# **Knowledge Graph Implementation**

## **Introduction**

This report summarizes the discussion on how to design and implement a Knowledge Graph (KG) for the Multi-Agent NWB Conversion and Validation Pipeline project. The KG integrates metadata, supports enrichment, and enables agents to validate and standardize diverse lab data into NWB.

## **1. What is a Knowledge Graph?**

A Knowledge Graph is a structured representation of information where **entities (nodes)** are connected by **relationships (edges)**. Unlike databases, KGs emphasize semantics, context, and relationships, enabling reasoning and query-based enrichment.

## **2. Why Use a Knowledge Graph in This Project?**

* **Metadata enrichment**: Fill missing fields using external references.
* **Standardization**: Ensure all labs follow consistent schemas.
* **Provenance tracking**: Differentiate between User, AI, and External sources.
* **Scalability**: Easily expand entities and relations without redesigning the pipeline.

## **3. Data Model (Ontology)**

Core entities:

* Dataset
* Session
* Subject
* Device
* Lab
* Protocol
* File
* ProvenanceRecord

Relations include: **SUBJECT\_OF**, **PART\_OF**, **USES\_DEVICE**, **RUN\_BY**, **FOLLOWS\_PROTOCOL**, **BELONGS\_TO**, and **ABOUT**.

Each fact is linked to a **ProvenanceRecord** to indicate origin and confidence.

## **4. Schema Validation with LinkML**

A **LinkML schema** ensures metadata follows a structured, machine-readable format. It can generate **Pydantic/JSON Schema** classes for validation before ingest into the KG. This keeps metadata consistent with NWB requirements.

## **5. Implementation Approaches**

Two main stacks were discussed:

* **Stack A: Neo4j (property graph + Cypher)**
  + Great for fast prototyping and visualization.
* **Stack B: RDF (GraphDB/Blazegraph/Neptune) + SPARQL + SHACL**
  + Strong ontology and semantic web alignment.

## **6. Example Implementations**

* **Neo4j**: Cypher queries for upserts, constraints, and enrichment rules.
* **FastAPI service**: Python API for upserting subjects and attaching provenance.
* **RDF Turtle + SHACL**: For schema validation and semantic interoperability.
* **SPARQL queries**: To check missing metadata and suggest enrichments.

## **7. Workflow Integration**

1. **Conversation Agent** extracts user/lab metadata → writes to KG.
2. **Generation Agent** enriches NWB metadata using KG.
3. **Evaluation Agent** validates metadata against LinkML and NWB Inspector.
4. **TUIE** checks provenance and ensures correct use of AI suggestions.

## **8. Enrichment Rules**

* Strain implies species (e.g., *C57BL/6* → *Mus musculus*).
* Normalize species labels to CURIEs (NCBITaxon).
* Device model lookup to fill manufacturer.
* Normalize ages (P60, 60d, ~2mo → P60D).
* Validate session integrity (start\_time < end\_time).

## **9. Deployment Considerations**

* Run with Docker (Neo4j/GraphDB + FastAPI service).
* Version catalogs (strain→species, device→manufacturer).
* Secure write access for agents, and log provenance for every field update.
* Monitor completion rates and user acceptance of AI/KG suggestions.

## **Conclusion**

By integrating a Knowledge Graph into the NWB pipeline, metadata can be enriched, validated, and standardized. This ensures reproducibility, transparency, and scalability, enabling multi-agent collaboration while maintaining user trust.

Would you like me to also create a **diagram (flowchart/infographic)** version of this report that visually shows how agents interact with the KG in your pipeline? That way, you could paste both the text and the diagram in your Google Doc.