High-Level OPC UA SaaS Architecture and Business Model Overview

This document presents a comprehensive high-level design overview for the proposed OPC UA SaaS platform. It answers the key questions raised by Kais regarding scalability, automated provisioning, subscription models, cost implications, and the role of third-party technologies. The goal is to provide a robust roadmap that ensures our solution is technically sound, scalable, and differentiated from existing competitors like TagoIO and Prosys.

Horizontal Scalability Strategy
Our design favors horizontal scalability to ensure reliability and flexibility. Each customer will be served through a dedicated OPC UA server instance, rather than a shared server, to guarantee data confidentiality and isolation. These server instances will run as containers orchestrated by Kubernetes. Auto-scaling policies will monitor resource usage (CPU, memory, connection count) and provision additional instances dynamically. This approach ensures performance even as the number of tenants grows.

We will leverage namespaces or virtual clusters in Kubernetes to isolate customer environments. This minimizes resource contention while making management more efficient. Additionally, load balancers will distribute traffic intelligently, ensuring that no server becomes a bottleneck.

Automated Instance Provisioning
To streamline onboarding, we will adopt automated provisioning workflows:
- When a new customer subscribes, the system automatically generates a new Kubernetes namespace and deploys a fresh OPC UA server container configured with customer-specific credentials.
- An Infrastructure-as-Code (IaC) tool such as Terraform or Helm will manage reproducible deployments.
- Secrets and certificates will be auto-generated and stored securely in AWS Secrets Manager or HashiCorp Vault.
- The provisioning workflow will take less than a few minutes per customer, ensuring rapid onboarding at scale.

This model not only increases operational efficiency but also ensures strong data isolation between tenants.

Security and Data Isolation
Security is a cornerstone of our architecture. Each OPC UA server will be assigned unique certificates and operate within its own namespace, preventing unauthorized cross-tenant

access. Role-Based Access Control (RBAC) will restrict administrative actions, ensuring that operators cannot accidentally interfere with other customers' environments. Audit logging will capture all actions, providing transparency and compliance support.

We also plan to support multiple security policies for OPC UA connections (Basic256Sha256, SignAndEncrypt), allowing customers to select their preferred level of encryption. This flexibility will stand out compared to competitors that typically enforce a single policy.

Proposed Pricing and Subscription Models

After studying competitors (TagoIO, Prosys, AVEVA, etc.), we identified that most use either annual subscriptions or pay-as-you-go based on message or data volume. Our proposed hybrid pricing model includes:

- **Base Subscription:** Fixed monthly or annual fee granting access to the platform.
- **Usage-Based Add-ons:** Additional charges for high data throughput (tags/sec) or extra storage.
- **Premium Features:** Advanced analytics, redundancy options, and priority support at higher tiers.

This structure ensures predictable revenue while allowing flexibility for small and large customers. Unlike competitors, we plan to emphasize transparency in billing, offering real-time dashboards where customers can track usage and projected costs, improving trust and reducing billing disputes.

Impact of Horizontal Scalability on Costs

Horizontal scaling inevitably increases infrastructure costs, as each tenant requires its own instance. However, containerization minimizes overhead compared to virtual machines. Running lightweight OPC UA servers in Docker containers ensures resource efficiency.

We will optimize costs by:

- Running idle or low-traffic instances on shared nodes with resource quotas.
- Using auto-scaling groups that spin down instances during periods of inactivity.
- Offering volume discounts for large enterprises to ensure customer retention while maintaining profitability.

Operational overhead will be reduced through automation of provisioning, monitoring, and patch management. Licensing costs will be managed by relying on open-source OPC UA stacks where possible, and negotiating enterprise licensing deals when required.

Third-Party Components and Justification

We propose leveraging the following technologies:

- **Kubernetes:** Orchestration of customer instances for scalability and resilience.
- **Terraform/Helm:** Infrastructure-as-Code for reproducible deployments.

- **AWS (EKS, Secrets Manager, CloudWatch):** Managed infrastructure, secret storage, and monitoring.
- **Prometheus + Grafana:** Metrics collection and visualization for system health.
- **Elastic Stack:** Centralized logging to track issues across all tenant instances.

Each third-party inclusion is justified by maturity, community adoption, and proven scalability in production. We will avoid over-engineering and select only components that deliver measurable value.

How Rayan Platform Stands Out

Our platform aims to differentiate from competitors by:

- Providing automated per-tenant provisioning with strong isolation (vs. shared environments).
- Offering transparent, real-time usage dashboards and billing.
- Supporting advanced analytics (predictive maintenance, churn analysis) as add-ons.
- Building modular APIs that customers can extend with their own integrations.
- Focusing on user-friendly onboarding with templates for common industrial use cases.

By combining flexibility, transparency, and strong data isolation, the Rayan Platform will appeal to both SMEs and large enterprises who value control and reliability in their SaaS provider.

Conclusion and Next Steps

We recommend moving forward with this high-level design as it addresses Kais's concerns regarding scalability, provisioning, and pricing. The next step will be to prototype a REST API component that demonstrates per-tenant provisioning in a Kubernetes environment. This prototype will validate our assumptions and help refine cost models before scaling to full development.

The proposed architecture ensures technical scalability, business viability, and differentiation from competitors. By executing this roadmap, we position the Rayan Platform as a competitive and forward-looking SaaS solution for OPC UA in industrial IoT.