DM/DAL/APP Joint Session:

Introductory remarks

Mark Cresitello-Dittmar

May 23, 2024

Statement of Purpose

Why are we here?

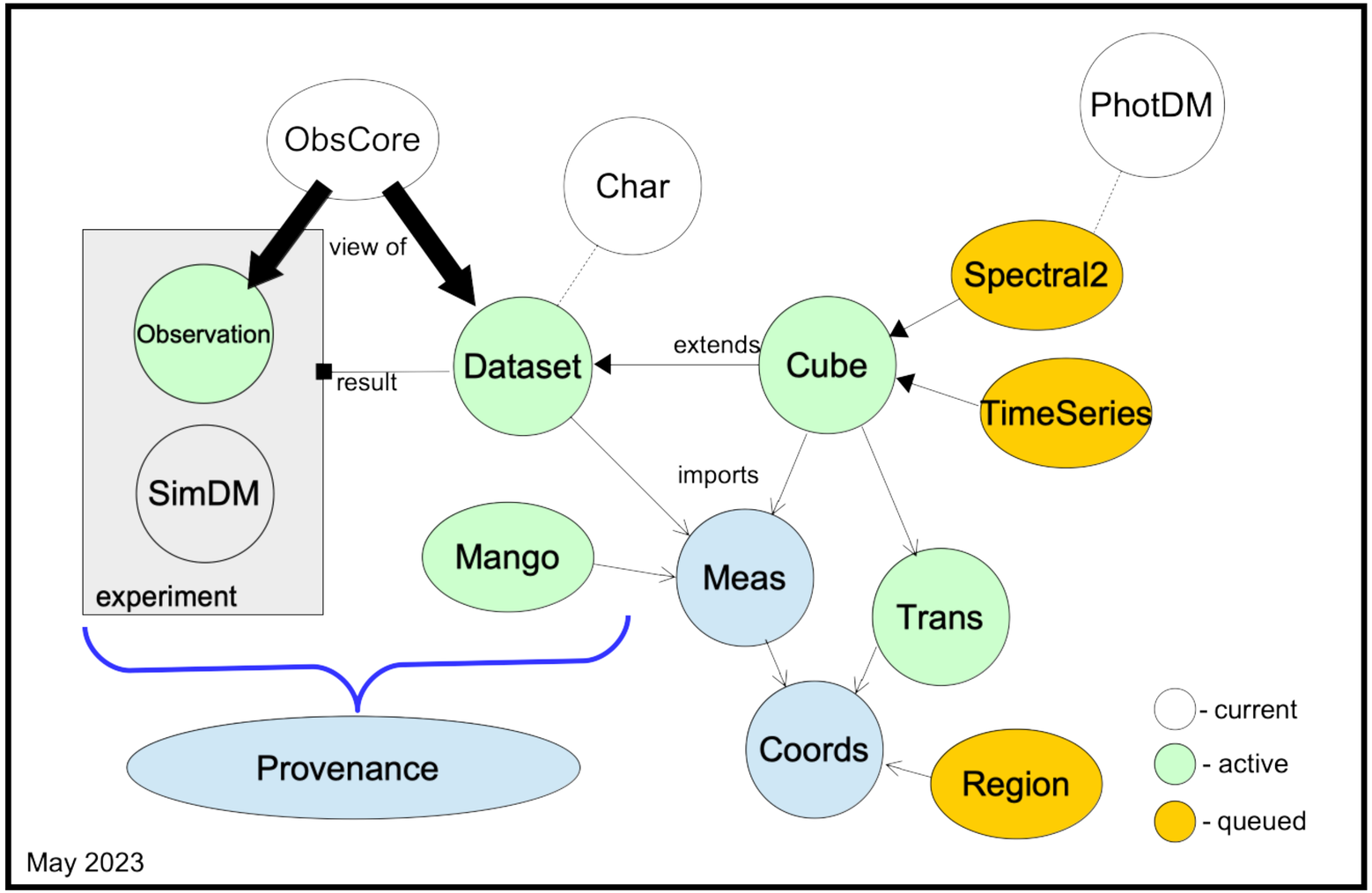
- To open a discussion about the PROCESS by which the IVOA data models serve their users (primarily DAL and APPS).
  - Satisfying the most sophisticated threads (eg: representing data products and catalogs), requires detailed and flexible data models.
  - However, MANY usage threads have much lighter requirements and those users want to deal with objects at the same level.
  - This is increasingly leading to models which describe the same concept domain at different levels. These often have inconsistencies since they are only tenuously related to their parent models.
  - What strategies do we want to employ to enable these usage threads, but ensure model consistency and make it clear to users which models apply to their needs?

Motivation

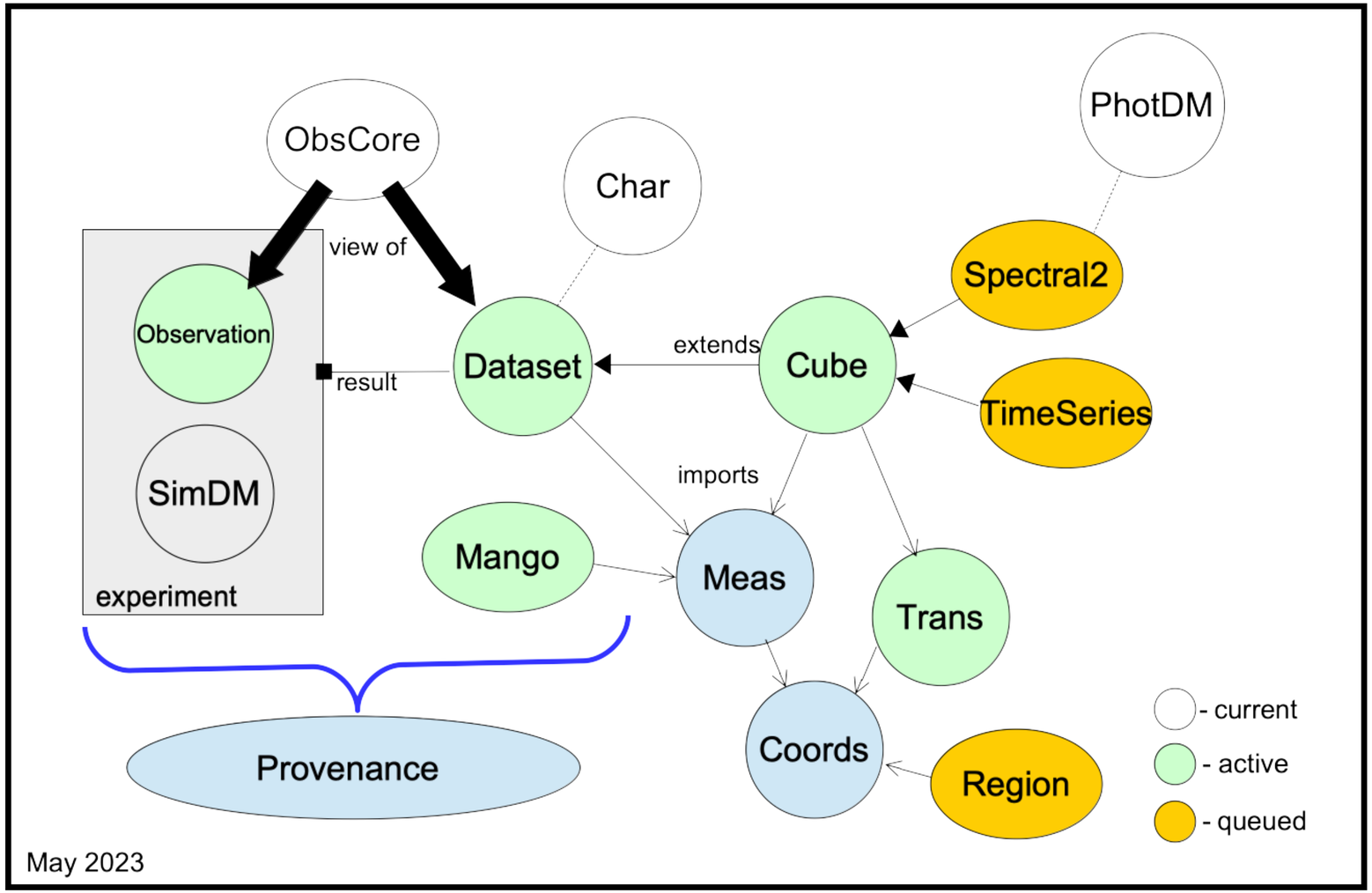
- Inspired by several discussions and talks at the Bologna interop and since.
  - S. Hughes talk including “Artifacts generated from the Model”
- Recent activities around the Epic Propagation thread, and the various solutions to this very question which it spawned.
  - This really highlights the need for an official strategy to employ.
- Ongoing work with ‘One-step Provenance’ to define a representation of Provenance suitable for usage within a dataset.
- Interest in reconciling the Shape concept which exists in STC-1.33 (Region), DALI-1.1 (base types), FOV-0.0 (Shape)
- The interest in folding CAOM into the IVOA Data Model landscape
  - This model has conceptual overlap with several models at a level somewhere between data products and data discovery.

Setting The Stage

Model Ecosystem diagram



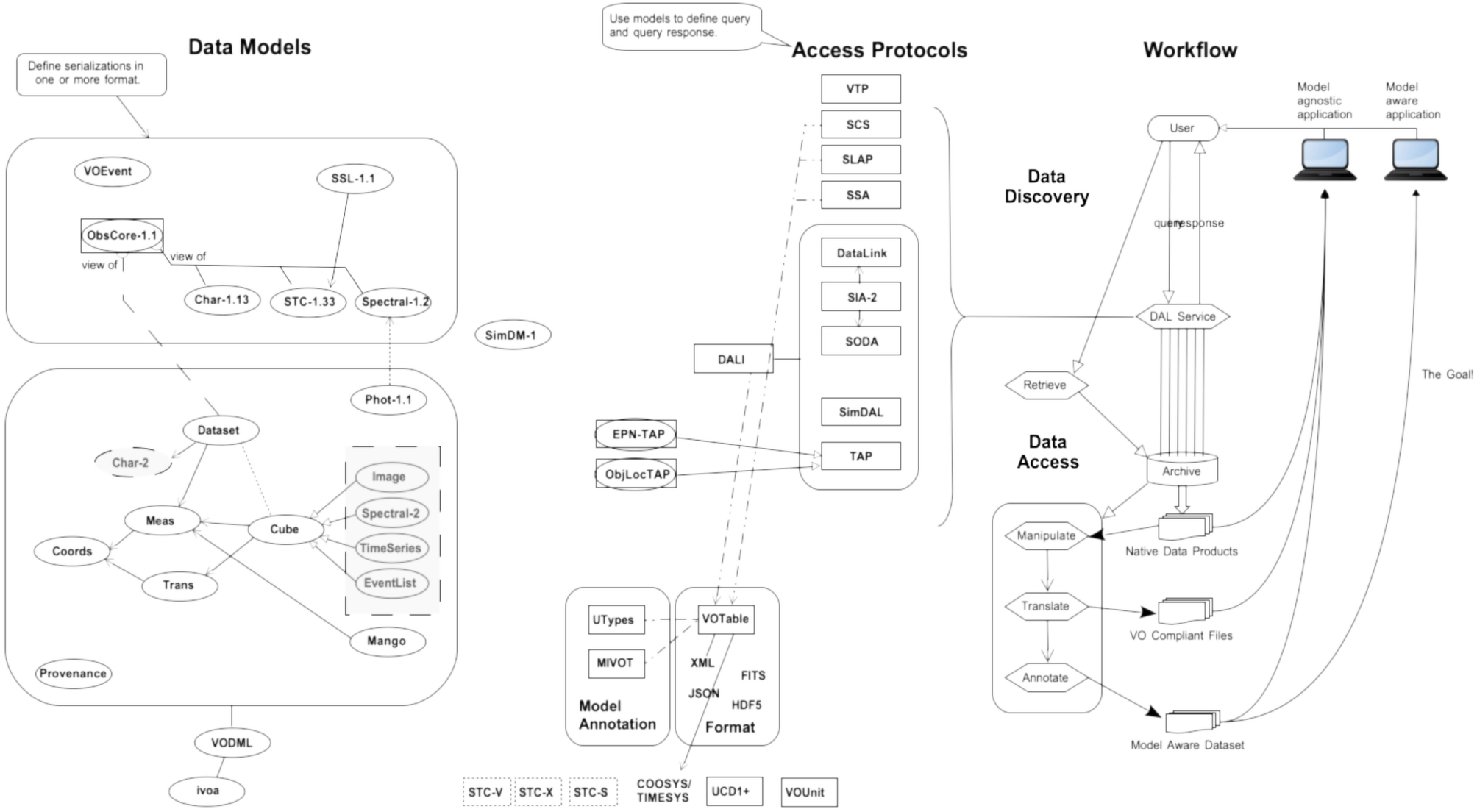
Model Ecosystem diagram



Missing Context

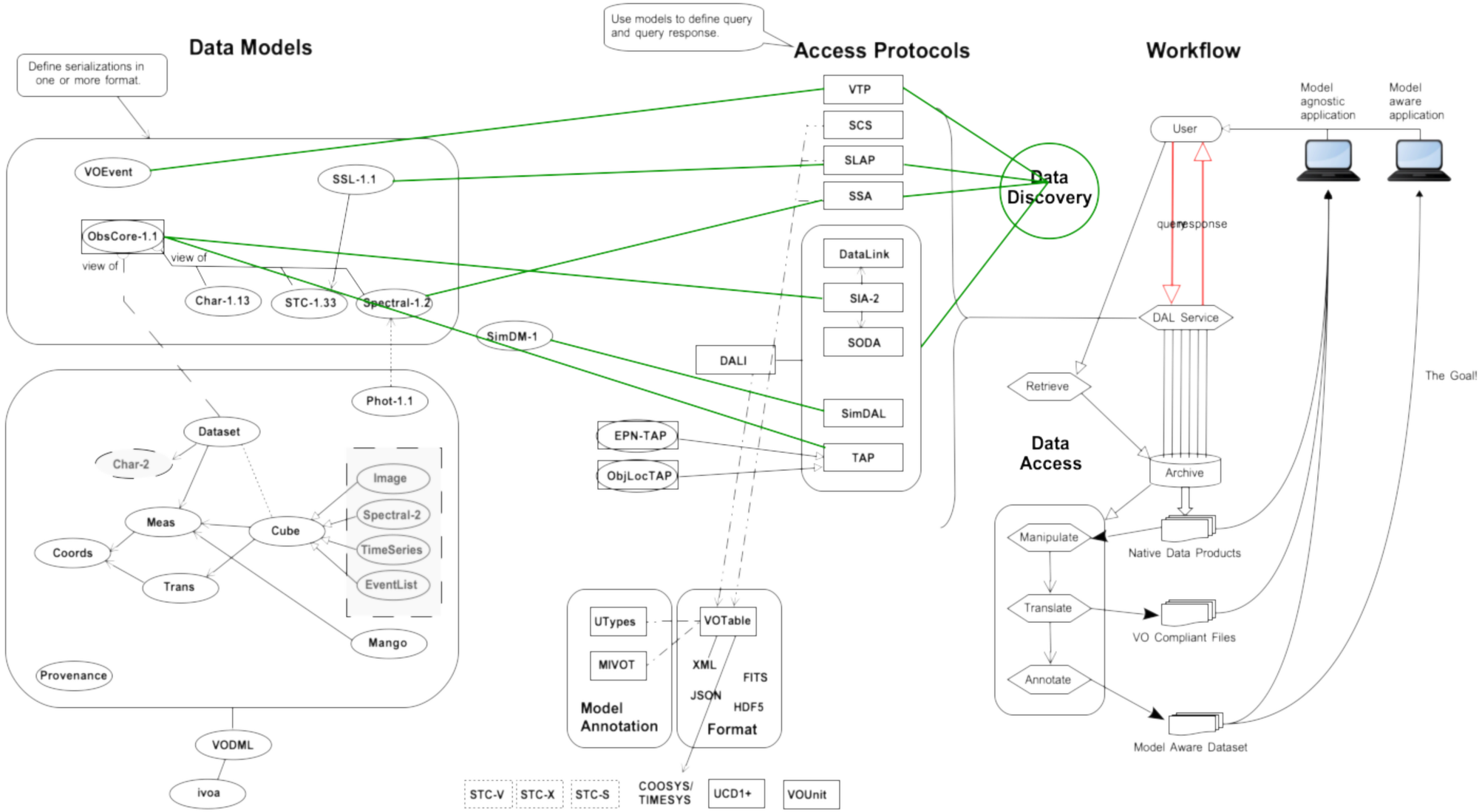
How/When are these models used?

Model Ecosystem: In Context



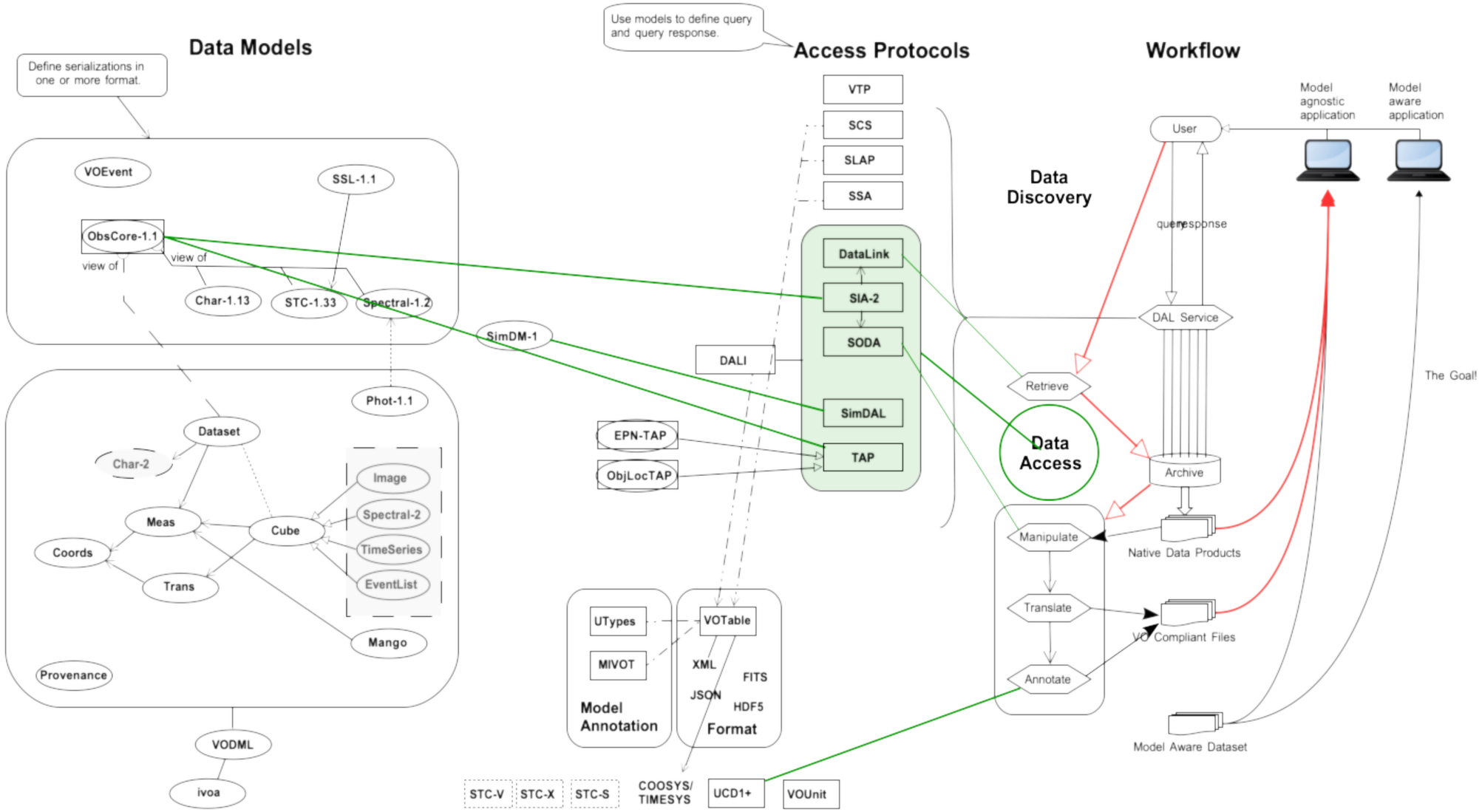
Serialization Standards

Model Ecosystem: Data Discovery Support



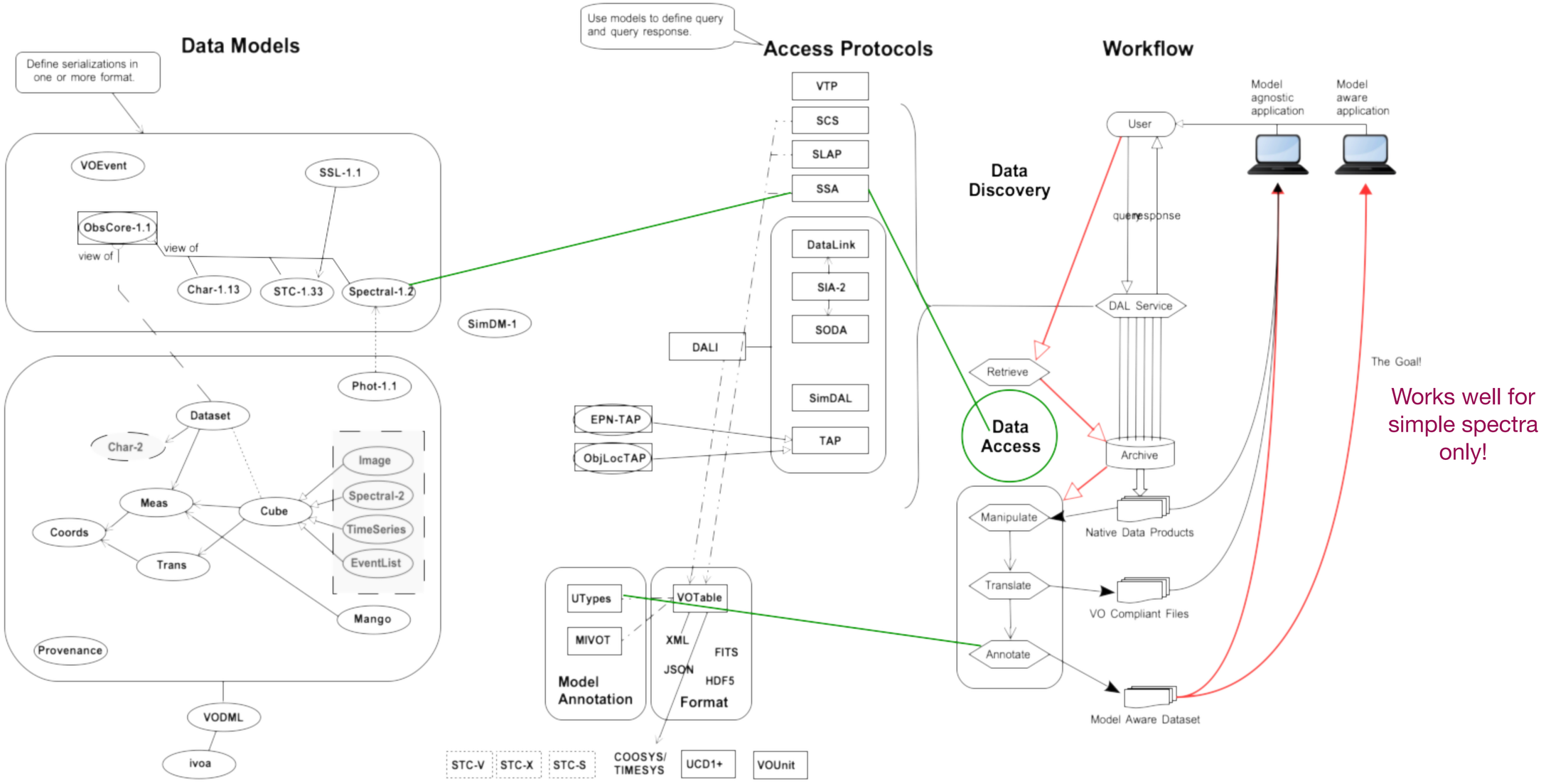
Serialization Standards

Model Ecosystem: Data Access Support

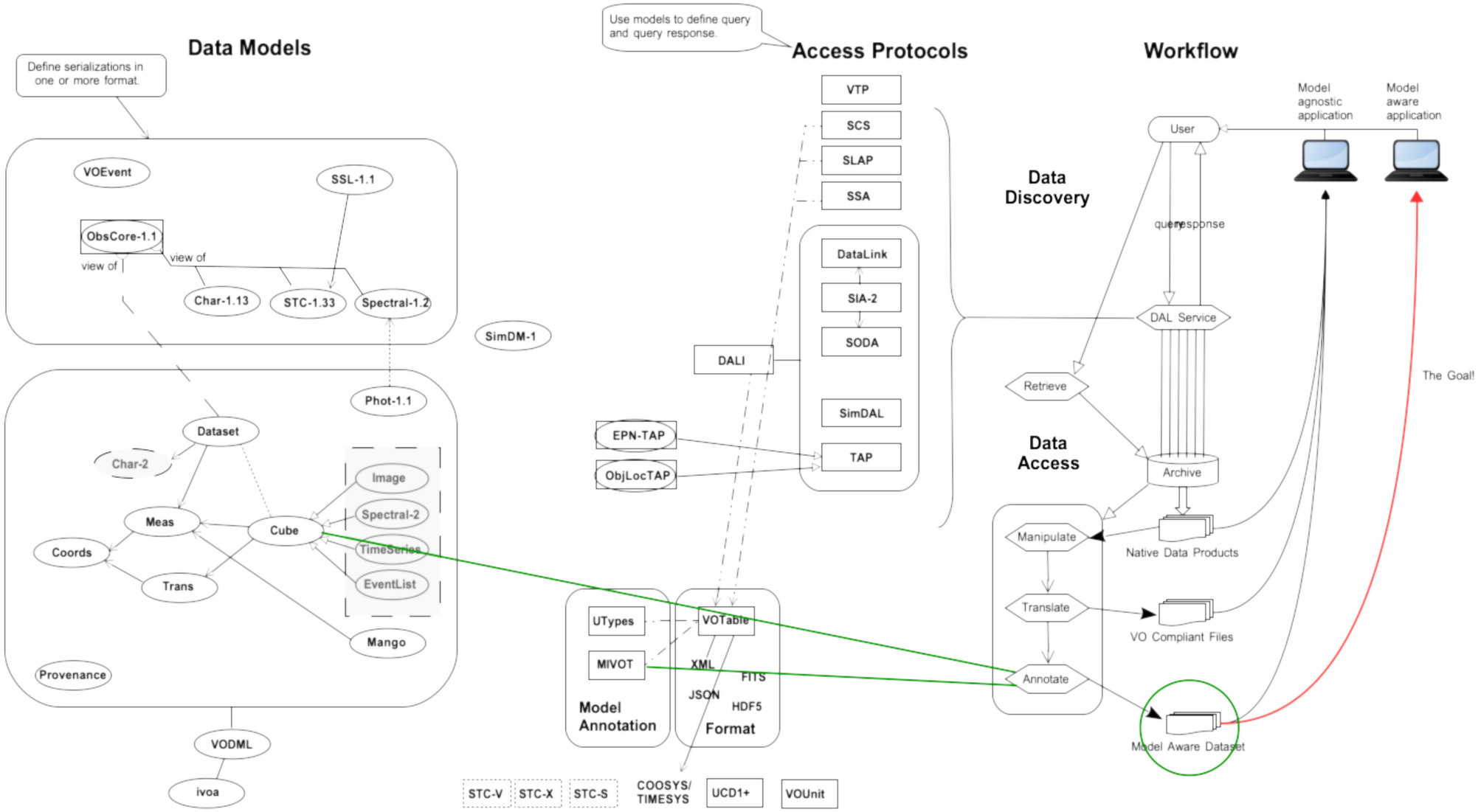


Serialization Standards

Model Ecosystem: Data Access - Spectrum

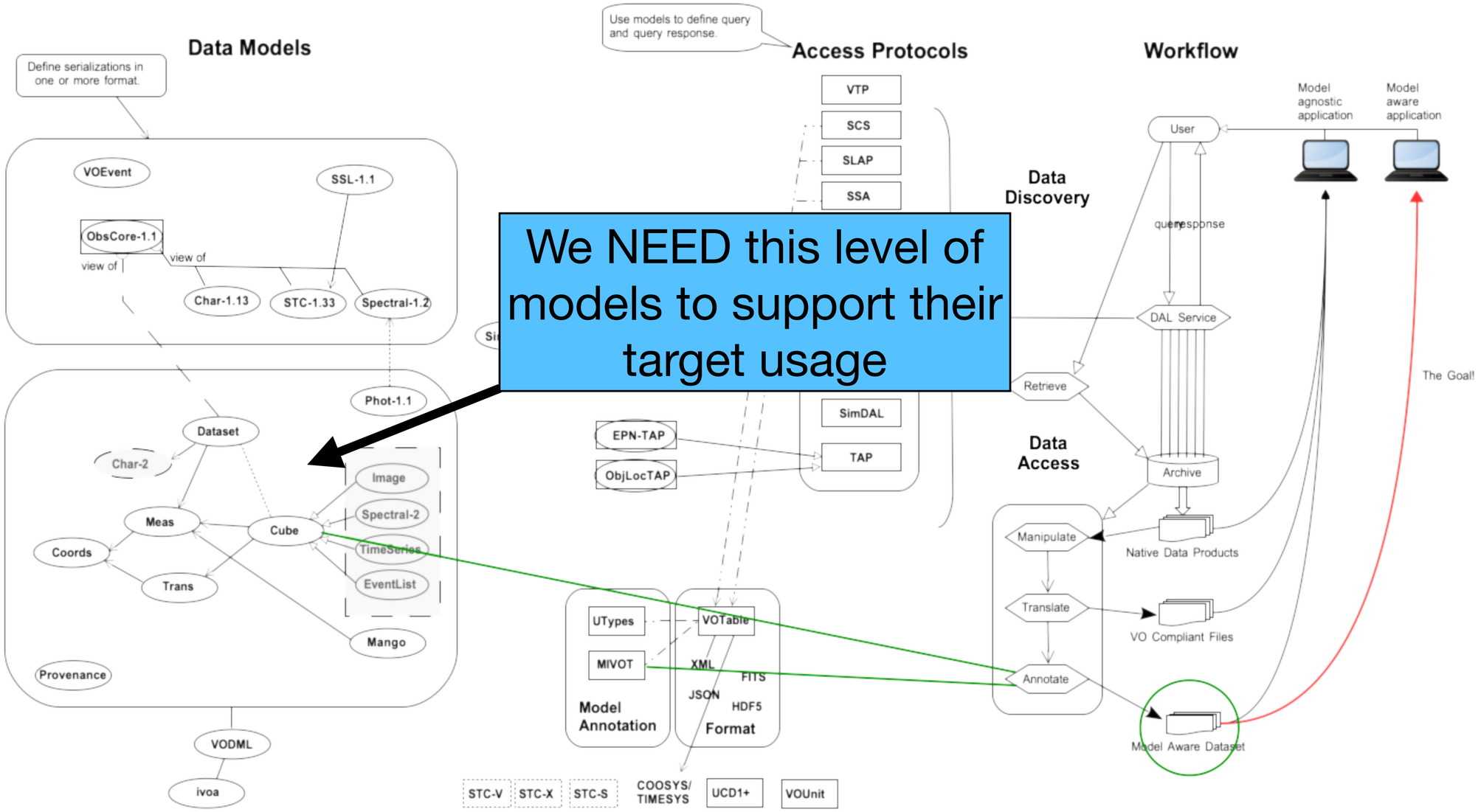


Model Ecosystem: Data Product - Cube



Serialization Standards

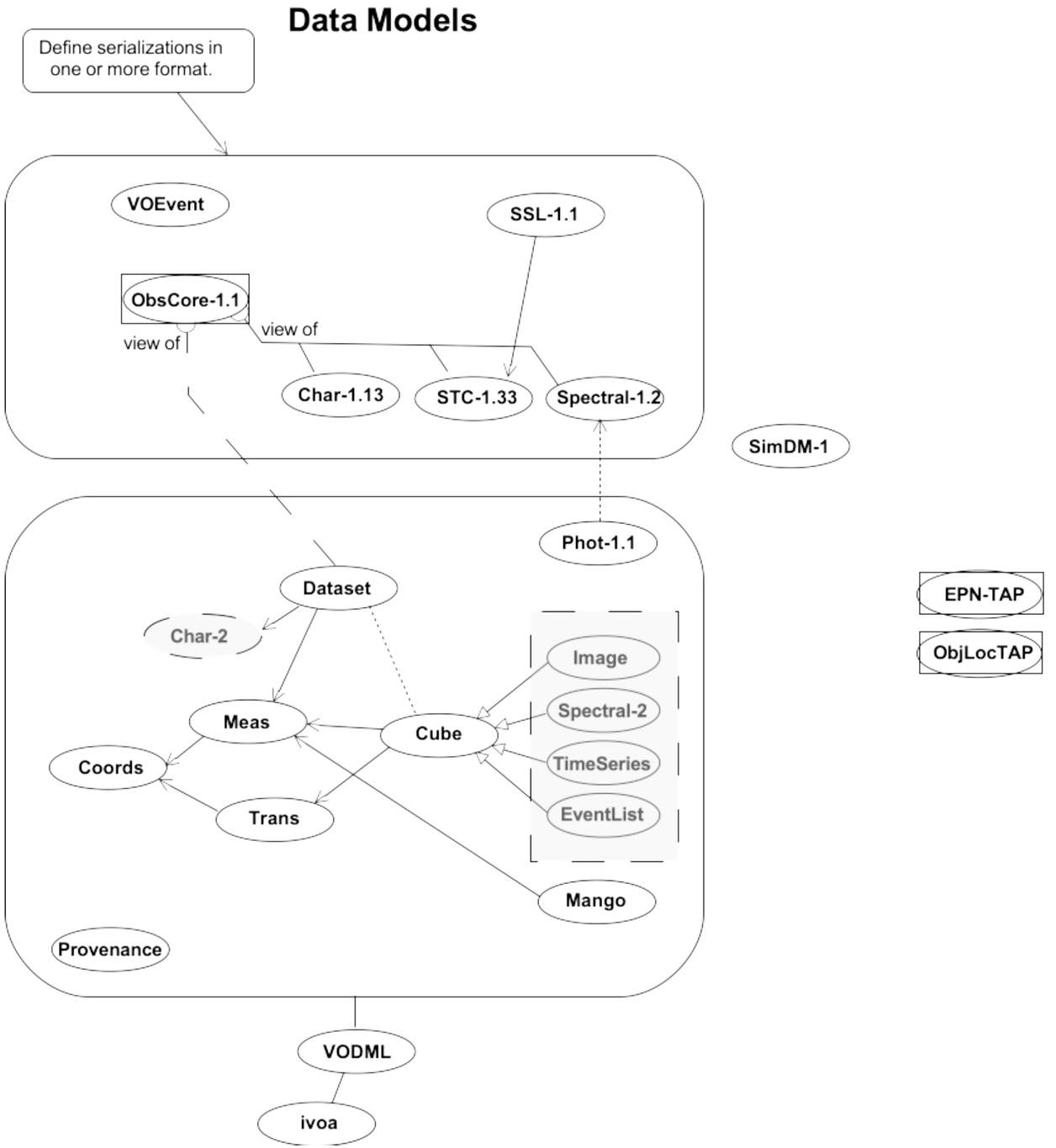
Model Ecosystem: Data Product - Cube



Getting to the point

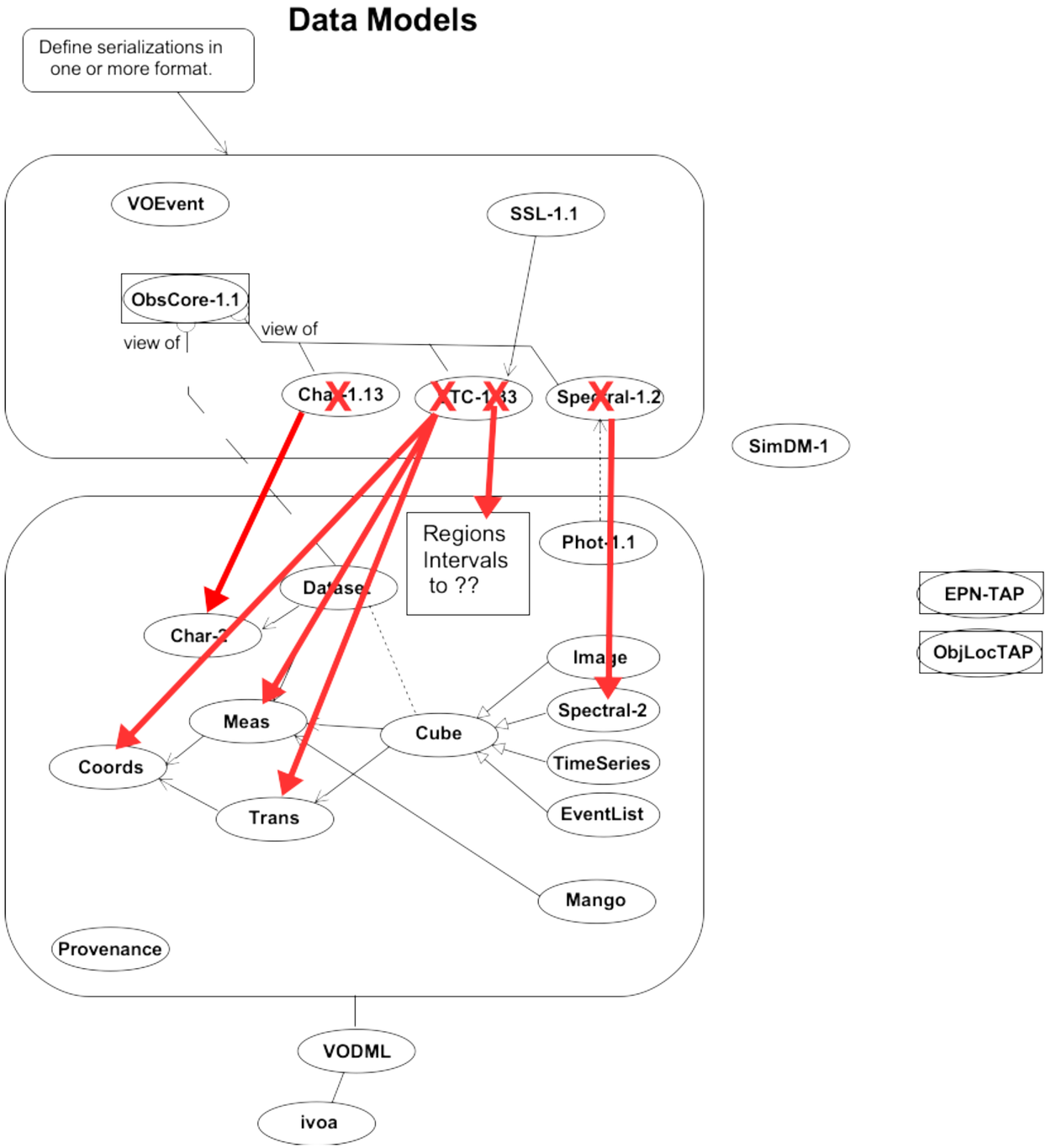
Models:

Next Phase



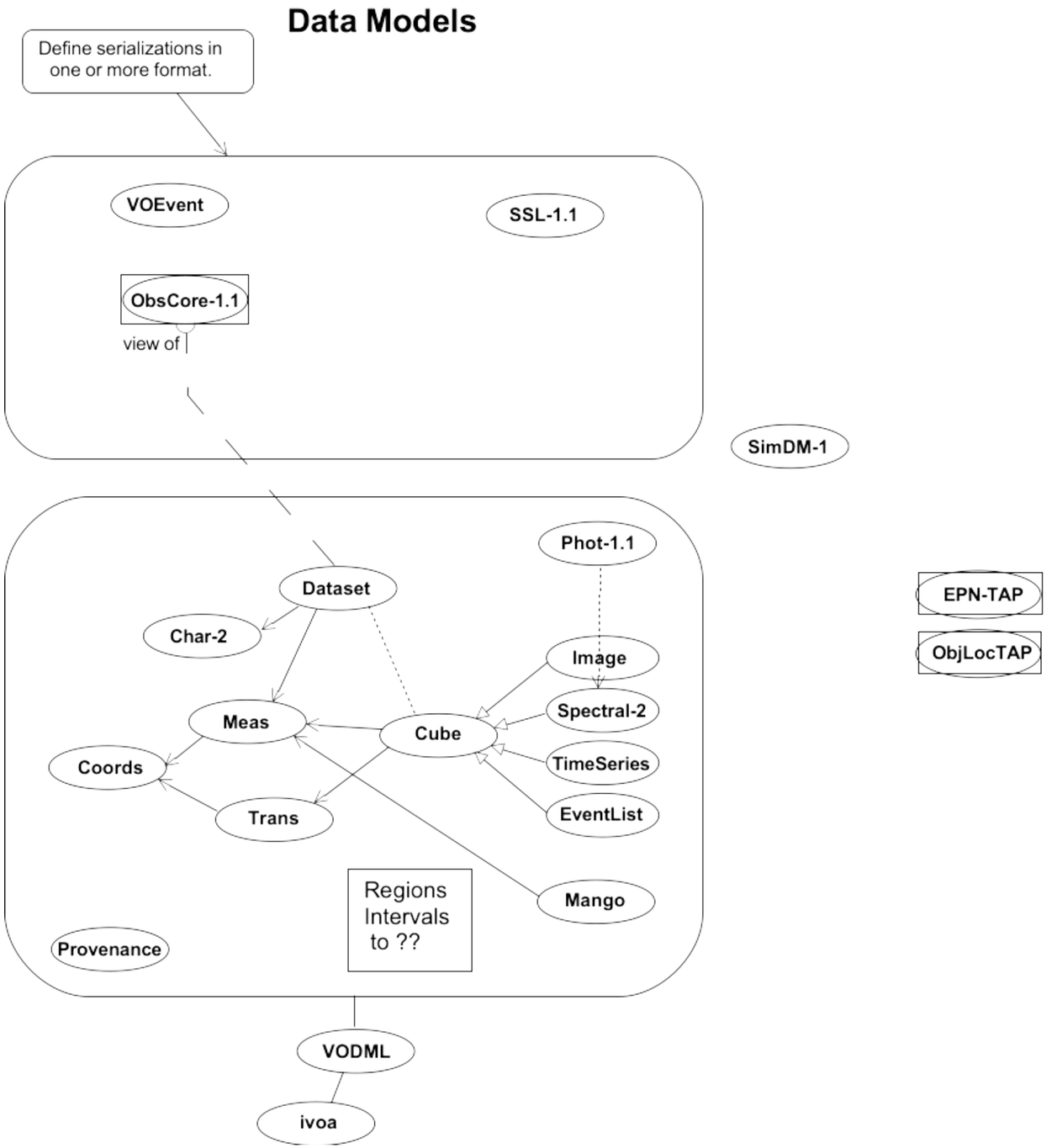
Models:

Next Phase



Models:

Next Phase



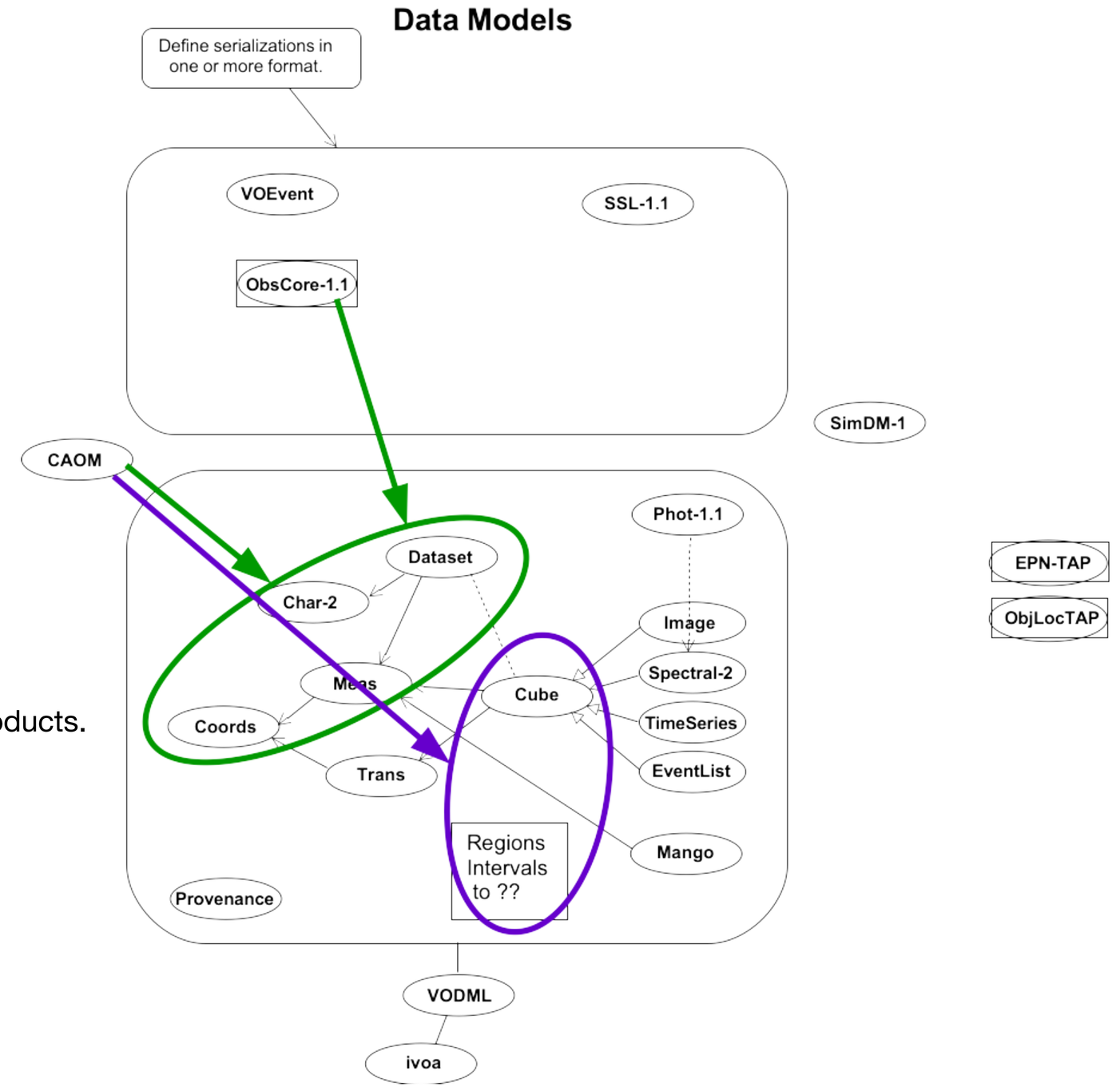
Models:

In comes CAOM

CAOM serves Observation models and is the original model basis for ObsCore.

CAOM content extends the concept overlap to include the Cube and Regions.

CAOM is NOT capable of representing Data Products.



Models:

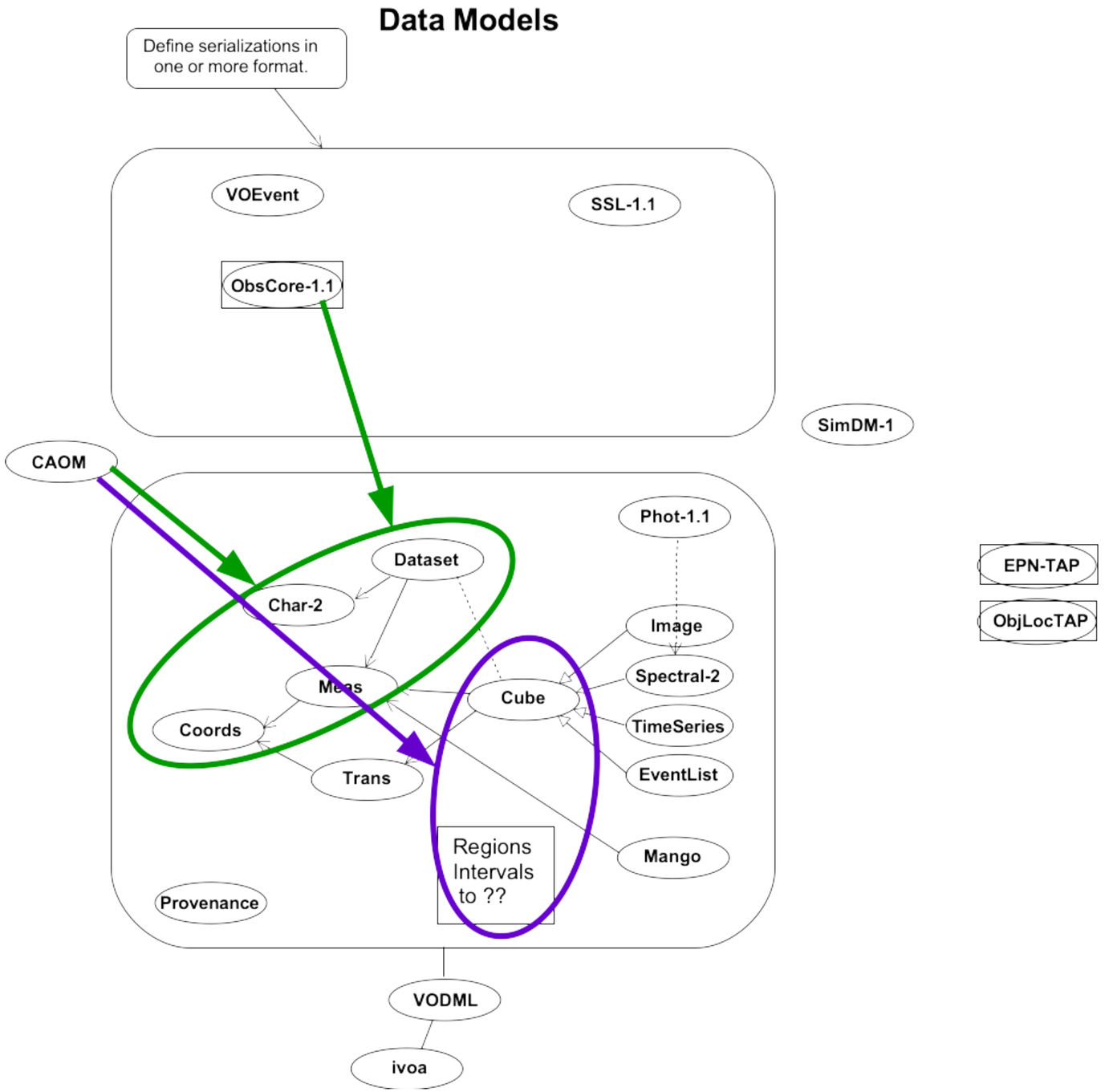
In comes CAOM

We would have 3 Layers of data models covering the same concepts, at different levels of detail to serve their respective clients:

* ObsCore ==> Data Discovery
* CAOM ==> Archives
* Cube family => Data Products

We need a strategy to:

* Make it clear who the intended client is for each model
* Relate the ‘view’ models to the base models. This is important for interoperability



And its not just CAOM

Application thread

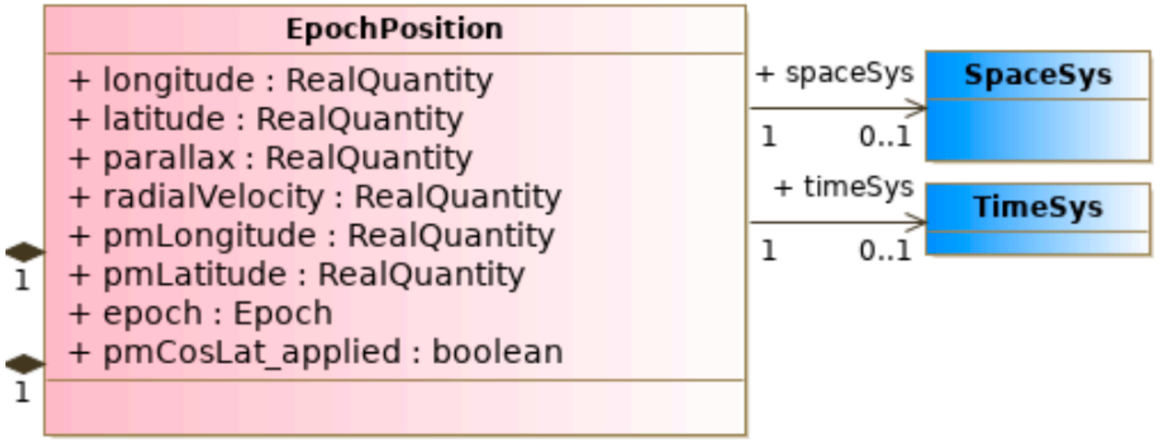
Epic Propagation

The base model representation (using Meas. and Coords. models) was considered too detailed for this usage thread.

A ‘local’ model was created which meets the needs of the usage thread.

This is a reasonable solution, BUT….

* The object is VERY bound to the details of the usage thread. For example, it assumes all values are provided together, and in the same coordinate system. This was NOT the case with the thread implemented at the DM Workshop 2020.



This situation will happen a LOT. If this strategy is applied, it raises the questions:

* Where does this object live? It should be close to the client.

* How is it related back to the base models?

Provenance

Recording Provenance in Datasets

Extension of voprov Python package

- Thanks to Benjamin Parciany (internship)
- **User friendly functions** :
  add_used entity()
  add_activity_description()
  add_agent()
  …
- **Dedicated add_one_step() function**
  builds the subgraph
  from a dictionary

```
onestep_1 = {
    "product_id": "obs:image1b",
    "product_role": "bias-subtracted-image",
    "contact_id": "mservillat",
    "step_id": "ps:2459",
    "step_name": "ps:bias_subtraction",
    "step_parameters": {"threshold": 10, "skip
    "step_description": "remove the readout no
    "step_software": ["gammapy_v1.2"],
    "used_ids": ["obs:image1", "obs:bias"],
    "generated_ids": [],
    "process_id": "7531",
    "workflow_name": "ImageCalibration",
    "instrument_id": "CTAO",
}
```

```
pdoc.add_one_step(onestep_1)
```

```
<VOProvDataSetEntity: obs:image1b>
```

* The full Provenance model is too abstract for inclusion in Data Products.
* A simpler model with the ‘minimum’ required information was derived.
* And implemented to record this content in the data product headers.

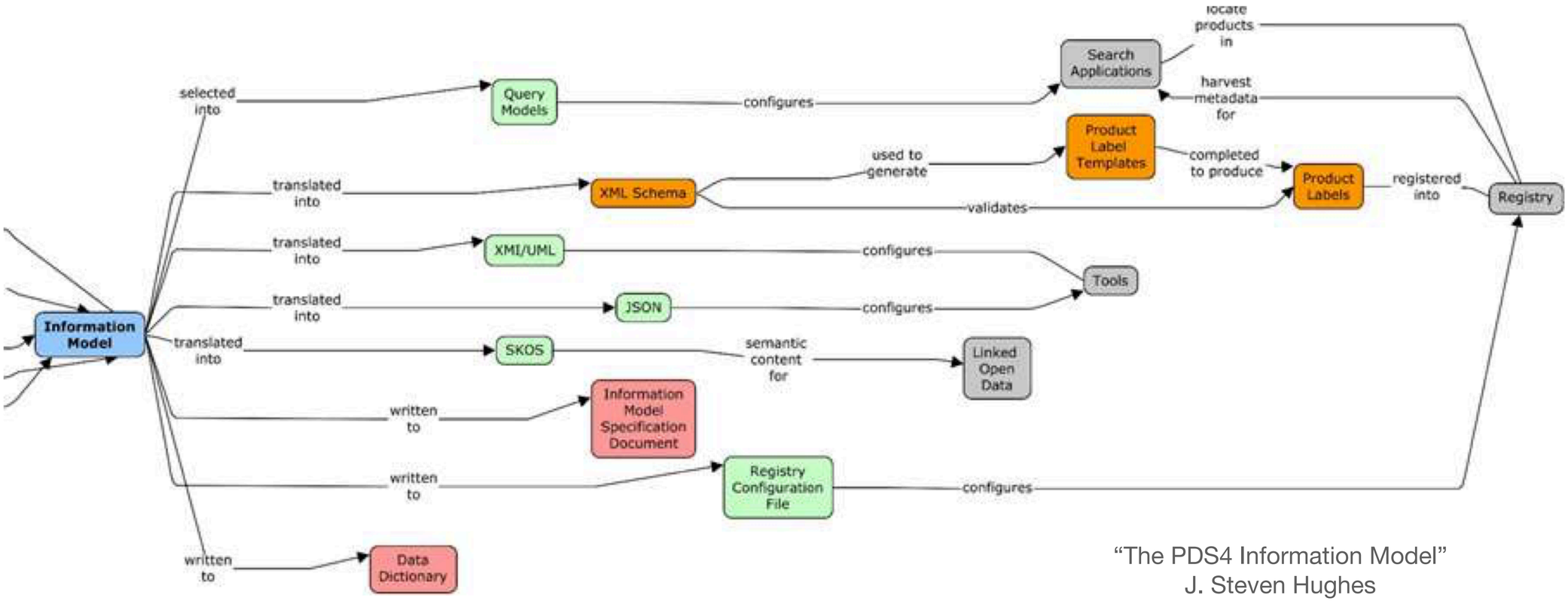
Finding a solution

- DAL and Apps are the primary clients for the Data Models.
  - DAL: to define query and response
  - APPS: accessing model information from serializations (ucd, UType, vodml)
- There are multiple strategies which could be applied …
  - Local models specialized for a particular usage
  - Serialization specs which enable flattening where possible
  - Can VODML facilitate ‘view’ model specification
- We need to work together to find a strategy which works for us all.

National Aeronautics and Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

Artifacts Generated from the Model



"The PDS4 Information Model"
J. Steven Hughes
IVOA Interop; May 2023

Discussion Questions

- How do we deal with the apparent need for models covering the same domain space at different levels (views)?
  - If so, at what granularity?
  - Can we derive them from the base models?
  - Do we need to explicitly relate them to the ‘base’ models? If so, how?
- DAL protocols are typically associated with Data Models to define the Query, Response and Results. Some include the local model along with the protocol (EPNTap, ObsLocTap). Should this be encouraged or discouraged?
- The connection between Apps and Data Models is more tenuous.. primarily through the serialization.. (ucd, UTypes, VODML annotation, Coosys, Timesys, STC-S, DAL Query) all convey model information at different levels to serve different usage threads. How can we improve this relationship to be more effective at getting DM standards to completion?
- Is the model representation the issue? is it more about the serialization? Where is the sweet spot?
  - Do we want serialization standards which apply to different threads?
- While working Meas/Coords, the primary objection was to the complexity, which is needed for the target client. If I could have said.. “For simpler usages, we would do X,Y,Z and you’ll see the content at the appropriate level.”, I think the whole process would have been MUCH easier for everyone.