INTERNATIONAL VIRTUAL OBSERVATORY ALLIANCE
US National Virtual Observatory

# IVOA Data Access Layer
## Table Access Protocol (TAP)
## Including NVO Prototypes

D. Tody (NVO)

# NVO TAP Prototypes

# Current NVO TAP Prototypes

- Current Status
  - IRSA (IPAC)
    - J. Good et. al.
    - ParamQuery, POS, REGION, TAP_SCHEMA metadata
  - HEASARC (GSFC)
    - T. McGlynn et. al. (ParamQuery, POS)
  - STScI (MAST)
    - R. Hanisch, K. Gillies, T. Dower, G. Greene
    - ParamQuery, POS, REGION, TAP_SCHEMA metadata, UWS async
  - JHU (SDSS/OSQ)
    - A. Thakar, S. Carliles
    - Front end to OpenSkyQuery (ParamQuery initially)
  - NRAO (DALServer)
    - D. Tody (Reference implementation, user service framework)

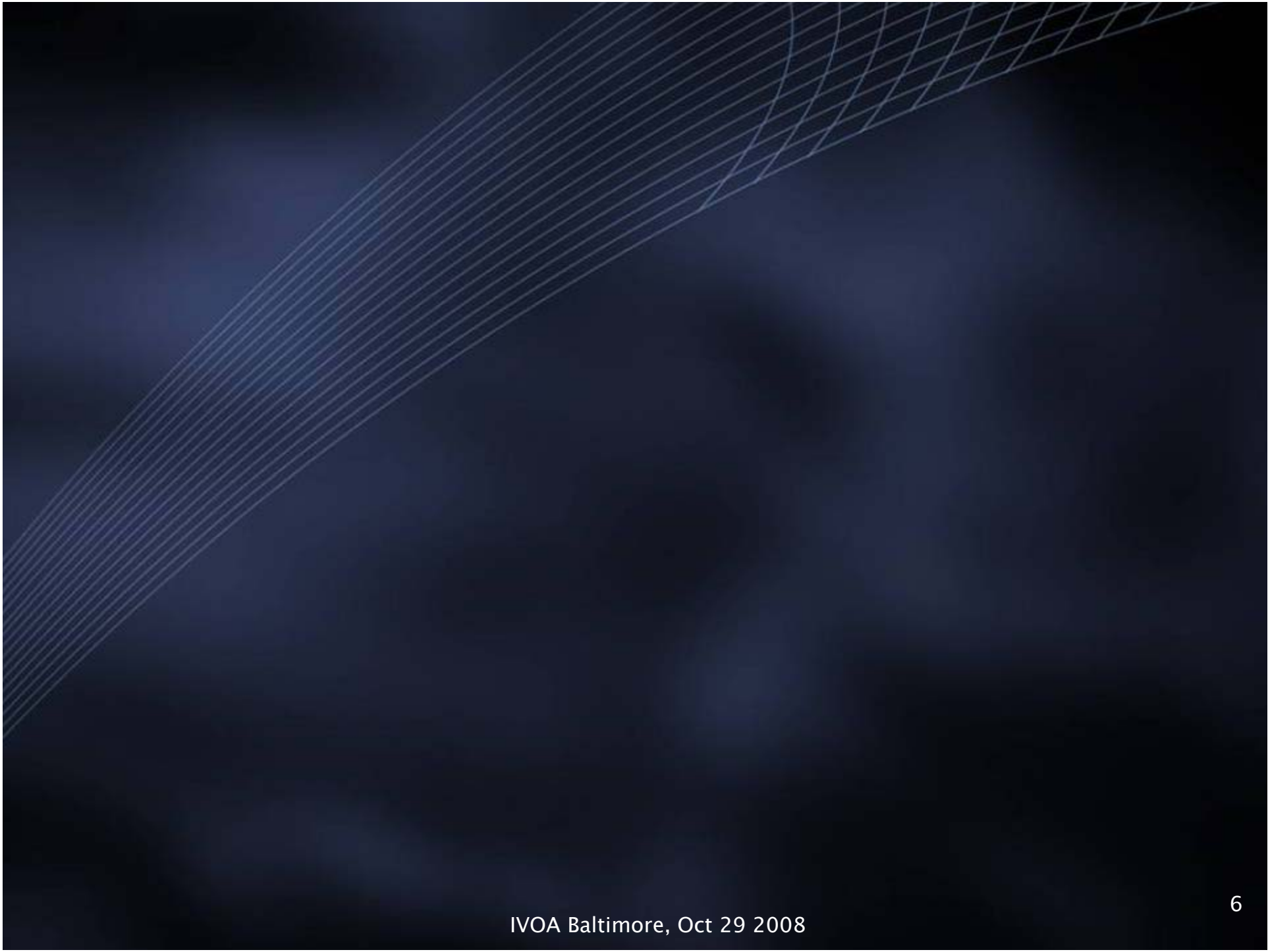
# NVO TAP Prototypes

- Philosophy
  - Target two kinds of "users"
    - End user, data center or advanced application/portal
  - This applies to all the DAL protocols
    - e.g., DAL protocols heavily used in NVO summer school
    - but we need more advanced capabilities for NVO portal

- Strategy
  - Primary use-cases first
    - Simple filter-type queries of astronomical catalogs
    - Client (user) table metadata queries
    - Cone search (optimized spatial query; also REGION)
  - Evolve functionality
    - Provide rigorous and robust basic interface first
    - Prototype advanced capabilities while basic stuff in use
      - ADQL regions, VOSpace, auth, etc.

# NVO TAP Prototypes

- Phase II Prototypes
  - Technology Development
    - Advanced ADQL (region expressions etc.)
    - VOSpace integration, authentication, async (UWS)
  - Portals
    - Casjobs type facility (needs auth, vospace, uws)
    - Distributed cross-match portal



# TAP Interface and Functionality

# ParamQuery

- Service Operation
  - GET|POST $*baseURL*/sync?REQUEST=ParamQuery&...
  - POST $*baseURL*/async: REQUEST=ParamQuery ...
- Parameters
  - POS, SIZE Optimized spatial query (cone search)
  - REGION STC-S region (STC-X possible)
  - SELECT Table fields to be returned ($STD, $ALL)
  - FROM Table to be queried (data or metadata)
  - WHERE Query constraint expression
  - *other* Common query parameters

# Data Queries

- Single Position
  - $*baseURL*/sync?REQUEST=ParamQuery&POS=12,34&SIZE=0.5&FROM=foo
  - POS=12,34 & SIZE=0.5 & FROM=foo & WHERE=jmag,12/
- Multiple Position
  - POS=@TAP_UPLOAD.sources & SIZE=0.5 & FROM=foo
  - POS=@fp_psc & SIZE=0.5 & REGION=<region> & FROM=foo
- General Table (Catalog) Query
  - SELECT=$ALL & FROM=fp_psc & WHERE=z,2.1/3.5;var,/0.1
  - SELECT=ra,dec,id,z & FROM=fp_psc & WHERE=z,2.1/3.5;var,/0.1

# Metadata Queries

- Concept
  - Use standard query mechanism to query TAP_SCHEMA schema
    - Core TAP schema content same as registry VODataService schema
  - All the standard output formats are available
  - Advanced services may permit general POS,SIZE and WHERE
- Basic Metadata Queries
  - List all tables in service tableset
    - REQUEST=ParamQuery & FROM=TAP_SCHEMA.tables
  - List columns of table "foo"
    - FROM=TAP_SCHEMA.columns & WHERE=table_name,foo
- VOSI getTableMetadata
  - FROM=TAP_SCHEMA.tableset & FORMAT=xml (votable also supported)
- Advanced Metadata Query
  - FROM=TAP_SCHEMA.tables & POS=12,34 & SIZE=0.5

# TAP Schema

The table "`TAP_SCHEMA.schemas`" contains the following columns:

| | |
|---|---|
| schema_name | fully qualified schema name (*catalog.schema*) |
| description | brief description of schema |
| utype | UTYPE if schema corresponds to a data model |

The table "`TAP_SCHEMA.tables`" contains the following columns:

| | |
|---|---|
| schema_name | fully qualified schema name (*catalog.schema*) |
| table_name | fully qualified table name (*catalog.schema.table*) |
| table_type | one of: base_table, view, output |
| description | brief description of table |
| utype | UTYPE if table corresponds to a data model |

# TAP Schema

The table "`TAP_SCHEMA.columns`" contains the following columns:

| | |
|---|---|
| column_name | column name |
| table_name | fully qualified table name (*catalog.schema.table)* |
| description | brief description of column |
| unit | unit in VO standard format |
| ucd | UCD of column if any |
| utype | UTYPE of column if any |
| datatype | datatype as in VOTable/Registry |
| arraysize | array dimensions as in VOTable/Registry |
| primary | column is visible in default selection |
| indexed | column is indexed on the server |
| std | standard column (as opposed to custom) |

# Common Query Parameters

| | |
|---|---|
| FORMAT | Format of output table (votable, csv, etc.) |
| UPLOAD | Table upload: UPLOAD=name,URI [; ...] |
| MAXREC | Max table records out. Set to a large value to stream large tables back synchronously |
| MTIME | Query modified or deleted table rows<br>Value is range-list, ISO8601 format |
| RUNID | Pass-through used for Job tracking |

# Query Output

- Output Specification
  - Via either MIME type or shortname for the standard formats
- Output Formats
  - VOTable — Default output format (mandatory)
  - CSV — Comma separate values (recommended)
  - FITS — FITS binary table (optional)
  - text — Pretty-printed text (optional)
  - html — Web page (can use Javascript etc.)
- Disposition
  - Direct back to client (synchronous query)
  - Cache on server for retrieval (UWS)
  - Direct to VOSpace, eventually