Requirements Specification: Intelligent Multi-Cloud Supply Chain & Operations Platform (IMSOP)

1. Introduction

This document outlines the detailed requirements for the **Intelligent Multi-Cloud Supply Chain & Operations Platform (IMSOP)**. IMSOP is a cloud-native platform designed to provide real-time visibility, predictive analytics, and intelligent automation for complex supply chain operations. The platform will be built on a modern, microservices-based architecture, leveraging a multi-cloud strategy to ensure resilience and avoid vendor lock-in. It will integrate a wide range of technologies, including .NET, Python, Kafka, Kubernetes, and various AI/ML frameworks, to deliver a robust and scalable solution.

2. Business Objectives

The primary business goals for IMSOP are as follows:

Objective	Description
Real-Time Visibility	Provide a unified, near real-time view of the entire supply chain by ingesting and processing data from disparate sources.
Predictive Insights	Utilize machine learning to predict operational delays, identify potential bottlenecks, and detect anomalies before they impact the business.
Intelligent Automation	Enable automated responses to predicted events and provide a conversational AI interface for human-in-the-loop decision-making.
Multi-Tenancy	Support multiple enterprise customers with secure data isolation and role-based access control.
Multi-Cloud Operation	Operate seamlessly across both Microsoft Azure and Amazon Web Services (AWS) to optimize cost, performance, and reliability.

3. Target Users

The platform is designed to serve a variety of user roles within an organization and its partner ecosystem:

User Role	Description of Interaction with IMSOP
Operations Managers	Monitor dashboards for real-time operational health, receive alerts on predicted disruptions, and use the platform to make informed decisions.
Data Analysts	Query historical and real-time data, build custom reports, and use the platform's analytics tools to identify trends and improvement opportunities.
External Partner Systems	Integrate with IMSOP via APIs to exchange data, such as shipment status, inventory levels, and delivery confirmations.
Customer Support Teams	Use the conversational AI (chatbot) to quickly answer customer inquiries about order status, expected delivery times, and other operational details.
Automation Workflows	Leverage platform APIs and events to trigger automated workflows in systems like Power Platform or Logic Apps.

4. Functional Requirements

Functional requirements are organized by Domain-Driven Design (DDD) bounded contexts to ensure a clear separation of concerns and a modular, scalable architecture.

4.1 Identity & Access Management

This context is responsible for user authentication, authorization, and securing access to the platform's resources.

Requirement ID	Description
IAM-001	User authentication will be handled via OAuth 2.0 and OpenID Connect, integrating with Microsoft Entra ID as the primary identity provider.
IAM-002	The system will implement Role-Based Access Control (RBAC) with predefined roles: Admin, Analyst, Operator, and Bot.
IAM-003	Third-party systems and internal microservices will use managed identities for secure, password-free access to Azure resources.
IAM-004	An API for token introspection will be provided to allow third-party integrations to validate access tokens.

4.2 Ingestion & Integration

This context handles the consumption of data from various external and internal sources.

Requirement ID	Description
ING-001	The platform must be able to consume data from third-party logistics providers via their REST APIs.
ING-002	The system will subscribe to Kafka topics to ingest streaming data from IoT devices and other real-time sources.
ING-003	A WebSocket endpoint will be provided for near real-time telemetry data ingestion.
ING-004	All incoming data will be validated against a predefined schema, and normalized into a canonical format.
ING-005	The platform will support schema versioning using AsyncAPI to manage changes in event-driven communication.

4.3 Operations Management

This context is the core of the platform, responsible for tracking and managing supply chain operations.

Requirement ID	Description
OPS-001	The system will track the status and location of shipments, orders, and assets in near real-time.
OPS-002	Operational entities will be managed using state machines to represent their lifecycle (e.g., Order: Placed -> Confirmed -> Shipped -> Delivered).
OPS-003	Transactional data will be persisted in a relational database (PostgreSQL or Azure SQL).
OPS-004	State changes and key business events will be published to Kafka topics to enable event-driven workflows.

4.4 Analytics & Intelligence

This context focuses on processing data to extract insights and train machine learning models.

Requirement ID	Description
AI-001	A dedicated Python microservice will be responsible for data preprocessing, feature extraction, and ML model inference.
AI-002	Batch analytics will be performed using Apache Spark on Azure Databricks to process large volumes of historical data.
AI-003	The platform will implement MLOps practices for managing the lifecycle of machine learning models, including versioning and deployment.
AI-004	The results of analytics and ML model outputs will be stored in Azure Data Lake Gen2 and Cosmos DB for further analysis and visualization.

4.5 Prediction & Anomaly Detection

This context uses machine learning models to provide predictive insights and identify potential issues.

Requirement ID	Description
PRED-001	The system will include a machine learning model to predict potential delays in shipments based on historical data and real-time events.
PRED-002	A demand forecasting model will be developed to predict future product demand based on historical sales data and market trends.
PRED-003	The platform will implement outlier detection algorithms to identify anomalous patterns in operational data that may indicate fraud or other issues.
PRED-004	Automated pipelines will be established for retraining machine learning models to ensure they remain accurate over time.

4.6 Conversational AI

This context provides a natural language interface for interacting with the platform.

Requirement ID	Description
CONV-001	A chatbot microservice will be developed using FastAPI to provide a conversational interface to the platform.
CONV-002	Users will be able to query operational data using natural language (e.g., “What is the status of order #12345?”).
CONV-003	The chatbot will integrate with the Analytics & Intelligence context to provide users with insights and predictions.
CONV-004	The chatbot will be accessible via both a REST API and a WebSocket for real-time interaction.

4.7 Visualization & Reporting

This context is responsible for presenting data and insights to users in a clear and actionable format.

Requirement ID	Description
VIS-001	A GraphQL API will be provided to enable efficient data fetching for frontend applications.
VIS-002	The platform will feature real-time dashboards that are updated via WebSockets to provide an up-to-the-minute view of operations.
VIS-003	Data visualizations will be generated using Plotly and Matplotlib to create interactive charts and graphs.
VIS-004	Users will be able to export reports and data visualizations in various formats (e.g., PDF, CSV) via a REST API.

5. Non-Functional Requirements

Non-functional requirements define the system’s quality attributes and operational characteristics.

5.1 Performance

Requirement ID	Description
PERF-001	All APIs will be designed to be asynchronous and non-blocking, utilizing .NET's `async/await` pattern to maximize throughput.
PERF-002	The platform must be horizontally scalable, with the ability to add or remove service instances automatically based on load, orchestrated by Kubernetes.
PERF-003	A distributed caching layer using Redis will be implemented to reduce latency for frequently accessed data.
PERF-004	The data ingestion pipeline, powered by Kafka, must support high-throughput data streams without data loss.

5.2 Reliability

Requirement ID	Description
REL-001	The system will implement fault tolerance patterns such as circuit breakers and retries to handle transient failures in downstream services.
REL-002	All microservices will expose health check and readiness probes to enable effective monitoring and orchestration by Kubernetes.
REL-003	The platform will be designed for graceful degradation, ensuring that the failure of a non-critical component does not bring down the entire system.
REL-004	Dead-letter queues will be used in the messaging system to handle messages that cannot be processed successfully, allowing for later analysis and reprocessing.

5.3 Security

Requirement ID	Description
SEC-001	All application secrets, connection strings, and API keys will be stored securely in Azure Key Vault.
SEC-002	A zero-trust networking model will be adopted, where all network traffic between services is authenticated and encrypted.
SEC-003	All communication channels, both internal and external, will be encrypted using TLS.
SEC-004	The platform will implement API throttling and rate limiting to protect against denial-of-service attacks and ensure fair usage.
SEC-005	The GraphQL schema will be designed with security in mind, preventing unauthorized data access and overly complex queries.

5.4 Observability

Requirement ID	Description
OBS-001	A centralized logging solution using the ELK Stack (Elasticsearch, Logstash, Kibana) will be implemented to aggregate logs from all microservices.
OBS-002	Distributed tracing will be used to monitor requests as they flow through the various microservices, enabling rapid root cause analysis.
OBS-003	The platform will expose key performance metrics for monitoring and alerting, integrated with Azure Monitor and Application Insights.
OBS-004	The team will establish clear workflows for root cause analysis of production incidents, supported by the observability platform.

6. System Architecture

The IMSOP platform follows a microservices architecture with clear separation of concerns through Domain-Driven Design bounded contexts. The architecture is

designed to be cloud-agnostic, with deployment targets across both Microsoft Azure and Amazon Web Services.

6.1 Frontend Layer

The frontend layer consists of modern web applications built with either React or Angular, utilizing TypeScript for type safety. The frontend communicates with backend services through multiple channels:

- **GraphQL API** for efficient data fetching with precise field selection
- **REST APIs** for standard CRUD operations and third-party integrations
- **WebSocket connections** for real-time updates to dashboards and notifications

6.2 API Gateway

Azure API Management serves as the central entry point for all external requests. The gateway provides:

- Request routing to appropriate microservices
- Authentication and authorization enforcement
- Rate limiting and throttling
- API versioning and documentation (OpenAPI/Swagger)
- Monitoring and analytics

6.3 Microservices Layer

Each bounded context is implemented as one or more microservices, deployed as Docker containers and orchestrated by Kubernetes. Services communicate through:

- **Synchronous communication:** REST APIs and GraphQL for request-response patterns
- **Asynchronous communication:** Kafka topics for event-driven workflows and Azure Service Bus for command messages

6.4 Data Layer

The data layer employs a polyglot persistence strategy, selecting the appropriate database technology for each use case:

Data Store	Use Case
PostgreSQL / Azure SQL	Transactional data requiring ACID guarantees (orders, shipments, assets)
MongoDB / Cosmos DB	Semi-structured data with flexible schemas (telemetry, logs, events)
Redis Cache	High-speed caching layer for frequently accessed data
Azure Data Lake Gen2	Long-term storage of analytics data and ML training datasets

6.5 Messaging and Streaming

Apache Kafka serves as the backbone for event streaming, enabling:

- Real-time data ingestion from IoT devices and third-party systems
- Event-driven microservices communication
- Stream processing for real-time analytics

Azure Service Bus complements Kafka for command-based messaging patterns where guaranteed delivery and ordering are critical.

6.6 Analytics and ML Layer

The analytics layer consists of:

- **Python microservices** (FastAPI) for real-time ML inference and data preprocessing
- **Apache Spark** on Azure Databricks for batch analytics and feature engineering
- **ML model registry** for versioning and managing trained models
- **MLOps pipelines** for automated model training, validation, and deployment

7. Deployment and Infrastructure

7.1 Containerization

All services are containerized using Docker, with images optimized for size and security. Docker Compose is used for local development environments, while Kubernetes orchestrates production deployments.

7.2 Kubernetes Orchestration

Kubernetes provides:

- Automated deployment and rollback
- Horizontal pod autoscaling based on CPU/memory metrics
- Service discovery and load balancing
- Health checks and self-healing
- Configuration and secrets management

7.3 Infrastructure as Code

All infrastructure is defined and provisioned using:

- **Terraform** for multi-cloud resource provisioning
- **Azure Bicep** for Azure-specific resources
- **Helm charts** for Kubernetes application deployment

7.4 CI/CD Pipelines

Continuous integration and deployment pipelines are implemented using:

- **Azure DevOps Pipelines** for .NET microservices
- **GitHub Actions** for Python services and infrastructure code
- **Jenkins** for legacy integrations and custom workflows

Pipelines include:

- Automated unit and integration testing

- Code quality analysis and security scanning
- Container image building and scanning
- Automated deployment to staging and production environments

8. Security Requirements

8.1 Authentication and Authorization

Requirement ID	Description
SEC-AUTH-001	All user authentication will be handled through Microsoft Entra ID using OAuth 2.0 and OpenID Connect.
SEC-AUTH-002	Service-to-service authentication will use managed identities to eliminate the need for storing credentials.
SEC-AUTH-003	API access tokens will have a maximum lifetime of 1 hour, with refresh tokens valid for 7 days.
SEC-AUTH-004	All API endpoints will enforce authorization checks based on user roles and permissions.

8.2 Data Protection

Requirement ID	Description
SEC-DATA-001	All data in transit will be encrypted using TLS 1.3 or higher.
SEC-DATA-002	Sensitive data at rest (PII, credentials) will be encrypted using AES-256.
SEC-DATA-003	Database connections will use encrypted channels and certificate validation.
SEC-DATA-004	Application secrets will be stored in Azure Key Vault and accessed via managed identities.

8.3 Network Security

Requirement ID	Description
SEC-NET-001	Microservices will be deployed in private subnets with no direct internet access.
SEC-NET-002	All ingress traffic will flow through the API Gateway with WAF (Web Application Firewall) protection.
SEC-NET-003	Network segmentation will be enforced using Virtual Networks and Network Security Groups.
SEC-NET-004	Zero trust networking will be implemented, requiring authentication for all service-to-service communication.

9. Operational Requirements

9.1 Monitoring and Alerting

Requirement ID	Description
OPS-MON-001	All microservices will emit structured logs to the centralized ELK stack.
OPS-MON-002	Application performance metrics will be collected by Azure Monitor and Application Insights.
OPS-MON-003	Distributed tracing will be implemented to track requests across microservices boundaries.
OPS-MON-004	Alerts will be configured for critical metrics (error rates, latency, resource utilization) with automated escalation.

9.2 Backup and Disaster Recovery

Requirement ID	Description
OPS-BCK-001	Database backups will be performed daily with a retention period of 30 days.
OPS-BCK-002	Critical data will be replicated across multiple availability zones for high availability.
OPS-BCK-003	Disaster recovery procedures will be documented and tested quarterly.
OPS-BCK-004	The platform will have a Recovery Time Objective (RTO) of 4 hours and Recovery Point Objective (RPO) of 1 hour.

9.3 Scalability

Requirement ID	Description
OPS-SCALE-001	The platform must support horizontal scaling of all stateless microservices.
OPS-SCALE-002	Kubernetes Horizontal Pod Autoscaler will automatically scale services based on CPU and memory utilization.
OPS-SCALE-003	Database read replicas will be used to distribute read traffic and improve performance.
OPS-SCALE-004	The platform must be able to handle a 10x increase in traffic without degradation in performance.

10. Compliance and Governance

10.1 Data Governance

The platform will implement data governance policies to ensure:

- Data quality and consistency across systems
- Data lineage tracking for audit and compliance purposes

- Data retention policies aligned with regulatory requirements
- Data access controls based on the principle of least privilege

10.2 Compliance Requirements

The platform will be designed to support compliance with relevant regulations, including:

- **GDPR** (General Data Protection Regulation) for handling EU citizen data
- **SOC 2** (Service Organization Control 2) for security and availability controls
- **ISO 27001** for information security management

11. Success Metrics

The success of the IMSOP platform will be measured using the following key performance indicators:

Metric	Target
System Availability	99.9% uptime (excluding planned maintenance)
API Response Time	95th percentile < 500ms for read operations
Data Ingestion Latency	End-to-end latency < 5 seconds for real-time data
Prediction Accuracy	Delay prediction accuracy > 85%
User Adoption	80% of target users actively using the platform within 6 months
Cost Efficiency	Operating costs within free tier limits for first 12 months

12. Future Enhancements

While not part of the initial release, the following enhancements are planned for future iterations:

- Mobile applications (iOS and Android) for on-the-go access
- Advanced analytics dashboards with customizable widgets

- Integration with additional third-party logistics providers
- Blockchain integration for supply chain traceability
- Edge computing capabilities for IoT data processing
- Multi-language support for global operations

13. Conclusion

This requirements specification provides a comprehensive foundation for building the Intelligent Multi-Cloud Supply Chain & Operations Platform. The platform leverages modern technologies and architectural patterns to deliver a scalable, secure, and intelligent solution for complex supply chain operations. By following Domain-Driven Design principles and implementing a microservices architecture, the platform is designed to evolve and adapt to changing business needs while maintaining high standards of quality and performance.

Document Version: 1.0
Last Updated: January 7, 2026
Author: Manus AI