ARCHITECTURE

The Pipeline: 8 Composable Stages

Use only what you need. Each stage streams via stdin/stdout for Unix-style chaining.

#	STAGE	PURPOSE	CLI	STATUS
1	Import	Search & fetch from HTTP/S, S3, FTP, REST API	`zyra acquire`	Implemented
2	Process	Decode, subset, convert (GRIB2, NetCDF, GeoTIFF)	`zyra process`	Implemented
3	Simulate	Generate synthetic/test data	–	*Planned*
4	Decide	Parameter optimization and selection	–	*Planned*
5	Visualize	Static maps, plots, animations, interactive	`zyra visualize`	Implemented
6	Narrate	AI-driven captions, summaries, reports	`zyra narrate`	Implemented
7	Verify	Quality checks and metadata validation	`zyra verify`	Partial
8	Export	Push to S3, FTP, Vimeo, local, HTTP POST	`zyra export`	Implemented

UNIX-STYLE STREAMING

```
zyra acquire http $URL -o - | \
zyra process convert-format - netcdf --stdout |
zyra visualize heatmap --input - --var TMP -o plot.png
```

Stages are composable — pipe any stage's output directly into the next. Every stage supports `.../...` for seamless chaining.

CAPABILITIES

Key Features

- **Scientific formats:** GRIB2, NetCDF, GeoTIFF with xarray, cfgrib, rasterio
- **Connectors:** HTTP/S, S3, FTP, REST API, Vimeo
- **Visualization:** Heatmaps, contours, vectors, particles, animations, interactive maps (Folium, Plotly)
- **Agentic orchestration:** Natural language intent → `zyra plan` generates an execution DAG; `zyra swarm` dispatches stage agents in parallel with provenance tracking
- **Narration swarm:** Multi-agent LLM chain (context → summary → critic → editor) generates validated scientific narrative; LLM-agnostic via `--provider`
- **Provenance:** SQLite-based event logging for full reproducibility
- **MCP server:** Exposes Zyra's full pipeline as Model Context Protocol tools — letting AI assistants like **Claude** or **ChatGPT** acquire data, run transforms, and generate visualizations on degC↺□□□