# **Project Report: Multi-Agent NWB Conversion and Validation Pipeline**

## **1. Introduction**

This project describes a **multi-agent pipeline** for converting heterogeneous neuroscience data into the **Neurodata Without Borders (NWB)** format. For now we are focussed on ephys data. The system integrates agents for decision-making, intermediate processes for transformation, and specialized tools/libraries for metadata standardization and quality control. External knowledge resources such as **Knowledge Graphs** and **LinkML** schemas are used to enrich metadata and ensure schema compliance.

The approach balances automation (via NeuroConv and evaluation agents) with human oversight (feedback loops), ensuring both scalability and correctness.

Architecture: <https://excalidraw.com/#json=hAte70KeTey64J-yNfaLw,lLk6pdnRMOzKiHsI4EXsww>

Codebase: <https://github.com/Python-AI-Solutions/agentic-neurodata-conversion/tree/main>

## **2. Agents**

### **2.1 Conversation Agent**

* Acts as the **entry point** for user interaction.
* Gathers information about the raw input dataset, experiment context, and missing metadata.
* Can request clarifications from the user when essential fields are incomplete.
* Ensures usability by translating technical requirements into prompts/questions understandable to non-expert users.

### **2.2 Generation Agent (Conversion Agent)**

* Responsible for **invoking the NeuroConv Tool/Library** to convert raw experimental data into NWB.
* Chooses the correct DataInterface based on input format (e.g., Open Ephys, SpikeGLX).
* Generates the NWB scaffold (NWBFile object) and populates metadata fields, marking those that are auto-generated vs. user-provided.
* Records provenance for every field, which is critical for later review.

### **2.3 Evaluation Agent**

* Validates the generated NWB file.
* Uses **NWB Inspector Tool** to check compliance with the NWB schema and best practices.
* Flags errors such as missing required fields, incorrect data types, or structural inconsistencies.
* If issues are found, sends the file back through the **Refinement Loop** with clear error messages.

### **2.4 Tool Use Instance Evaluator (TUIE)**

* Monitors whether the tool usage was **appropriate and efficient**.
* Evaluates the correctness of tool calls (e.g., was NeuroConv invoked with the right parameters?).
* Ensures the system is not only producing valid outputs but also using the correct processes to achieve them, making the most of the input data, and flagging issues independent of alignment with the schema.
* Acts as a meta-validator for the pipeline.

## **3. Intermediate Processes**

### **3.1 Data Preprocessing and Analysis**

* Standardizes raw input data into formats compatible with NeuroConv.
* Tasks include signal alignment, channel labeling, timestamp normalization, and file integrity checks.
* Reduces downstream errors by ensuring inputs are clean and structured.

### **3.2 Metadata Extraction**

* Pulls metadata from raw files (e.g., acquisition timestamps, electrode mappings, subject IDs).
* Integrates with **Knowledge Graphs** to fill gaps where possible.
* Can suggest autofills via LLMs but clearly labels these as “AI-suggested.”

### **3.3 Data Content Analysis**

* Performs semantic analysis of dataset contents.
* Example: identifying whether signals represent spikes, LFPs, or behavioral events.
* Supports enrichment by categorizing experimental conditions and mapping them to controlled vocabularies.

### **3.4 Output and Reporting**

* Produces **Conversion Logs** and **Quality Reports**.
* Summarizes what was auto-filled, what was validated, and what failed.
* Ensures transparency by giving users a clear audit trail.

### **3.5 Evaluation and Refinement**

* Feedback mechanism where failed evaluations loop back into the system.
* Human-in-the-loop involvement allows experts to override, correct, or accept autofilled values.
* Prevents silent propagation of errors and builds user trust.

## **4. Tools and Libraries**

### **4.1 NeuroConv Tool**

The practical interface for performing conversions with NeuroConv.

* Provides ready-to-use command-line and script-based workflows.
* Uses modular DataInterface classes to handle data from different acquisition systems.
* Automates repetitive conversion tasks while remaining flexible for lab-specific customization.
* Designed for researchers who want reproducible conversions without writing extensive code.

### **4.2 NeuroConv Library**

The underlying Python framework that powers the tool.

* Exposes APIs for developers to build or extend conversion functionality.
* Offers fine-grained control for advanced users to design custom workflows.
* Ensures conversions are reproducible by defining them in code rather than manual, ad hoc steps.
* Serves as the foundation for the NeuroConv Tool’s higher-level workflows.

### **4.3 Knowledge Graph and LinkML**

* **Knowledge Graph**:  
  + Stores entities (subjects, devices, sessions, tasks) and their relationships.
  + Ensures metadata consistency across experiments and studies.
  + Enables reasoning about experimental context (e.g., linking subject genotype to experimental condition).
* **LinkML (Linked Data Modeling Language)**:  
  + Provides schemas for structured metadata.
  + Ensures that metadata suggestions and autofills comply with community standards.
  + Facilitates validation against domain-specific ontologies.

### **4.4 NWB Inspector Tool**

Community-developed tool for validating NWB files beyond schema compliance.

* Uses the official NWB Validator for baseline schema checks.
* Adds best-practice validations (e.g., appropriate chunking, compression, and metadata completeness).
* Produces detailed error and warning messages, which can be fed back to the Evaluation Agent for correction.

## **5. Inputs and Outputs**

### **Inputs**

* **Raw Input Data Folder and User Input**: Contains the original lab recordings plus any manually provided metadata files.

### **Outputs**

* **Proposed NWB Output**: Preliminary file generated by NeuroConv before validation.
* **Final NWB Output Folder**: Validated NWB files ready for archival or analysis.
* **Conversion Log and Quality Report**: Summaries detailing the conversion process, metadata provenance, and validation results.
* **Human Feedback and Refinement Loop**: Captures corrections and domain expertise that improve future automated suggestions.

## **6. Benefits**

* **Standardization**: All data ends up in NWB, a widely accepted format in neuroscience.
* **Transparency**: Every field is traceable to user input, automatic extraction, or AI suggestion.
* **Error Resilience**: Validation and refinement loops prevent incorrect files from being finalized.
* **Knowledge-Enriched**: KG + LinkML integration ensures semantic consistency.
* **Scalability**: Modular agents and tools can be extended to new data formats and domains.

## **7. Challenges**

* **Metadata Completeness**: Many labs record incomplete metadata; autofills risk hallucination if not carefully validated.
* **Agent Coordination**: Multiple agents must communicate smoothly without creating bottlenecks.
* **User Trust**: Highlighting AI-suggested vs. user-provided values is essential to maintain reliability.

## **8. Conclusion**

This system leverages **NeuroConv, Knowledge Graphs, LinkML, NWB Inspector, and multi-agent orchestration** to provide a robust pipeline for neuroscience data conversion. By combining automation with human oversight, it addresses the critical challenges of metadata completeness, schema compliance, and reproducibility.

The result is a transparent, extensible, and knowledge-driven framework that can standardize diverse lab data into a common format, paving the way for more reproducible and shareable neuroscience research.