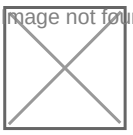
Project Organization

Data models must employ certain organizational requirements in XINA to ensure they are interpretted correctly by struct API calls and front end tools. These apply to both structures within model groups, as well as the organization of model groups themselves.

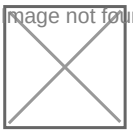
Projects / Categories

A **project** should be defined by a single XINA group at the top level. Each model is then defined by a single XINA group, which contain all groups and databases associated exclusively with the model. These should either be defined in the project group, or may be subdivided into **category** groups.



A project may use a mix of both approaches or additional levels of subcategories if required, but it is recommended to either use a flat structure or single level of category groups to avoid confusion. Models may be referred to by the path relative to their project group (in the above example, model_a would be referenced as `model_a` or `category_a.model_a`, respectively).

Project and category groups may also include additional groups and databases of data or resources which are not model specific, such as journals or definitions databases. In most cases with standard structures, models will default to databases or groups within the model, but search for them up the tree if not found. A complete project group might look like:



Project Configuration

A group is defined as a project by the `xs_struct_project` key. The value is a JSON object with the following definition:

Key	Value	Default
`def_mn`	relative path to mnemonic definitions database	`def.mn`
`def_prof`	path to profile definitions database	`def.prof`
`def_plot`	path to plot definitions database	`def.plot`

A group is defined as a category by the `xs_struct_category` key. The value is a JSON object extending the definition of the `xs_struct_project` key, automatically inheriting any unset values from the project configuration.

All models are required to provide an `mn_def`, `prof_def`, and `plot_def` database. It is **strongly recommended** that these be shared by the entire project, and that all models use the same temporal precision, to maximize

intercompatibility between models. Sharing definitions databases does not preclude identifying particular definitions as relevant only to specific models.

Model Organization

Data within a model falls into four primary classifications:

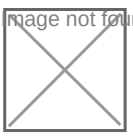
- **Telemetry**
  - source data file(s) from data collection point
  - typically stored in a raw (sometimes binary) format
  - storage cost is cheap
  - accessing data means downloading files or most likely requires custom XINA tools
  - may be divided into multiple **data sources** (see below)
- **Viewable Data**
  - extracted from telemetry into XINA database(s)
  - telemetry is the single source of truth for this data, not intended to be user editable
    - (except under controlled circumstances with struct API calls)
  - data is either **mnemonic**, **instant**, or **interval** (see below)
  - can be accessed and analyzed with built-in XINA tools
  - storage is expensive
  - optimizations may be needing depending on project requirements, data volumes
- **User Metadata**
  - additional data added by users, often directly through the XINA interface
  - XINA likely the primary repoository for this data
  - for example, a journal
- **Definitions / References**
  - may be user entered or defined outside XINA
  - may exist at model level or above (category/project level)
  - more formal and restricted than user metadata

Model Configuration

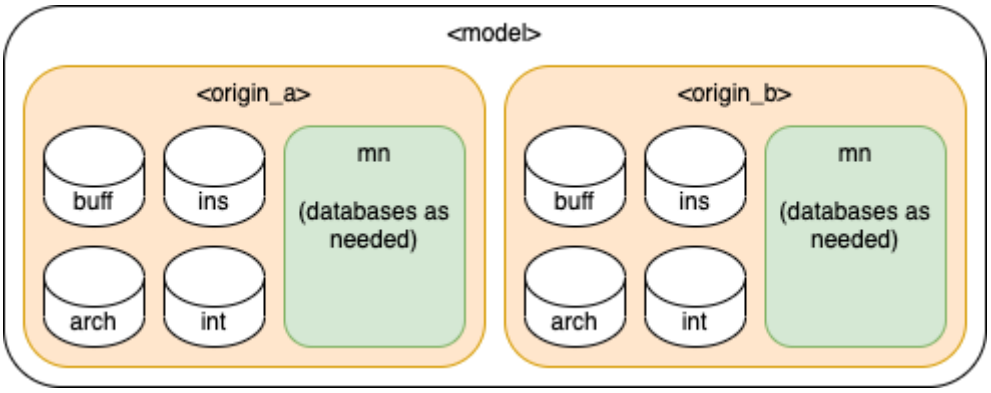
A group is recognized as a model if the `xs_struct_model` key is set in the group objects. The value is a JSON object extending the definition of the `xs_struct_project` key, automatically inheriting any unset values from the parent project or category configuration.

Origin

Abstractly, a **data origin** (or simply **origin**) is a single point of data import to a model. In many cases, a model will only have a single data origin; for example, if all data is provided directly from a single instrument, or multiple components are merged into a single data stream through FEDS before import into XINA. In these cases delineation by origin is not required in model organzation, and should use this pattern:



However, in environments with multiple import points running in parallel, databases must be designed with multiple origins.



In this example each source file would need to specify either `origin_a` or `origin_b`. Additionally, each origin has distinct databases for instant, interval, and mnemonic data. This would be required if each data source provided all three data types. As requirements for instants and intervals are less stringent than mnemonics, in some circumstances instants and intervals could be considered a single source and populated independently:

Image not found or type unknown

Revision #11
Created 28 July 2022 15:51:05 by Nick Dobson
Updated 17 July 2023 14:09:08 by Nick Dobson