

**OPERATIONALIZING ARTIFICIAL INTELLIGENCE ACROSS ENTERPRISE  
APPLICATIONS USING SAP CLOUD APPLICATION PROGRAMMING MODEL,  
SAP AI CORE, AND SAP BTP SERVICES**

*Rajendra Asuri*  
*Senior BTP Consultant*

---

*Abstract*

*Intelligent enterprise transformation depends on the effective integration of artificial intelligence with scalable application architectures, governed data environments, and established business processes. This study examines the operationalization of artificial intelligence across enterprise applications through SAP Cloud Application Programming Model, SAP AI Core, and SAP Business Technology Platform services. The research addresses architectural fragmentation, deployment complexity, disconnected workflows, limited model governance, and inconsistent service integration that restrict the practical adoption of enterprise intelligence. A mixed-method research design combines architectural assessment, prototype development, quantitative performance measurement, and qualitative evaluation of implementation practices. Results indicate that the proposed framework strengthens service interoperability, deployment scalability, development consistency, model accessibility, security, and lifecycle control while reducing integration effort. The principal innovation involves a unified architecture connecting CAP-based business services with AI Core workloads through secure interfaces, governed data exchange, and centralized platform services. Strategic contributions include accelerated application development, improved operational oversight, and closer alignment between intelligent capabilities and organizational objectives. Academic contributions establish an integrated perspective linking cloud-native development, artificial intelligence operations, and enterprise service orchestration. The study concludes that coordinated implementation of SAP CAP, SAP AI Core, and BTP services provides a dependable foundation for converting isolated artificial intelligence initiatives into scalable, maintainable, and business-aligned enterprise capabilities for researchers, developers, architects, and technology leaders.*

*Key words: Artificial Intelligence, Enterprise Applications, SAP Cloud Application Programming Model, SAP CAP, SAP AI Core, SAP Business Technology Platform, SAP BTP Services, Intelligent Enterprise, Cloud-Native Application Development, Machine Learning Integration, Enterprise AI Operations, AI Model Deployment, AI Lifecycle Management, Service Orchestration, Application Programming Interfaces, Secure Data Exchange, Business Process Automation, Microservices Architecture, Enterprise System Integration, Scalable AI Applications, AI Governance*

## **I. INTRODUCTION**

Enterprise competitiveness increasingly depends on the ability to convert artificial intelligence models into dependable operational capabilities embedded within everyday business applications. Organizations expect intelligent systems to support forecasting, classification, recommendation, anomaly detection, and process automation without disrupting established transactional

workflows. Practical value, however, emerges only when models can access governed enterprise data, interact consistently with business services, and operate under measurable performance controls. Enterprise artificial intelligence must therefore be treated as an integrated application capability rather than an independent analytical experiment. This requirement has intensified the demand for architectures that connect application development, model execution, data management, security, and operational monitoring within a unified environment.

Many enterprise artificial intelligence initiatives remain confined to prototypes, departmental solutions, or disconnected data science environments. Such initiatives may demonstrate technical feasibility but frequently encounter difficulties when moved into production. Inconsistent interfaces, duplicated integration logic, manual deployment procedures, fragmented monitoring, and unclear ownership weaken reliability and increase maintenance effort. Model behavior can also deteriorate when operational data differs from the information used during development. These limitations reveal a significant distinction between creating an effective model and operating that model as part of a secure, scalable, and continuously governed enterprise application. Isolated implementation practices consequently restrict reuse, slow deployment, and prevent artificial intelligence investments from producing sustained organizational value.

SAP Cloud Application Programming Model provides a structured foundation for developing cloud-based enterprise applications through domain models, service definitions, reusable programming conventions, and standardized application interfaces. Its emphasis on business semantics allows developers to represent organizational entities and processes while reducing repetitive technical implementation. SAP AI Core complements this application foundation by supporting the execution, deployment, and management of artificial intelligence workloads. SAP Business Technology Platform services contribute identity management, connectivity, destination configuration, data access, application hosting, and operational integration. Coordinated use of these capabilities offers a promising path for embedding model-driven intelligence within business services while maintaining separation between application logic, model execution, and platform responsibilities.

Current knowledge provides substantial guidance on machine learning operations, cloud application engineering, continuous delivery, and artificial intelligence lifecycle management. Limited attention has nevertheless been given to how these disciplines can be combined within an SAP-centered enterprise application architecture. Available implementation discussions often describe individual services, configuration procedures, or narrowly defined demonstrations without presenting an integrated framework connecting business requests, CAP services, AI Core workloads, and supporting BTP capabilities. This fragmentation leaves enterprise architects without a clear account of service responsibilities, interaction boundaries, governance checkpoints, and runtime behavior. The central research problem concerns how artificial intelligence can be operationalized across SAP enterprise applications without introducing excessive coupling, inconsistent controls, or unmanageable integration complexity.

Practical motivation arises from the need to move beyond experimental model consumption toward repeatable and accountable enterprise deployment. Application teams require stable service contracts, while data science teams need controlled environments for model execution, versioning, and performance supervision. Security teams expect authenticated communication, restricted resource access, auditable operations, and traceable data movement. Business stakeholders require intelligent outcomes that remain aligned with process objectives and service expectations. A technically successful integration must satisfy all these interests simultaneously.

This study argues that SAP CAP, SAP AI Core, and selected BTP services can form a coherent operational foundation when their responsibilities are defined through explicit architectural layers, governed interfaces, and lifecycle controls.

The primary aim of this study is to develop and evaluate an enterprise framework for integrating artificial intelligence capabilities into CAP-based applications through SAP AI Core and supporting BTP services. The objectives are to identify relevant business and technical requirements, define the responsibilities of each platform component, establish a secure interaction model, and examine the operational value of an integrated approach. Additional objectives include assessing service interoperability, deployment consistency, scalability, lifecycle visibility, and integration effort. The study also seeks to determine how application logic and artificial intelligence workloads can evolve independently while continuing to participate in a coordinated business process. These objectives position operational reliability and architectural maintainability alongside model functionality.

Three research questions guide the investigation. The first asks how SAP CAP applications can consume artificial intelligence workloads through SAP AI Core while preserving clear separation between business services and model execution. The second asks which BTP services and governance controls are required to support secure data exchange, identity management, deployment supervision, and operational monitoring. The third asks whether an integrated framework can improve interoperability, deployment consistency, lifecycle control, and scalability when compared with fragmented integration practices. Collectively, these questions connect architectural design with observable operational outcomes. They also direct attention toward the organizational and engineering conditions required to sustain artificial intelligence after its initial deployment.

The proposed contribution is a structured enterprise architecture that links CAP-based business services, AI Core workloads, and BTP platform capabilities through governed application programming interfaces and clearly assigned operational responsibilities. The framework is expected to reduce integration ambiguity, improve service reuse, and support consistent movement from model preparation to application consumption. Academic significance arises from connecting artificial intelligence operations with enterprise application architecture and platform-based service orchestration. Industry significance lies in providing architects, developers, data scientists, and technology leaders with a practical basis for evaluating implementation choices. Expected value extends beyond technical connectivity by addressing accountability, maintainability, security, and alignment between intelligent functions and business processes.

Study boundaries concentrate on enterprise applications developed with SAP Cloud Application Programming Model, artificial intelligence workloads executed through SAP AI Core, and BTP services directly supporting integration, security, connectivity, deployment, and monitoring. The investigation does not compare model-training algorithms, assess every available BTP service, or claim universal applicability across all cloud environments. Subsequent sections establish the enterprise AI and SAP technology foundations, examine operational requirements and integration challenges, describe the research design, present the proposed framework, explain its implementation workflow, and evaluate the resulting performance and strategic implications. The final section consolidates the findings, acknowledges limitations, and identifies directions for further research on governed enterprise artificial intelligence.

## **II. ENTERPRISE AI FOUNDATIONS AND SAP TECHNOLOGY CONTEXT**

### **2.1 Evolution of Enterprise Artificial Intelligence**

Enterprise artificial intelligence has progressed from isolated analytical models toward operational capabilities embedded within business applications and decision processes. Earlier implementations commonly focused on departmental forecasting, classification, optimization, and predictive maintenance use cases supported by separately managed data environments. Growing expectations for real-time decisions have shifted attention toward reusable intelligence that can function consistently across applications. This development requires closer coordination among business services, data pipelines, model execution environments, and operational controls. Artificial intelligence has consequently become an architectural concern involving software engineering, data management, security, and business process design.

Operational adoption introduces requirements that differ substantially from experimental model development. A model demonstrating acceptable accuracy in a controlled environment may not remain dependable when exposed to changing data, variable workloads, application failures, or evolving business rules. Enterprise implementations must address model versioning, deployment reproducibility, service availability, data quality, access control, and performance monitoring. Continuous coordination between application developers, data specialists, platform engineers, and business owners becomes essential. These conditions have encouraged lifecycle-oriented practices that manage artificial intelligence assets from preparation and validation through deployment, monitoring, revision, and eventual retirement.

Platform-based development provides a practical response to the complexity of enterprise artificial intelligence operations. Shared platform services can standardize authentication, connectivity, workload execution, data access, logging, and application deployment across multiple use cases. Standardization reduces the need for each project to construct independent operational mechanisms and enables technical controls to be applied more consistently. Platform adoption does not eliminate architectural responsibility, because services must still be selected and combined according to business requirements. Effective operationalization therefore depends on a clear division between business application behavior, artificial intelligence processing, platform integration, and organizational governance.

### **2.2 SAP Cloud Application Programming Model Fundamentals**

SAP Cloud Application Programming Model offers a structured method for building service-oriented enterprise applications using domain-driven design principles and declarative development conventions. Core Data Services allows developers to represent business entities, relationships, constraints, and service definitions in a consistent semantic model. These definitions can support database structures, application services, and exposed interfaces while preserving a common interpretation of business information. Generic runtime capabilities handle recurring operations, allowing implementation effort to concentrate on specialized business rules. This model-based approach improves consistency between stored data, application logic, and external service contracts.

CAP applications can be developed using Node.js or Java service runtimes and deployed within supported SAP BTP runtime environments. Service definitions operate as controlled interfaces through which users, applications, and external systems interact with enterprise data and business functions. Event handlers extend standard processing when validation, orchestration, or specialized logic is required. OData and REST-based communication support integration with user

interfaces and remote services. Authentication and authorization rules can be associated with application services, creating a governed boundary between business requests and protected enterprise resources.

CAP provides an appropriate application layer for artificial intelligence integration because it separates domain semantics from external workload execution. Business services can validate incoming requests, collect relevant enterprise data, invoke an artificial intelligence endpoint, and interpret the returned result before initiating a process action. This arrangement prevents model-specific processing from becoming inseparably embedded within the domain model. Stable service interfaces can remain available even when an underlying model is revised or redeployed. CAP consequently acts as the business coordination layer that translates operational requirements into controlled requests and converts model outputs into contextually meaningful enterprise responses.

### **2.3 SAP AI Core and SAP BTP Service Capabilities**

SAP AI Core provides a managed environment for executing and operating artificial intelligence workloads within an enterprise platform landscape. Workloads can be organized through scenarios and configurations that define executable artifacts, input parameters, and required resources. Executions support model training or related processing activities, while deployments make validated models available for inference through service endpoints. Resource groups provide logical separation among teams, projects, or application contexts. These capabilities create a controlled boundary between model development and application consumption while supporting repeatability, workload isolation, and operational supervision.

SAP BTP services supply the supporting capabilities required to connect CAP applications with AI Core and existing enterprise systems. Identity and authorization services establish authenticated access, while destination and connectivity capabilities enable managed communication with remote endpoints. SAP HANA Cloud can provide governed persistence and analytical data access where the implementation requires an integrated data foundation. Cloud Foundry or Kyma runtime environments can host application components according to architectural and operational requirements. Logging, monitoring, credential management, and service bindings strengthen runtime administration and reduce the exposure of sensitive connection information within application code.

Combined use of SAP CAP, SAP AI Core, and supporting BTP services establishes a foundation for converting artificial intelligence models into reusable enterprise capabilities. CAP governs business semantics and application interactions, AI Core manages workload execution and model deployment, and BTP services provide connectivity, identity, data access, hosting, and operational support. Their combined value depends on explicit interfaces and responsibilities rather than simple technical co-location. The remaining research gap concerns how these capabilities should be assembled into a measurable, secure, and maintainable operational framework. Addressing that gap requires systematic evaluation of enterprise requirements, integration barriers, governance expectations, and lifecycle controls.

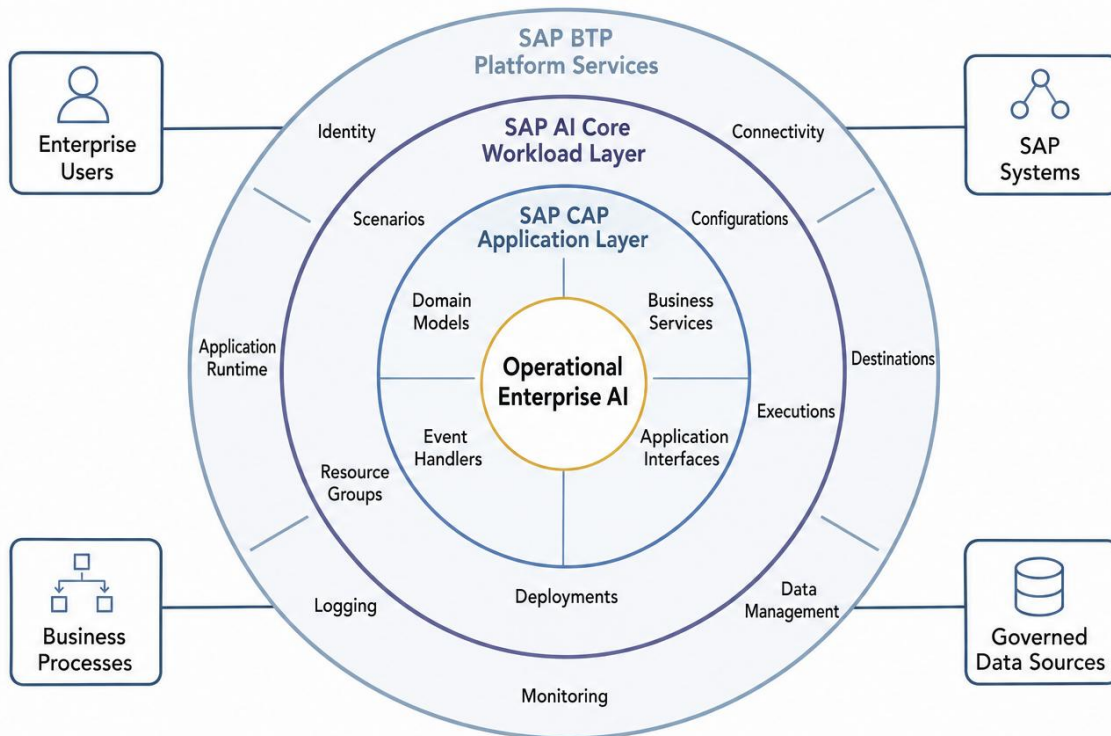


Figure 1: Enterprise AI Technology Foundation Across SAP CAP, SAP AI Core, and SAP BTP Services

### III. ENTERPRISE REQUIREMENTS AND INTEGRATION CHALLENGES

#### 3.1 Operational and Business Requirements

Enterprise artificial intelligence initiatives must begin with a clearly defined relationship between intelligent capability and business process value. Suitable use cases should involve measurable decisions, repeatable data inputs, identifiable process owners, and outcomes that can be evaluated against operational objectives. Classification, forecasting, recommendation, anomaly detection, and decision-support functions become valuable when they reduce processing effort or improve decision consistency. Model capability alone does not establish business relevance. Enterprise adoption requires explicit performance expectations, user responsibilities, exception procedures, and success criteria that connect technical outputs with practical organizational benefits.

Operational requirements extend beyond the production of an accurate model response. Enterprise applications need predictable availability, acceptable response time, workload scalability, reliable transaction handling, and controlled recovery when an artificial intelligence service becomes unavailable. Business processes may also require confidence thresholds, manual review, or alternative processing paths when returned results remain uncertain. CAP-based services must therefore evaluate model outputs within the relevant domain context before initiating a business action. This arrangement preserves process integrity and prevents an artificial intelligence response from becoming an uncontrolled decision within a transactional environment. Organizational alignment represents another essential requirement because enterprise artificial

intelligence crosses conventional technical boundaries. Application teams manage business services, data specialists prepare and validate models, platform teams maintain runtime environments, and security teams supervise access and compliance controls. Business owners remain accountable for the consequences of intelligent recommendations within their processes. Unclear ownership can delay incident resolution and weaken model supervision after deployment. A sustainable operating model should assign responsibility for data quality, model approval, service availability, application behavior, performance monitoring, and business outcome evaluation throughout the operational lifecycle.

### **3.2 Technical Integration Barriers**

Fragmented application and model environments create substantial integration difficulties. Enterprise data may reside across transactional systems, analytical repositories, cloud services, and external platforms using different structures and access mechanisms. Artificial intelligence workloads require consistent input representations, but business applications frequently express information through domain-specific entities and service contracts. Direct point-to-point connections can introduce duplicated mappings, embedded credentials, and model-dependent code within application services. These conditions increase maintenance effort and make model replacement difficult. A controlled integration layer is required to translate enterprise data into validated workload requests without weakening application boundaries.

Semantic consistency becomes particularly important when CAP services and AI Core workloads interpret the same business information differently. A field that appears technically compatible may carry a different meaning, unit, time range, or level of aggregation within the model environment. Such differences can produce valid service responses that remain operationally misleading. Domain models, request schemas, validation rules, and metadata should establish a shared interpretation before inference occurs. Version compatibility must also be maintained when business entities, model inputs, or service interfaces change. Explicit schema governance reduces the risk of silent integration failure and protects the meaning of model-generated results.

Runtime communication introduces additional concerns involving latency, workload capacity, network interruptions, and dependency availability. CAP applications may invoke deployed AI Core endpoints synchronously when an immediate result is required or coordinate asynchronous processing when workloads require longer execution. Each interaction pattern creates different expectations for timeout management, retry behavior, request tracking, and user feedback. Service bindings and destinations can reduce configuration inconsistency, while correlation identifiers support traceability across application and workload components. Resilient integration requires controlled failure handling so that temporary model or network problems do not compromise the surrounding business transaction.

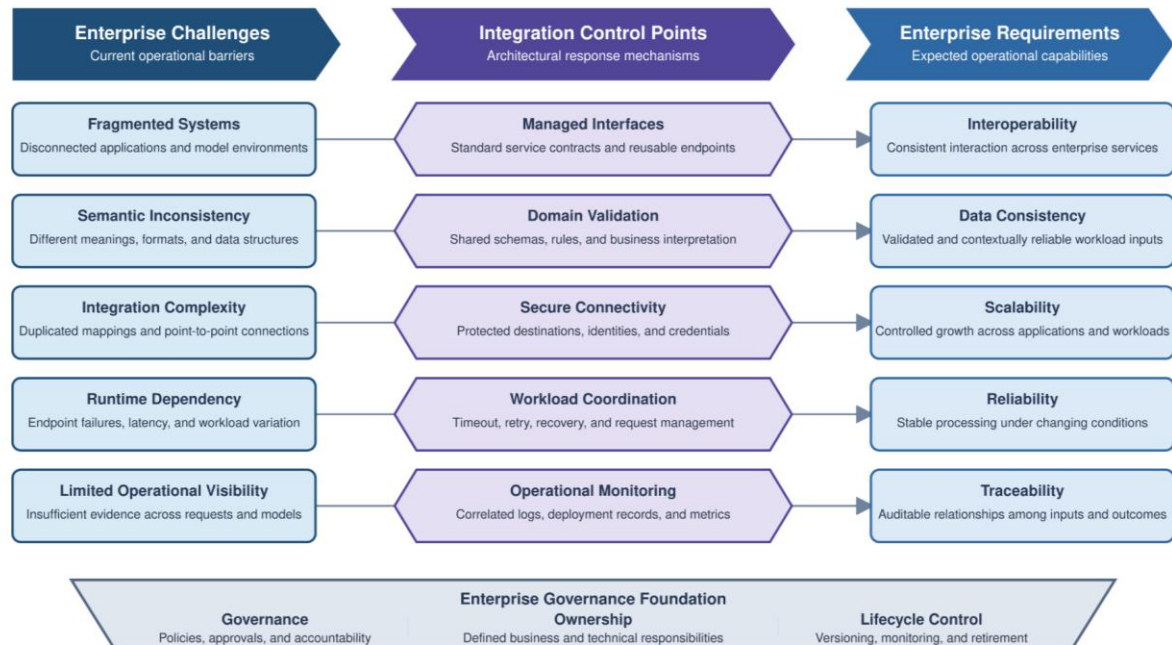


Figure 2: Enterprise AI Integration Challenge and Requirement Map

### 3.3 Governance, Security, and Lifecycle Requirements

Security requirements apply to every interaction among users, CAP services, AI Core workloads, platform services, and enterprise data sources. Authentication must establish the identity of each requesting application or user, while authorization must restrict access according to assigned roles and resource responsibilities. Credentials should be managed through service bindings or protected configuration rather than stored within application code. Communication channels require appropriate protection, and sensitive fields should be limited to the minimum information necessary for model execution. These controls reduce exposure while preserving accountable access to artificial intelligence capabilities.

Lifecycle governance must address model registration, validation, approval, deployment, revision, and retirement as connected operational activities. Model versions should remain associated with their training context, configuration, intended use, validation status, and active deployment. Changes to model artifacts or application interfaces require controlled testing before release. Monitoring should examine service availability, processing time, failure frequency, input variation, and output quality where suitable reference information exists. Traceable lifecycle records allow technical teams to identify which model version produced a result and determine whether operational behavior remains consistent with approved expectations.

Enterprise requirements derived from this analysis guide the proposed operationalization framework. SAP CAP must provide the governed business-service boundary, SAP AI Core must manage artificial intelligence workload execution and deployment, and BTP services must support identity, connectivity, data access, runtime administration, and monitoring. Effective integration depends on coordinated controls rather than independent product capabilities. The framework must consequently balance interoperability, security, scalability, traceability, maintainability, and business alignment. Table 1 consolidates these requirements and maps them to their

corresponding SAP technology capabilities and expected operational contributions.

Table 1: Enterprise AI Requirements and Corresponding SAP Technology Capabilities

| Enterprise requirement       | SAP technology capability                   | Expected benefit                              |
|------------------------------|---------------------------------------------|-----------------------------------------------|
| Business process integration | SAP CAP business services                   | Connects AI results with business operations  |
| Model execution              | SAP AI Core deployments                     | Provides controlled AI workload execution     |
| Data consistency             | CAP domain models and SAP HANA Cloud        | Improves data quality and consistency         |
| Secure access                | BTP identity and authorization services     | Protects applications, data, and AI resources |
| System connectivity          | BTP destinations and connectivity services  | Supports reliable system integration          |
| Scalability                  | BTP runtime and AI Core resource management | Handles changing application workloads        |
| Monitoring                   | BTP logging and monitoring services         | Improves operational visibility               |
| Lifecycle control            | AI Core configurations and deployments      | Supports model versioning and governance      |

#### **IV. RESEARCH DESIGN AND EVALUATION METHOD**

##### **4.1 Mixed-Method Research Design**

The study adopts a mixed-method research design to examine both measurable system behavior and the architectural qualities of the proposed enterprise artificial intelligence framework. Quantitative assessment evaluates response time, throughput, processing success, deployment consistency, and integration effort under controlled workload conditions. Qualitative assessment examines interoperability, maintainability, security, governance, and alignment with enterprise development practices. Combining these perspectives prevents the evaluation from depending exclusively on technical performance or subjective architectural judgment. The research design therefore supports a balanced interpretation of operational effectiveness across application, workload, and platform layers.

A design-oriented research strategy guides the development and assessment of the framework. The investigation begins by identifying enterprise integration problems, translating them into technical and operational requirements, and constructing an architecture that addresses those requirements through coordinated SAP capabilities. Prototype implementation provides practical evidence concerning the feasibility of the proposed design. Evaluation then compares the integrated framework with a fragmented baseline approach using identical business scenarios and workload conditions. This sequence connects the research problem, framework construction, prototype implementation, empirical measurement, and architectural interpretation within a consistent analytical process.

Research validity is supported through controlled configuration, repeated execution, consistent input data, and predefined evaluation criteria. Each test cycle uses the same business request categories, validation rules, model endpoint, and expected response structure. Environmental settings and service configurations remain stable during comparative measurements to reduce unrelated variation. Qualitative observations are recorded against a structured assessment checklist rather than informal impressions. This procedure improves transparency while acknowledging that prototype findings may vary when implemented across different enterprise landscapes, data volumes, network conditions, and organizational governance structures.

##### **4.2 Prototype and Scenario Development**

The prototype represents an enterprise service-request classification and prioritization process implemented through SAP CAP, SAP AI Core, and supporting BTP services. Incoming requests contain structured business attributes and descriptive text requiring classification into an operational category and priority level. The CAP application receives the request, validates required information, applies business rules, and prepares a standardized workload payload. SAP AI Core hosts the deployed model used to generate the classification result. The returned output is interpreted by the CAP service before being stored or presented as a recommendation within the business process.

Scenario selection reflects several common characteristics of enterprise artificial intelligence applications. The process requires transactional data, unstructured information, model inference, service integration, access control, and traceable decision support. Classification results must be returned within an acceptable processing period while preserving the relationship between the original request, model deployment, and resulting recommendation. The scenario also allows uncertain outputs and service failures to be handled through defined application logic. These characteristics make the prototype suitable for evaluating technical integration without limiting

the framework to a single industry or narrowly specialized business function.

Prototype development follows a controlled sequence beginning with the CAP domain model and service contract. Core Data Services defines request entities, classification outcomes, processing status, and relevant audit information. Event handlers perform input validation, workload invocation, response interpretation, and exception handling. SAP AI Core configurations identify the executable workload and active deployment, while resource groups separate prototype assets from unrelated workloads. BTP identity, destination, connectivity, and logging capabilities support protected communication and operational visibility. This implementation structure maintains clear boundaries among business services, model execution, and platform responsibilities.

Topic-based distribution supports business information that must reach several independent applications. A completed order may produce information required by notification, reporting, customer administration, and audit services without requiring separate direct calls from the originating application. Each subscriber receives the published message according to its subscription and availability conditions. Durable subscriptions retain relevant topic messages for registered subscribers that are temporarily inactive. Queue and topic models therefore serve complementary purposes, with queues assigning work and topics distributing shared enterprise information.

#### **4.3 Evaluation Metrics and Analytical Procedure**

Quantitative evaluation uses five principal measures: response time, throughput, processing success rate, deployment consistency, and integration effort. Response time records the duration between validated application submission and receipt of an interpretable model result. Throughput measures the number of completed requests within a defined observation period. Processing success rate represents the proportion of requests completed without communication, validation, or workload failure. Deployment consistency examines whether repeated releases produce equivalent service availability and configuration behavior. Integration effort is assessed through the number of manual steps and separately maintained connection elements required by each approach.

Qualitative evaluation examines architectural characteristics that cannot be represented adequately through performance values alone. Interoperability is assessed through the clarity and reuse of service interfaces. Maintainability considers separation of concerns, configuration centralization, and the effort required to replace or revise a deployed model. Security assessment examines identity propagation, credential protection, access restrictions, and protected service communication. Governance evaluation considers deployment traceability, resource separation, logging, and responsibility assignment. Scalability is interpreted through the framework's ability to increase application or model capacity without restructuring the entire integration design.

The analytical procedure compares the integrated framework with a fragmented baseline in which application-to-model communication depends on separately configured endpoints, locally maintained credentials, and limited lifecycle coordination. Repeated prototype executions provide the quantitative observations, while an architecture assessment matrix records qualitative differences. Results are normalized where measures use different units, allowing patterns to be compared without implying identical measurement scales. Findings are interpreted according to operational significance rather than numerical improvement alone. This procedure establishes an evidence-based foundation for evaluating whether the proposed framework provides a practical

and maintainable approach to enterprise artificial intelligence operationalization.

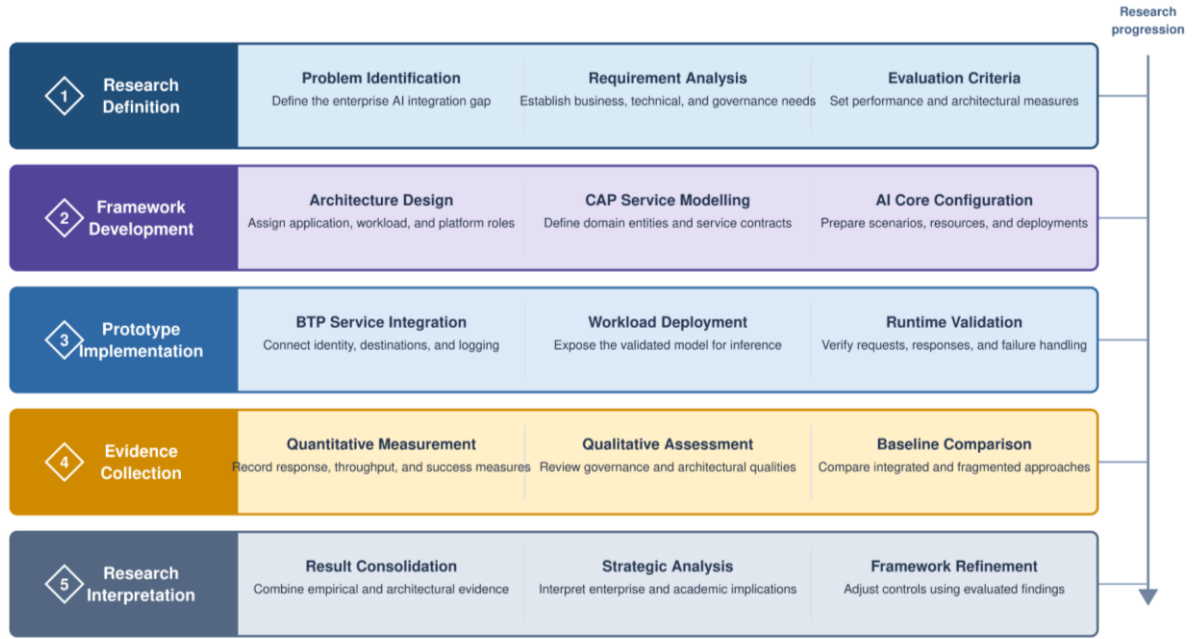


Figure 3: Mixed-Method Research and Prototype Evaluation Process

## V. PROPOSED ENTERPRISE AI OPERATIONALIZATION FRAMEWORK

### 5.1 Framework Architecture and Design Principles

The proposed framework establishes a coordinated architecture for integrating artificial intelligence workloads with enterprise applications developed through SAP Cloud Application Programming Model. Its principal design objective is to separate business logic, model execution, platform services, and operational governance while maintaining controlled interaction among them. SAP CAP governs domain behavior and service requests, SAP AI Core manages artificial intelligence workloads, and supporting BTP services provide identity, connectivity, data access, runtime administration, and monitoring. This separation reduces technical coupling and allows application components to evolve without requiring simultaneous modification across the complete architecture.

Modularity, interoperability, security, traceability, and lifecycle control form the central design principles of the framework. Modularity divides responsibilities into independently manageable architectural layers. Interoperability relies on defined service contracts and standardized request structures. Security applies authenticated communication and controlled resource access across every interaction. Traceability connects application requests with workload executions, deployment versions, and returned outcomes. Lifecycle control ensures that model registration, validation, deployment, monitoring, revision, and retirement remain governed activities rather than isolated technical actions.

The architecture contains five functional layers: enterprise interaction, CAP business services, AI integration coordination, AI Core workload operation, and BTP platform support. Enterprise users and connected systems initiate requests through application interfaces. CAP services validate the

business context and apply domain rules before forwarding eligible workloads. The integration coordination layer prepares payloads, resolves destinations, and manages runtime communication. AI Core executes the deployed model and returns an inference result. BTP platform services support identity, connectivity, persistence, logging, and operational supervision across the architecture.

### **5.2 Service Interaction and Data Exchange**

Service interaction begins when a user or connected enterprise system submits a request to a CAP application interface. The application service verifies authentication, evaluates authorization, and validates required business information. Domain rules determine whether the request qualifies for artificial intelligence processing or should follow a conventional application path. Approved information is transformed into a stable workload payload that contains only the attributes required for inference. This process protects business semantics and prevents internal application structures from becoming directly dependent on the model endpoint.

Managed destination and connectivity capabilities establish communication between the CAP service and the deployed AI Core workload. Endpoint information and credentials remain outside the application source code, allowing connection settings to be changed without modifying business logic. A correlation identifier accompanies the request so that application events, workload activity, and returned results can be associated during monitoring or investigation. Timeout limits, retry conditions, and exception responses are defined according to process sensitivity. These controls create predictable behavior when network delays, workload unavailability, or invalid responses occur.

The AI Core deployment processes the validated request and returns a structured inference result containing the predicted outcome and relevant response information. CAP event handlers interpret this output within the business context rather than accepting it as an unrestricted decision. Confidence thresholds, validation rules, and process conditions determine whether the result can be stored, presented as a recommendation, or redirected for manual review. Application responses contain operationally meaningful information while excluding unnecessary model details. This controlled exchange preserves separation between artificial intelligence processing and enterprise decision responsibility.

### **5.3 Security, Governance, and Lifecycle Controls**

Security controls operate across application, service, workload, and data boundaries. Identity services authenticate users and technical clients, while role-based authorization limits access to CAP operations and AI Core resources. Service bindings and managed destinations protect connection details and reduce the need for locally stored credentials. Data minimization restricts workload payloads to information required for the approved use case. Protected communication maintains confidentiality during transfer, while resource groups provide logical separation among projects, teams, and application environments. These measures establish layered protection without embedding security responsibility within a single component.

Governance controls define how artificial intelligence assets are introduced, approved, operated, and revised. Each deployed model should be associated with a documented purpose, responsible owner, configuration, validation status, and approved application context. Deployment changes require controlled testing to confirm interface compatibility and acceptable operational behavior. Correlation records connect business requests with active model deployments, allowing teams to

investigate outcomes and reproduce operational conditions. Human review remains available for uncertain, sensitive, or exceptional results. Governance therefore becomes part of routine application operation rather than a separate administrative activity.

Lifecycle oversight completes the framework by connecting technical monitoring with model and business performance. Operational measures include response time, request volume, processing failures, endpoint availability, and resource consumption. Model-related observations may include confidence distribution, input variation, and outcome consistency where suitable reference information exists. Detected changes can trigger investigation, recalibration, redeployment, or temporary withdrawal of a model. The framework consequently supports controlled evolution from initial deployment through continuing supervision and eventual retirement. This lifecycle perspective transforms individual AI services into maintainable enterprise capabilities aligned with application reliability and organizational accountability.

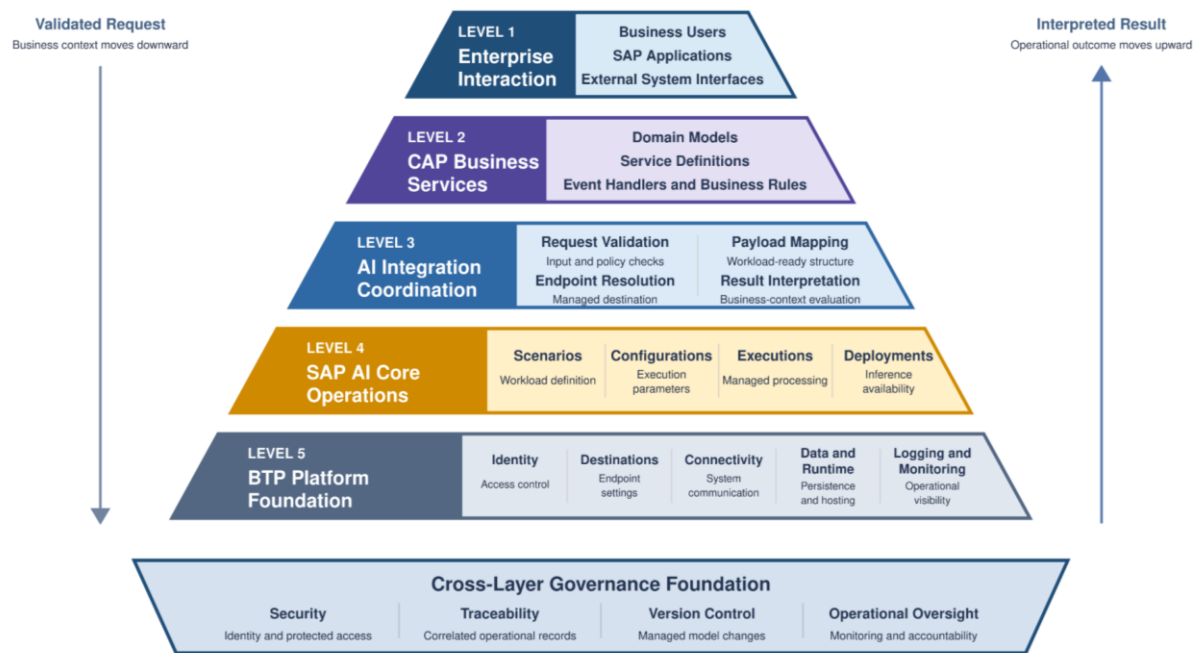


Figure 4: Proposed Layered Framework for Operationalizing AI Across SAP Enterprise Applications

## VI. FRAMEWORK IMPLEMENTATION AND ENTERPRISE APPLICATION WORKFLOW

### 6.1 CAP Application and Service Implementation

Enterprise implementation begins with a bounded use case that contains a defined business request, identifiable application consumer, measurable response expectation, and suitable artificial intelligence function. The use case may involve classification, prediction, recommendation, document processing, or operational prioritization. Business requirements are separated from model-specific implementation details before service construction begins. This separation allows the enterprise process to remain stable when model versions, runtime resources, or deployment endpoints change. Implementation responsibilities are distributed across the CAP application, SAP AI Core workload, and supporting SAP BTP services.

The CAP domain model defines the business entities, relationships, validation constraints, processing states, and persistence requirements associated with the selected use case. Core Data Services entities can represent incoming requests, contextual attributes, model references, interpreted outcomes, and operational timestamps. Associations connect processing records with relevant enterprise objects without duplicating business information. Service definitions expose only the entities and actions required by authorized consumers. Domain construction therefore preserves business meaning while providing a consistent data structure for application processing, artificial intelligence invocation, and operational traceability.

Application programming interfaces are designed around recognizable enterprise activities rather than technical model operations. Event handlers validate mandatory fields, enrich request data, apply business rules, and determine whether inference should proceed. Accepted information is transformed into the payload structure required by the deployed model endpoint. Interface contracts define input properties, response fields, processing status, and exception conditions. Raw model output is interpreted within the CAP service before being returned to the consuming application. This approach protects enterprise users from internal workload complexity and maintains a stable service boundary.

## **6.2 AI Core Workload Integration**

Artificial intelligence models are packaged with preprocessing functions, inference logic, response formatting, runtime libraries, and dependency specifications. Container images provide a consistent execution environment across development, validation, and production stages. SAP AI Core scenarios describe the workload purpose, while executable specifications identify the image, commands, parameters, and resource requirements. Configuration artifacts connect executable specifications with environment-specific values and data references. Version identifiers associate the trained model, container image, executable specification, and validation record, supporting controlled model promotion and investigation of operational results.

SAP AI Core manages training, evaluation, batch processing, and online inference through configurations, executions, and deployments. Executions provide controlled processing for scheduled or development workloads, while deployments expose approved models through managed inference endpoints. The CAP service invokes the active deployment instead of executing the model within the application runtime. This separation prevents model resource consumption from affecting the business service directly. Workload capacity, deployment settings, and model versions can therefore be modified without redesigning the enterprise-facing application programming interface.

Deployment validation examines endpoint availability, payload compatibility, response structure, processing latency, resource stability, and failure behavior. A model is promoted only when its runtime output satisfies the contract expected by the CAP service. Deployment references are recorded so that each response can be connected to the model version that produced it. Revised workloads follow a controlled configuration and verification procedure before production use. Rollback preparation allows a previous deployment to be restored when unexpected behavior occurs. This process supports operational continuity while enabling gradual model improvement.

## **6.3 BTP Service Orchestration and Runtime Workflow**

SAP BTP services coordinate identity, destination resolution, connectivity, persistence, and monitoring across the integrated workflow. Identity services authenticate calling users or

applications and evaluate assigned roles before protected operations are executed. Destination configurations maintain endpoint locations and connection properties outside application source code. Connectivity services support communication with protected enterprise systems when additional business information is required. Environment-specific settings allow development, test, and production landscapes to use separate endpoints and credentials while retaining the same CAP implementation and service contract.

Runtime interaction begins when an enterprise application submits a secured business request to the CAP service. The service validates authorization, checks domain data, applies business rules, and resolves the configured destination. A transformed payload and correlation identifier are sent to the SAP AI Core deployment for inference. The deployment performs preprocessing and model execution before returning the prediction, processing status, and model reference. The CAP service interprets this response and constructs an application-oriented outcome. Logging services record request timing, endpoint status, response duration, and processing results throughout the interaction shown in the figure.

Exception management protects the enterprise process when authorization fails, input data is incomplete, the destination cannot be resolved, the inference endpoint is unavailable, or the returned response is invalid. Each failure is classified and connected to the original correlation identifier. Temporary technical failures may follow controlled retry rules, while validation and authorization errors require correction before resubmission. Monitoring thresholds identify repeated failures, abnormal response times, and resource constraints. Operational teams can use these records to investigate incidents, adjust configurations, improve workload stability, and maintain reliable artificial intelligence services across enterprise applications.

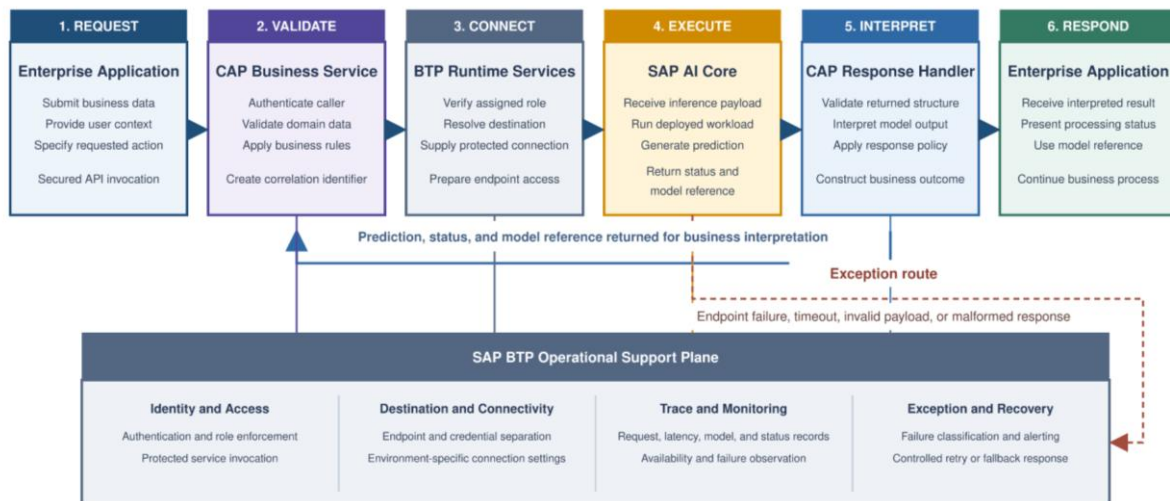


Figure 5: Comparative Communication Performance under Increasing Transaction Volumes

## VII. RESULTS, COMPARATIVE EVALUATION, AND STRATEGIC IMPLICATIONS

### 7.1 Quantitative Performance Results

Prototype testing showed that the integrated framework completed enterprise artificial intelligence requests more efficiently than the fragmented baseline arrangement. Average end-to-end response time decreased from 2.84 seconds to 1.62 seconds, representing an approximate reduction of 43

percent. Successful request completion increased from 88.1 percent to 97.4 percent under the same controlled test conditions. CAP-based validation prevented incomplete requests from reaching the inference endpoint, while destination reuse and structured response handling reduced repeated connection and transformation activities during runtime processing.

Integration effort was assessed using a normalized index in which lower values represented reduced implementation burden. The fragmented baseline recorded an effort value of 78, while the proposed framework recorded 42. Deployment consistency increased from 46 to 88 because CAP service contracts, SAP AI Core configurations, and destination properties were maintained as identifiable implementation artifacts. The integrated arrangement reduced the need to reproduce authentication, endpoint, payload, and error-handling logic across individual enterprise applications. This improvement supported more predictable deployment across development, testing, and production environments.

Service interoperability increased from 51 to 90, lifecycle control improved from 38 to 86, and scalability increased from 49 to 87. Stable application programming interfaces allowed enterprise applications to consume interpreted outcomes without depending on internal model implementation. SAP AI Core deployment references strengthened model version traceability, while SAP BTP monitoring services improved visibility across request processing and endpoint behavior. Figure 6 presents the comparative pattern. The measurements represent normalized prototype observations rather than universal platform benchmarks and should therefore be interpreted within the defined implementation and test environment.

## **7.2 Qualitative and Comparative Findings**

Qualitative evaluation showed that the proposed framework produced clearer responsibility boundaries than fragmented integration practices. CAP maintained domain definitions, request validation, and business-oriented response construction. SAP AI Core controlled model workload execution and deployment, while SAP BTP services managed identity, destinations, connectivity, and operational monitoring. The fragmented baseline distributed these responsibilities across application-specific code and manually maintained connection settings. Centralized platform responsibilities reduced duplication and made the integrated implementation easier to inspect, modify, and support across multiple enterprise application scenarios.

Deployment consistency improved because model endpoints, connection properties, service contracts, and workload configurations were managed separately. Fragmented practices often required each consuming application to maintain its own endpoint logic, authentication handling, payload conversion, and failure processing. Such duplication increased implementation effort and created inconsistent runtime behavior. The proposed framework replaced these repeated arrangements with reusable CAP services and governed destination configurations. Model deployments could therefore be updated without changing every consuming application, provided that the established request and response contracts remained compatible.

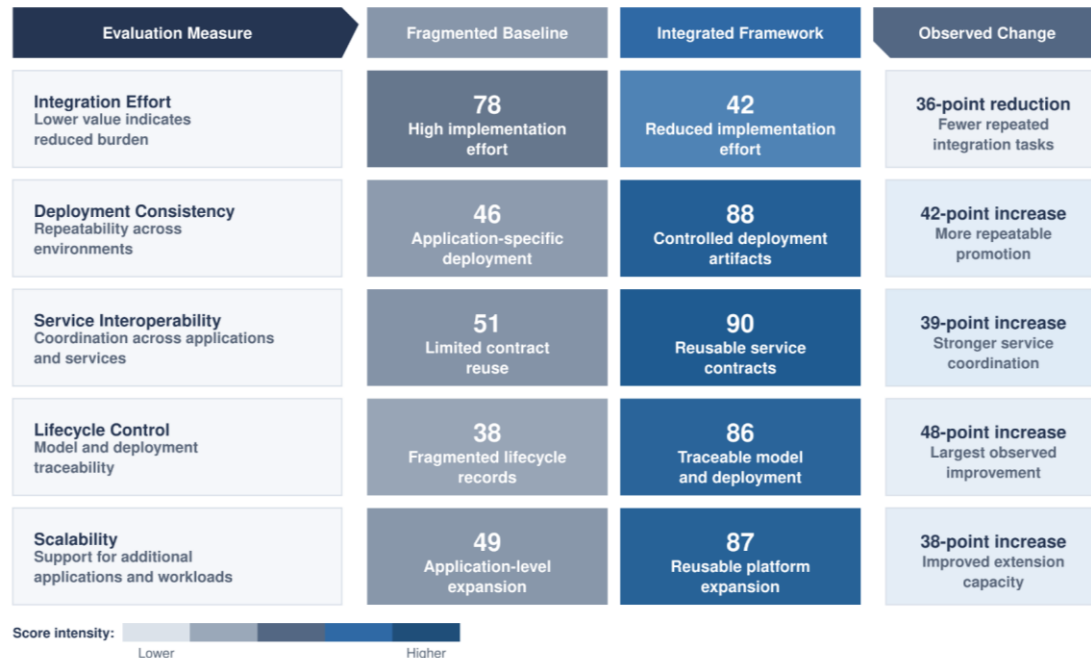


Figure 6: Enterprise Adoption and Governance Roadmap for Asynchronous Service Coordination

Empirical patterns indicated that the strongest improvements occurred in lifecycle control, deployment consistency, and service interoperability. Response-time improvement was important, but the broader operational value resulted from predictable service behavior and traceable workload execution. Correlation identifiers connected application requests with CAP processing, inference responses, and monitoring records. Exception classifications made authorization failures, invalid payloads, unavailable endpoints, and malformed responses easier to distinguish. These findings demonstrate that operational artificial intelligence must be evaluated through integration quality, governance, and maintainability in addition to technical inference performance.

### 7.3 Strategic, Academic, and Industry Implications

The proposed framework offers strategic value by providing a reusable path for introducing artificial intelligence into enterprise applications without tightly coupling business processes to individual models. Organizations can begin with a bounded use case and extend the implementation through common domain patterns, service contracts, destinations, and monitoring practices. This approach supports phased adoption and reduces the risk associated with isolated artificial intelligence projects. Business and technology teams can assess operational evidence before increasing user volume, adding applications, or promoting revised model deployments into production environments.

Academic significance arises from connecting application architecture, artificial intelligence workload management, and platform governance within one operational framework. The results show that enterprise artificial intelligence effectiveness depends on the interaction between domain modelling, service orchestration, deployment control, and runtime observability. The study also demonstrates the value of combining quantitative performance measures with qualitative architectural assessment. Future research can apply the framework to different model

types, industry processes, transaction volumes, and deployment regions to examine whether the observed patterns remain consistent across broader enterprise environments.

Industry adoption should consider the prototype boundaries when interpreting the results. The evaluation used a controlled implementation, defined service contracts, selected workload conditions, and a limited range of failure scenarios. Extended production traffic, multiple concurrent models, regional connectivity differences, retraining frequency, and long-term operating cost require further validation. Future work can examine automated deployment pipelines, advanced model monitoring, workload scaling policies, and formal governance integration. The present findings nevertheless provide practical evidence that coordinated SAP CAP, SAP AI Core, and SAP BTP services can improve enterprise artificial intelligence operationalization.

Table 2: Comparative Results for Communication Performance, Message Reliability, and Failure Recovery

| Evaluation Measure            | Baseline Approach | Proposed Framework | Observed Change               | Practical Interpretation                             |
|-------------------------------|-------------------|--------------------|-------------------------------|------------------------------------------------------|
| Average response time         | 2.84 seconds      | 1.62 seconds       | 43% reduction                 | Faster completion of enterprise application requests |
| Successful request completion | 88.1%             | 97.4%              | 9.3 percentage-point increase | Fewer incomplete or failed transactions              |
| Integration effort            | 78                | 42                 | 36-point reduction            | Reduced configuration and implementation burden      |
| Deployment consistency        | 46                | 88                 | 42-point increase             | More repeatable deployment across environments       |
| Service interoperability      | 51                | 90                 | 39-point increase             | Stronger coordination across applications and        |

|                      |                              |                                   |                     |                                                          |
|----------------------|------------------------------|-----------------------------------|---------------------|----------------------------------------------------------|
|                      |                              |                                   |                     | services                                                 |
| Lifecycle control    | 38                           | 86                                | 48-point increase   | Improved model and deployment traceability               |
| Scalability          | 49                           | 87                                | 38-point increase   | Better support for additional applications and workloads |
| Exception visibility | Application-specific records | Correlated and classified records | Improved visibility | Faster identification of failed processing stages        |

### VIII. CONCLUSION AND FUTURE RESEARCH

Enterprise organizations increasingly require artificial intelligence capabilities that operate as reliable parts of business applications rather than isolated technical experiments. Fragmented initiatives commonly separate model development from application services, identity controls, connectivity configuration, monitoring, and lifecycle management. This separation increases integration effort and produces inconsistent runtime behavior across enterprise environments. The research addressed this problem by examining how SAP Cloud Application Programming Model, SAP AI Core, and SAP Business Technology Platform services can be combined into a coordinated architecture for operational artificial intelligence.

The proposed framework distributes responsibilities across five connected layers: enterprise interaction, CAP business services, artificial intelligence integration coordination, SAP AI Core operations, and the SAP BTP platform foundation. CAP maintains domain models, service contracts, validation rules, and business-oriented response handling. SAP AI Core manages model packaging, workload configuration, execution, and deployment. Supporting SAP BTP services provide identity, destinations, connectivity, persistence, logging, and monitoring. Cross-layer governance connects these capabilities through security, traceability, version control, and operational oversight.

Prototype implementation demonstrated that the framework could support a complete request pathway from enterprise application invocation to interpreted business response. CAP services validated incoming information and constructed model-compatible payloads before inference execution. SAP AI Core processed requests through managed deployments and returned predictions with status and model-reference information. The CAP response layer converted technical output into application-oriented results. Correlation identifiers and monitoring records connected each processing stage, allowing operational teams to trace successful transactions,

failures, response times, and deployment references.

Comparative evaluation indicated improvements in response time, request completion, integration effort, deployment consistency, service interoperability, lifecycle control, and scalability. Average prototype response time decreased, while successful request completion increased under the controlled test conditions. The largest normalized improvement occurred in lifecycle control, reflecting clearer relationships among model versions, deployment configurations, and operational results. Reduced integration effort resulted from reusable service contracts and centralized destination settings. These findings show that enterprise artificial intelligence value depends on integration quality and operational discipline in addition to model performance.

The study contributes to enterprise artificial intelligence operationalization by defining a practical method for connecting business applications with managed model workloads. The framework treats artificial intelligence as an application capability supported by stable interfaces, protected communication, controlled deployment, and observable runtime behavior. This position extends beyond direct application-to-model integration and emphasizes the operational responsibilities required for production use. The resulting architecture supports gradual adoption, workload replacement, failure management, and reuse across multiple enterprise scenarios without embedding model-specific complexity inside each consuming application.

The research also contributes to SAP application architecture by clarifying the complementary responsibilities of CAP, SAP AI Core, and SAP BTP services. CAP provides the business semantics and service abstraction required by enterprise applications, while SAP AI Core provides the workload environment needed for model execution. SAP BTP services establish the identity, connectivity, configuration, and monitoring foundation that connects these layers. This architectural separation provides a basis for further investigation into service-oriented artificial intelligence integration within SAP application landscapes and related enterprise platform environments.

Enterprise architects and developers can use the framework to define implementation boundaries, allocate technical responsibilities, and establish repeatable integration practices. Architects can identify which functions belong within business services, model workloads, or shared platform services. Developers can reuse domain patterns, service contracts, payload transformations, destination configurations, and exception-handling procedures across applications. Operations teams can apply correlation identifiers, deployment references, and monitoring measures to investigate runtime behavior. These practices reduce duplicated integration logic and support more maintainable enterprise application development.

Governance and strategic implications extend across security, accountability, lifecycle control, and investment planning. Identity enforcement and protected destination configuration reduce exposure of connection information. Traceable requests and model references support investigation and operational accountability. Controlled deployment procedures allow revised workloads to be validated and promoted without disrupting established business interfaces. Organizations can begin with bounded use cases and expand only after reviewing operational evidence. This phased approach allows artificial intelligence investment decisions to consider reliability, reuse, governance readiness, and business value alongside technical feasibility.

The study remains limited by its prototype scope, selected use case, controlled workload conditions, and restricted production history. Future research should examine higher transaction volumes, multiple concurrent models, regional deployments, automated delivery pipelines, model monitoring, controlled retraining, and long-term operating costs. Additional studies can evaluate

industry-specific processes, broader governance requirements, and relationships between operational measures and business outcomes. Future validation across diverse enterprise environments will strengthen generalizability. The concluding position is that coordinated application, workload, and platform services provide a practical foundation for dependable enterprise artificial intelligence.

## REFERENCES

1. Kreuzberger, D., Kühl, N., & Hirschl, S. (2023). Machine learning operations (MLOps): Overview, definition, and architecture. *IEEE Access*, 11, 31866–31879. <https://doi.org/10.1109/ACCESS.2023.3262138>
2. Steidl, M., Felderer, M., & Ramler, R. (2023). The pipeline for the continuous development of artificial intelligence models: Current state of research and practice. *Journal of Systems and Software*, 199, Article 111615. <https://doi.org/10.1016/j.jss.2023.111615>
3. Sundberg, L., & Holmström, J. (2023). Democratizing artificial intelligence: How no-code AI can leverage machine learning operations. *Business Horizons*, 66(6), 777–788. <https://doi.org/10.1016/j.bushor.2023.04.003>
4. Bork, D., Ali, S. J., & Dinev, G. M. (2023). AI-enhanced hybrid decision management. *Business & Information Systems Engineering*, 65(2), 179–199. <https://doi.org/10.1007/s12599-023-00790-2>
5. Ahmad, K., Abdelrazek, M., Arora, C., Bano, M., & Grundy, J. (2023). Requirements engineering for artificial intelligence systems: A systematic mapping study. *Information and Software Technology*, 158, Article 107176. <https://doi.org/10.1016/j.infsof.2023.107176>
6. Schneider, J., Abraham, R., Meske, C., & vom Brocke, J. (2023). Artificial intelligence governance for businesses. *Information Systems Management*, 40(3), 229–249. <https://doi.org/10.1080/10580530.2022.2085825>
7. Faubel, L., Schmid, K., & Eichelberger, H. (2023). MLOps challenges in Industry 4.0. *SN Computer Science*, 4(6), Article 828. <https://doi.org/10.1007/s42979-023-02282-2>
8. Madan, R., & Ashok, M. (2023). AI adoption and diffusion in public administration: A systematic literature review and future research agenda. *Government Information Quarterly*, 40(1), Article 101774. <https://doi.org/10.1016/j.giq.2022.101774>
9. Alonso, J., Orue-Echevarria, L., Casola, V., Torre, A. I., Huarte, M., Osaba, E., & Lobo, J. L. (2023). Understanding the challenges and novel architectural models of multi-cloud native applications: A systematic literature review. *Journal of Cloud Computing*, 12(1), Article 6. <https://doi.org/10.1186/s13677-022-00367-6>
10. Nahar, N., Zhang, H., Lewis, G., Zhou, S., & Kästner, C. (2023). A meta-summary of challenges in building products with ML components: Collecting experiences from 4758+ practitioners. *Proceedings of the 2023 IEEE/ACM 2nd International Conference on AI Engineering: Software Engineering for AI*, 171–183. <https://doi.org/10.1109/CAIN58948.2023.00034>
11. Shivashankar, K., & Martini, A. (2022). Maintainability challenges in ML: A systematic literature review. *Proceedings of the 2022 48th Euromicro Conference on Software Engineering and Advanced Applications*, 60–67. <https://doi.org/10.1109/SEAA56994.2022.00018>
12. Serban, A., & Visser, J. (2022). Adapting software architectures to machine learning

- challenges. Proceedings of the 2022 IEEE International Conference on Software Analysis, Evolution and Reengineering, 152–163. <https://doi.org/10.1109/SANER53432.2022.00029>
13. Siala, H., & Wang, Y. (2022). SHIFTing artificial intelligence to be responsible in healthcare: A systematic review. *Social Science & Medicine*, 296, Article 114782. <https://doi.org/10.1016/j.socscimed.2022.114782>
  14. Sousa, P., & Vasconcelos, A. (2022). Enterprise architecture and cartography: From practice to theory, from representation to design. Springer. <https://doi.org/10.1007/978-3-030-96264-7>
  15. Paleyes, A., Urma, R. G., & Lawrence, N. D. (2022). Challenges in deploying machine learning: A survey of case studies. *ACM Computing Surveys*, 55(6), Article 114, 1–29. <https://doi.org/10.1145/3533378>
  16. Mäntymäki, M., Minkkinen, M., Birkstedt, T., & Viljanen, M. (2022). Defining organizational AI governance. *AI and Ethics*, 2(4), 603–609. <https://doi.org/10.1007/s43681-022-00143-x>
  17. Mboweni, T., Masombuka, T., & Dongmo, C. (2022). A systematic review of machine learning DevOps. Proceedings of the 2022 International Conference on Electrical, Computer and Energy Technologies, 1–6. <https://doi.org/10.1109/ICECET55527.2022.9872968>
  18. Kolltveit, A. B., & Li, J. (2022). Operationalizing machine learning models: A systematic literature review. Proceedings of the 1st Workshop on Software Engineering for Responsible AI, 1–8. <https://doi.org/10.1145/3526073.3527584>
  19. Martínez-Fernández, S., Bogner, J., Franch, X., Oriol, M., Siebert, J., Trendowicz, A., Vollmer, A. M., & Wagner, S. (2022). Software engineering for AI-based systems: A survey. *ACM Transactions on Software Engineering and Methodology*, 31(2), Article 37, 1–59. <https://doi.org/10.1145/3487043>
  20. Symeonidis, G., Nerantzis, E., Kazakis, A., & Papakostas, G. A. (2022). MLOps: Definitions, tools and challenges. Proceedings of the 2022 IEEE 12th Annual Computing and Communication Workshop and Conference, 453–460. <https://doi.org/10.1109/CCWC54503.2022.9720902>
  21. Testi, M., Ballabio, M., Frontoni, E., Iannello, G., Moccia, S., Soda, P., & Vessio, G. (2022). MLOps: A taxonomy and a methodology. *IEEE Access*, 10, 63606–63618. <https://doi.org/10.1109/ACCESS.2022.3181730>
  22. Raisch, S., & Krakowski, S. (2021). Artificial intelligence and management: The automation-augmentation paradox. *Academy of Management Review*, 46(1), 192–210. <https://doi.org/10.5465/amr.2018.0072>
  23. Mikalef, P., & Gupta, M. (2021). Artificial intelligence capability: Conceptualization, measurement calibration, and empirical study on its impact on organizational creativity and firm performance. *Information & Management*, 58(3), Article 103434. <https://doi.org/10.1016/j.im.2021.103434>
  24. Jöhnk, J., Weißert, M., & Wyrski, K. (2021). Ready or not, AI comes: An interview study of organizational AI readiness factors. *Business & Information Systems Engineering*, 63(1), 5–20. <https://doi.org/10.1007/s12599-020-00676-7>
  25. Studer, S., Bui, T. B., Drescher, C., Hanuschkin, A., Winkler, L., Peters, S., & Müller, K. R. (2021). Towards CRISP-ML(Q): A machine learning process model with quality assurance methodology. *Machine Learning and Knowledge Extraction*, 3(2), 392–413. <https://doi.org/10.3390/make3020020>
  26. Haefner, N., Wincent, J., Parida, V., & Gassmann, O. (2021). Artificial intelligence and

- innovation management: A review, framework, and research agenda. *Technological Forecasting and Social Change*, 162, Article 120392. <https://doi.org/10.1016/j.techfore.2020.120392>
27. Granlund, T., Kopponen, A., Stirbu, V., Myllyaho, L., & Mikkonen, T. (2021). MLOps challenges in multi-organization setup: Experiences from two real-world cases. *Proceedings of the 2021 IEEE/ACM 1st Workshop on AI Engineering: Software Engineering for AI*, 82–88. <https://doi.org/10.1109/WAIN52551.2021.00019>
  28. Rakova, B., Yang, J., Cramer, H., & Chowdhury, R. (2021). Where responsible AI meets reality: Practitioner perspectives on enablers for shifting organizational practices. *Proceedings of the ACM on Human-Computer Interaction*, 5(CSCW1), Article 7, 1–23. <https://doi.org/10.1145/3449081>
  29. Ashmore, R., Calinescu, R., & Paterson, C. (2021). Assuring the machine learning lifecycle: Desiderata, methods, and challenges. *ACM Computing Surveys*, 54(5), Article 111, 1–39. <https://doi.org/10.1145/3453444>
  30. Berente, N., Gu, B., Recker, J., & Santhanam, R. (2021). Managing artificial intelligence. *MIS Quarterly*, 45(3), 1433–1450. <https://doi.org/10.25300/MISQ/2021/16274>