Skill Import Industrial Ontology

AI skill: import an RDF/OWL ontology file into the FrameworX Unified Namespace using the `import_graph_model` MCP tool. Ships with policy starters for ISA-88 and IOF.

Version 10.1.5+

When to invoke

- “Here's an IOF biopharma ontology. Build me the UDTs and tags.”
- “Import this RDF/JSON ontology file I got from the modelling team.”
- “Materialize the ISA-88 master recipe I downloaded as RDF/JSON.”

Preconditions

1. Solution open in Designer (check via `get_state`).
2. User has `AllowEditTags` permission.
3. Source file is RDF/JSON (`.rj`) reachable from Designer, OR an inline RDF/JSON string. Other serializations (JSON-LD, Turtle, RDF/XML, N-Triples on disk) are on the roadmap — use `.rj` for reliable import in 10.1.5. If the user has only a SPARQL endpoint, fetch the graph via WebData first and pass the RDF/JSON response as inline content.

Steps

1. Confirm the solution is the right target. If not, offer to `create_solution` or `open_solution`.
2. Pick a policy: `iof` for IOF ontologies, `isa88` for batch hierarchies, `custom` for unknown schemas (or omit the parameter for defaults). Ask the user if unsure.
3. Run **dry-run first**:

```
import_graph_model(source="file", file_path="C:/.../file.rj", policy="iof", dry_run=true)
```

4. Summarize the dry-run to the user: class count, instance count, `warnings[]`. Flag cycles and multi-parent cases.
5. On user approval, re-run with `dry_run=false`.
6. Call `generate_uns_visual(scope="full")` to verify the imported structure visually.

Gotchas to surface

- **Do NOT fabricate class or policy names.** If the user describes an ontology without providing the file, ask for the path or the triples. Fabricated policy files are the #1 misuse mode.
- **File format today is `.rj` (RDF/JSON).** If the user has Turtle, JSON-LD, RDF/XML, or N-Triples, ask them to convert to RDF/JSON first (many graph tools offer this export). Non-`.rj` formats are on the roadmap; today the import will fail or silently fall back.
- **`/Attr` is the STATIC half only.** Typed OWL individuals representing real equipment (mixers, motors, valves) become `/Attr` tags by default. This captures the STATIC ontology metadata only — `/Attr` members carry literals from the RDF, populated as `StartValue`. If the imported NamedIndividual is intended as live equipment, ALSO author a naked UserType-instance tag at `<path>/<name>` (no `/Attr` suffix) via `write_objects` after the import. The `/Attr` envelope and the naked instance coexist as siblings; the naked instance carries dynamic members fed by Devices, Scripts, or simulators. See Industrial Ontology Integration How-to Two paradigms section.
- **Re-import collision**. Default is `skip` (user edits win). Ask before switching to `overwrite`.
- **Runtime `Attributes` do not round-trip** automatically. Run `promote_retentive_attributes` first if the user expects them to.

See also

- Import an OWL/RDF ontology into your UNS. Human walkthrough for the Designer wizard path.
- Skill Export UNS as RDF. Reverse direction.
- Skill Generate UNS Visual. Verify the result.
- Industrial Ontology Integration How-to. Full standards map, policy templates, and architecture alignment.