Software Specification Request (SSR)

# 1. Project Overview

This project proposes the development of a reactive microservices framework for Business and Enterprise Application Integration using Semantic Web technologies. The goal is to unify heterogeneous application sources through RDF-based data modeling and inference, allowing applications to expose discoverable, context-aware APIs and frontends.

# 2. Objectives

- Integrate diverse existing applications using RDF triples

- Perform semantic inference over integrated data

- Expose use cases and interactions as a discoverable interface (API / Frontend)

- Utilize functional/reactive programming for streaming processing

# 3. Scope and Use Cases

Scope includes transforming structured/unstructured sources into RDF triples, inferring types, states, and contexts, and generating discoverable APIs and UI. Example use cases include:

- A greengrocer supply chain (supplier, seller, buyer roles)

- Business report generation based on available indicators

# 4. System Architecture

Microservice architecture orchestrated by an Augmentation Service, with functional streaming between services:

- Datasource Service

- Aggregation Service

- Alignment Service

- Activation Service

- Consumer API / Frontend

# 5. Component Specifications

• Datasource Service: ETL source to RDF SPO Triples.

• Aggregation Service: Type, State, Order inference via Classification.

• Alignment Service: Ontology matching, clustering, equivalences.

• Activation Service: Contexts, Interactions, Roles via Regression.

• API/Frontend: REST/HATEOAS interfaces for interaction with inferred use cases.

# 6. Data Flow and Inference Layers

• Aggregation: From RDF triples, infer Types (attribute sets), States (value sets), and Orders.

• Alignment: Determine entity equivalence, linking, and upper ontology alignment.

• Activation: Contexts and Interactions inferred from aligned data. Actors play roles in Contexts.

# 7. Technology Stack

- Spring Microservices with Reactive Extensions (RxJava)

- RDF4J / Neo4j for graph data management

- Graph Neural Networks, LLMs for semantic inference

- Support for Web3 identifiers (DIDs)

# 8. Integration and Interfaces

The Consumer API Service enables frontends (web, assistant, wizard) to navigate, invoke, and manage interactions within and across applications. All interactions are discoverable through metadata inferred by the Activation Service.

# 9. Administration & Helper Services

- Registry: URI-based stream results, provenance, ML-enabled

- Naming: NLP/NER alignment, upper ontology mapping

- Indexing: Context similarity, embedding-based role resolution

# 10. Next Steps / To-Do

- Define RDF graph schema for I/O at each component

- Implement functional streaming pipelines

- Synchronize backend systems with activation layer

- Design administration UIs for each service