International Virtual Observatory Alliance

IVOA Data Access Layer
SSAP Update and Issues

Doug Tody (NRAO/NVO)
Markus Dolensky (ESO/EuroVO)
Data Access Layer Working Group

Spectral Access Protocols

- Agenda
  - DAL-1 (Monday afternoon)
    - Doug Tody – SSA Protocol Update
    - Doug Tody – getCapabilities Proposal and Issues
    - Ray Plante – Defining VO Extensions for DAL Protocols
    - All – Discussion of getCapabilities, Registry/DAL integration
  - DAL-2 (Joint session with DM) (Tuesday morning)
    - Jesus Salgado, Pedro Osuna – SLAP Update and Plans
    - Jonathan McDowell – Spectrum Data Model
    - All – Discussion on taking SSAP, Spectrum to PR

Simple Spectral Access

- Status
  - Resolved major data model issues by summer 2006
  - SSAP sufficient to support initial implementations since November
    - Half a dozen or so implementations completed or in progress
    - Reference implementations of SSAP, Spectrum available
  - Current specifications
    - Updated SSAP V1.0WD now available
    - Spectrum complete, reasonably stable
    - Integration with Characterization improved
- Issues
  - Only real remaining issue is getCapabilities
  - Registry/GWS integration is becoming the new hot topic

Recent SSAP Changes

- Input parameters
  - POS, SIZE
    - How to support coord frames
    - Current syntax: coord1,coord2;<*frame*>
    - Is only the frame name enough?
  - SPECRP replaces SPECRES
    - Spectral resolving power

- New parameters
  - VarAmpl (specified as a range)
  - FluxCalib (boolean)

Recent SSAP Changes

- Handling large queries
  - Issue is how to simplify implementations
  - Suggested change is to replace TOKEN with MAXREC
    - Default value of MAXREC is chosen for speed
    - Client bumps MAXREC to maximum to attempt large queries
  - TOKEN approach
    - May require caching query response on the server

- Range Lists
  - Heavily used within SSAP query parameters
    - Both lists and ranges proved popular in implementations
  - Ordered, unordered range lists
    - Service sorts ordered range lists upon input
    - Unordered lists are processed in the client-specified order
  - Both numeric and string valued lists are useful

Recent SSAP Changes

- Mime-type issue
  - Query response vs dataset serialized as VOTable
    - text/xml;<*param*>'='<*value*>,...
    - application/x-votable+xml
- Query response
  - Metadata issues deferred to Spectrum DM discussion
  - Use of ID as short name for Utype (convention)
  - Use of GROUP/UType constructs
    - minimize nesting; use fully qualified UTypes
- Use of Associations
  - Replaces old "logical name" proposal

Recent SSAP Changes

- Service Operation Response
  - Normal completion
  - Error response
  - Overflow condition
- Use of VERSION
  - Protocol versions supported returned by getCapabilities
  - Check version (to level 2) if specified
  - Otherwise default to highest standard version
- Compression
  - File-level
  - Protocol-level

Advanced Service Operations

getCapabilities

Advanced Service Operations

- Basic DAL Service Profile
  - Common pattern for all 2ndGen DAL services
  - Common starting point for advanced capabilities
    - asynchrony, registry integration, etc.
  - Promotes code sharing between client/server implementations

- Operations

– queryData	find Datasets, often virtual
– (getData)	retrieve a single dataset
– getCapabilities	get the service capabilities
– (stageData)	initiate asynchronous operation (w. UWS)
– getAvailability	monitor service health (w. VOSI/GWS)

getCapabilities Operation

- Metadata Handling
  - getCapabilities addresses part of the metadata handling problem
- Classes of Metadata
  - Resource metadata (uniform profile for all resources)
  - Service metadata (service-specific; uniform approach)
  - Dataset content metadata (e.g., table/column)
  - General dataset metadata (DataID, Char, etc.)

getCapabilities Operation

- Purpose
  - Return *service* metadata
  - Metadata is returned directly to a client application
  - May also be cached in registry and used for discovery
    - Hence, registry integration is important
- Motivation
  - Replaces old FORMAT=METADATA mechanism
  - Specify capabilities, limitations of a service instance
  - Provide introsection of service interface (input params)
    - needed to support custom service-level parameters (eg TSAP)
    - also provides way to tell which params are actually used
  - Support integration of service metadata with registry

getCapabilities Proposal

- Approach
  - Returns XML doc containing only service metadata
  - Includes service-defined "Capabilities" element of a VOResource
  - Can be autogenerated, or produced from a simple fixed template
- User app as client
  - Not even needed in many cases due to registry-based selection
  - Parse/load response into client-side Capabilities class
  - Application can directly query capabilities and interface
- Registry as client
  - Registry Web or programmatic API can be used for registration
  - Thereafter, registry can "pull" information from service
  - Only service capabilities are affected

getCapabilities

- Issues
  - Who is the client?
  - What metadata does getCapabilities return?
    - Only “service metadata”, or more?
  - Complexity and size of metadata
    - Minimize burden on service implementors, operational staff
      - Split responsibility between the registry and the service; allow registry to define, curate high level resource metadata
    - Modularize overall system
      - Service implementor should not have to understand entire VO
    - Optimize client interface for each major class of metadata
  - Registry semantics, e.g., if resource metadata is included; where is resource metadata curated?