore relevant both as a research contribution in intelligent business analytics and as a basis for future enterprise-oriented analytical assistants built on Oracle EBS and related ERP environments.

Author Contributions: Conceptualization, V.K.; methodology, V.K. and G.S.; software, G.S.; validation, V.K. and G.S.; formal analysis, V.K. and G.S.; investigation, G.S.; resources, G.S.; data curation, G.S.; writing—original draft preparation, G.S.; writing—review and editing, V.K. and G.S.; visualization, G.S.; supervision, V.K.; project administration, V.K. and G.S.; funding acquisition, V.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Project "Digitalisation of the Economy in a Big Data Environment" BG16RFPR002-1.014-0013, funded by the European Regional Development Fund (ERDF) through the "Programme Research, Innovation and Digitalisation for Smart Transformation (PRIDST)".

Acknowledgments: During the preparation of this manuscript, the authors used Codex app for the purposes of Figure 1, Figure 2 and Figure 3 generation. The authors have reviewed and edited the output and take full responsibility for the content of this publication.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; et al. Language Models are Few-Shot Learners. Adv. Neural Inf. Process. Syst. 2020. https://doi.org/10.48550/arXiv.2005.14165
2. Bommasani, R.; Hudson, D.A.; Adeli, E.; Altman, R.; Arora, S.; von Arx, S.; et al. On the Opportunities and Risks of Foundation Models. arXiv 2021, arXiv:2108.07258. https://doi.org/10.48550/arXiv.2108.07258
3. Vyhmeister, E.; Hennig, L.; Song, Y.Q.; Senkerik, R. Metrics, KPIs, and Taxonomy for Data Valuation and Monetisation -- A Systematic Literature Review. arXiv 2025, arXiv:2508.18331. https://doi.org/10.48550/arXiv.2508.18331
4. Kaplan, R.S.; Norton, D.P. The Balanced Scorecard—Measures That Drive Performance. *Harv. Bus. Rev.* 1992, 70, 71–79.
5. Madsen, D.Ø. The Balanced Scorecard: History, Implementation, and Impact. Encyclopedia 2025, 5(1), 39. https://doi.org/10.3390/admsci15010039
6. Dvořáková, A.; Kotková Striteská, M. Shortcomings of Current Performance Measurement and Management Systems: A Literature Review. Proc. European Conference on Knowledge Management, 2023, 318-326. https://doi.org/10.34190/eckm.24.1.1478
7. Andonov, V. Enhancing University Students' Activities with Interactive Immediate Feedback Using Customized LLMs, *2024 12th International Scientific Conference on Computer Science (COMSCI)*, Sozopol, Bulgaria, 2024, pp. 1-5, doi: 10.1109/COMSCI63166.2024.10778529
8. Smajić, A.; Karlović, R.; Dasko, M.; Lorencin, I. Large Language Models for Structured and Semi-Structured Data, Recommender Systems and Knowledge Base Engineering: A Survey of Recent Techniques and Architectures, *Electronics* **2025**, *14*(15), 3153. arXiv 2023, arXiv:2305.14228
9. Ji, Z.; Lee, N.; Frieske, R.; et al. Survey of Hallucination in Natural Language Generation. ACM Computing Surveys 55(12). 2023, 55, 1–38. https://doi.org/10.1145/3571730.
10. Marzovanova, M.; Mihova, V. Business logic unit testing optimization in. NET using AI, Innovative Information Technologies for Economy Digitalization (IITED), Sofia, Bulgaria, 2025, https://ideas.repec.org/a/nwe/iitfed/y2024i1p250-256.html
11. Velkova, I. Next-Gen Accounting and the Disruptive Power of AI in Financial Forecasting and Efficiency, *Innovative Information Technologies for Economy Digitalization (IITED)*, University of National and World Economy, Sofia, Bulgaria, 2024, pp. 205-213. https://www.unwe.bg/doi/iited/2024/IITED.2024.26.pdf
12. Wooldridge, M. An Introduction to MultiAgent Systems, 2nd ed.; Wiley: Chichester, UK, 2009.
13. Park, J.S.; O'Brien, J.C.; Cai, C.J.; Morris, M.R.; Liang, P.; Bernstein, M.S. Generative Agents: Interactive Simulacra of Human Behavior. Proc. ACM UIST 2023, 1–22. https://doi.org/10.1145/3586183.3606763

 doi:10.20944/preprints202604.2193.v1

14. Du, Y.; Li, S.; Torralba, A.; Tenenbaum, J.B.; Mordatch, I. Improving Factuality and Reasoning in Language Models through Multiagent Debate. arXiv 2023, arXiv:2305.14325.
15. Schick, T.; Dwivedi-Yu, J.; Dessi, R.; et al. Toolformer: Language Models Can Teach Themselves to Use Tools. Adv. Neural Inf. Process. Syst. 2023, 36.
16. Lewis, P.; Perez, E.; Piktus, A.; et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. Adv. Neural Inf. Process. Syst. 2020, 33, 9459–9474. https://doi.org/10.48550/arXiv.2005.11401
17. Patil, S.G.; Zhang, T.; Wang, X.; Gonzalez, J.E. Gorilla: Large Language Model Connected with Massive APIs. arXiv 2023, arXiv:2305.15334.
18. Badawy, M.; Abd El-Aziz, A.A.; Idress, A.M.; Hefny, H.; Hossam, S. A Survey on Exploring Key Performance Indicators. Future Computing and Informatics Journal, volume 1, issues 1–2, pages 47–52. https://doi.org/10.1016/j.fcij.2016.04.001
19. Karjust, K.; et al. Development of a Sustainability-Oriented KPI Selection Model for Manufacturing Processes. Sustainability 2025, 17, 6374. https://doi.org/10.3390/su17146374.
20. Li, X.; et al. BEAVER: An Enterprise Benchmark for Text-to-SQL. arXiv 2024, arXiv:2409.02038.
21. Milev, P.; Tabov, Y. Conceptual Approach For Presenting Text Data From Web-Based Information Systems In Structured Form, *Business Management*, no. 1, pp. 46 - 57, 2022.
22. Kandpal, N.; Deng, H.; Roberts, A.; et al. Large Language Models Struggle to Learn Long-Tail Knowledge. Proc. ICML 2023, 15696–15707. https://doi.org/10.48550/arXiv.2211.08411
23. Bodensohn, H.; et al. Unveiling Challenges for LLMs in Enterprise Data Engineering. arXiv 2025, arXiv:2504.10950.
24. Chen, A.; et al. Text-to-SQL for Enterprise Data Analytics. arXiv 2025, arXiv:2507.14372.
25. Gómez, J R.; et al. Chatting with your ERP: A Recipe. arXiv 2025, arXiv:2507.23429.
26. Yang, H.; et al. FinRobot: Generative Business Process AI Agents for Enterprise Resource Planning in Finance. arXiv 2025, arXiv:2506.01423.
27. Yang, Y.; et al. LLM-based Multi-Agent Systems: Techniques and Business Perspectives. arXiv 2024, arXiv:2411.14033.
28. Choi, C.; et al. Structuring the Unstructured: A Multi-Agent System for Extracting and Querying Financial KPIs and Guidance. arXiv 2025, arXiv:2505.19197.
29. Ehtesham, A.; et al. A survey of agent interoperability protocols: Model Context Protocol (MCP), Agent Communication Protocol (ACP), Agent-to-Agent Protocol (A2A), and Agent Network Protocol (ANP). arXiv 2025, arXiv:2505.02279.
30. Mastouri, M.; et al. From REST to MCP: An Empirical Study of API Wrapping and Automated Server Generation for LLM Agents. arXiv 2025, arXiv:2507.16044.
31. Narajala, V.; Habler, I. Enterprise-Grade Security for the Model Context Protocol (MCP): Frameworks and Mitigation Strategies. arXiv 2025, arXiv:2504.08623.