Module Interface Specification for Live Neuro

Bo Liang

April 11, 2025

1 Revision History

Date	Version	Notes
March 17	1.0	Initial Draft

2 Symbols, Abbreviations and Acronyms

See SRS Documentation at SRS

Contents

3 Introduction

This document specifies the interfaces for modules in the Live Neuro system, an interactive neural data visualization tool. It complements the SRS and MG, with full implementation details available at GitHub Repository.

4 Notation

The following table summarizes the primitive data types used by Live Neuro.

Data Type	Notation	Description
String	$\mathbb{S}$	a character string
data type dictionary	$\mathbb{D}$	a storage type for KV structures
bool	$\mathbb{B}$	Boolean data type, which can hold one of two values: either true or false
integer	$\mathbb{Z}$	a number without a fractional component in (-∞, ∞)
natural number	$\mathbb{N}$	a number without a fractional component in [1, ∞)
real	$\mathbb{R}$	any number in (-∞, ∞)

The specification of Live Neuro uses some derived data types: sequences, strings, and tuples. Sequences are lists filled with elements of the same data type. Strings are sequences of characters. Tuples contain a list of values, potentially of different types. In addition, Live Neuro uses functions, which are defined by the data types of their inputs and outputs. Local functions are described by giving their type signature followed by their specification.

5 Module Decomposition

The following table is taken directly from the Module Guide document for this project.

Level 1	**Level 2**	**Module ID**
Hardware-Hiding Module	Hardware-Hiding Module	M1
Behaviour-Hiding Module	Input Format Module	M2
	Data Processing Module	M3
	Visualization Module	M4
Software Decision Module	TRF Calculation Module	M5

Table 1: Module Hierarchy

6 MIS of Hardware-Hiding Module(M1)

6.1 Module

M1: OS Abstraction Layer

6.2 Uses

Directly interacts with OS APIs (e.g., file I/O, hardware drivers).

6.3 Syntax

6.3.1 Exported Access Programs

Name	In	Out	Exceptions
readFile	$\mathbb{S}$ (file path)	Sequence	FileNotFoundError
savePlot	plot data	file(.jpeg)	SavePlotError

6.4 Semantics

6.4.1 Environment Variables

File system, display hardware.

6.4.2 Assumptions

OS compatibility (Linux/Windows/macOS)

6.4.3 Access Routine Semantics

- readFile(): Reads neural data from disk, returns a sequence of $\mathbb{R}$ numbers.
- savePlot(): Renders visualization output to screen or file.

7 MIS of Input Format Module

7.1 Module

Multi-Format MEG/EEG Data Parser

7.2 Uses

M1 (readFile for raw data loading).

7.3 Syntax

7.3.1 Exported Constants

supported formats = [EDF, FIF, BrainVision](supported data formats).
max channels = 256(maximum allowed channels per dataset).

7.3.2 Exported Access Programs

Name	In	Out	Exceptions
load edf()	$\mathbb{S}$ (EDF file path)	$\mathbb{D}$<$\mathbb{S}$: $\mathbb{R}$[]>	EDFFileError
load fif()	$\mathbb{S}$ (FIF file path)	$\mathbb{D}$<$\mathbb{S}$: $\mathbb{R}$[]>	FIFFileError
load brain-vision()	$\mathbb{S}$ (.vhdr file path)	$\mathbb{D}$<$\mathbb{S}$: $\mathbb{R}$[]>	VHDRFileError

7.4 Semantics

7.4.1 State Variables

- currentFormat: $\mathbb{S}$(last detected data format, e.g., "FIF").
- metadataCache: $\mathbb{D}$(cached metadata from parsed files).

7.4.2 Environment Variables

7.4.3 Access Routine Semantics

- load edf():

  Output:

  data: Time-series array of shape[channels × samples].

  Exception:

  EDFHeaderError: Invalid EDF header structure.

  Implementation: Uses eelbrain for EDF parsing.

- load fif():

  Output:

  data: MEG/EEG sensor data.

  Exception:

  FIFFileError: FIF file version mismatch or missing data tags.

  Implementation: Relies on mne-python library.

- load brainvision():

  Output:

  data: EEG data segmented by markers

  Exception:

  VHDRParseError: Inconsistent header fields in.vhdr file.

7.4.4 Local Functions

- parse edf header(): Extracts EDF header fields and validates integrity.

- align brainvision files(): Synchronizes.vhdr,.vmrk, and.eeg data.

8 MIS of Data Processing Module

8.1 Module

Statistical Preprocessing and Validation
Primary Function: Validates input data integrity and applies statistical preprocessing to neural signals.

8.2 Uses

- M2(Input Format Module): Receives parsed raw data (e.g., MEG/EEG time-series).
- M5(TRF Calculation Module): Provides preprocessed data for dipole current computation.

8.3 Syntax

8.3.1 Exported Access Programs

Name	In	Out	Exceptions
validate input	$\mathbb{D}$<$\mathbb{S}$: $\mathbb{R}$[]>	$\mathbb{B}$	InvalidDataError
compute statistics	$\mathbb{R}$[]	$\mathbb{D}$<$\mathbb{S}$: $\mathbb{R}$>	NaNError

8.4 Semantics

8.4.1 State Variables

- validatedSignals: $\mathbb{D}$<$\mathbb{S}$: $\mathbb{R}$[]>(cached validated data).

- baselineStats: $\mathbb{D}$<$\mathbb{S}$: $\mathbb{R}$>(mean and std of reference signals).

8.4.2 Access Routine Semantics

- validate input(raw data: $\mathbb{D}$):

  Validation Steps:

  Check if raw data[data] is of type $\mathbb{R}$[] and non-empty.

  Ensure no NaN or infinite values in the signal.

  Output:

  Returns True if validation passes.

  Exceptions:

  InvalidDataError: Non-numeric values or mismatched channel counts.

- compute statistics($\mathbb{R}$[]):

  Output:

  mean, std, min, max and other statistical information

  Exceptions:

  NaNError if signal contains invalid values after preprocessing

9 MIS of Visualization Module

9.1 Module

Interactive Neuro-imaging Module

9.2 Uses

M1 (writePlot), M3 (processed data), M5 (TRF results)

9.3 Syntax

9.3.1 Exported Access Programs

Name	In	Out	Exceptions
plot	List<$\mathbb{R}$[]> (dipole currents), List<$\mathbb{S}$>(plot type), $\mathbb{S}$(interaction type)	void	Plotting Error

9.4 Semantics

9.4.1 State Variables

- activePlots: 𝔻<PlotID, PlotData> (metadata for open plots).
- linkedViews: Set<PlotID>(plots synchronized via linkPlots).

9.4.2 Environment Variables

GPU acceleration (enabled via plotly.graph_objects and nilearn.plotting)

9.4.3 Access Routine: Semantics

Syntax:

- `plot(data: List<R[]>, plotTypes: List<String>, interactionMode: String)`

Valid `plotTypes` values (case-insensitive):

- `"time_series"` – Renders time vs amplitude plots for each channel.
- `"glass_brain"` – Displays brain activity using anatomical projections (e.g., with `nilearn`).
- `"topomap"` – Sensor-space topographic activation maps.

Valid `interactionMode` values:

- `"zoom"` – Enables scroll and drag-based zooming.
- `"drag"` – Enable click and drag plots
- `"click"` – Enable click in data plots

9.4.4 Local Functions

- _updateLinkedAxes(): Propagates axis changes to all linked plots.

10 Appendix

10.1 Dependencies

- Plotly: Used for interactive HTML5 visualizations (zooming, panning, tooltips).
- Nilearn: Handles neuroimaging-specific rendering (glass brain plots).

10.2 Performance Notes

- GPU acceleration is optional but recommended for real-time interaction with large datasets (>10 samples).