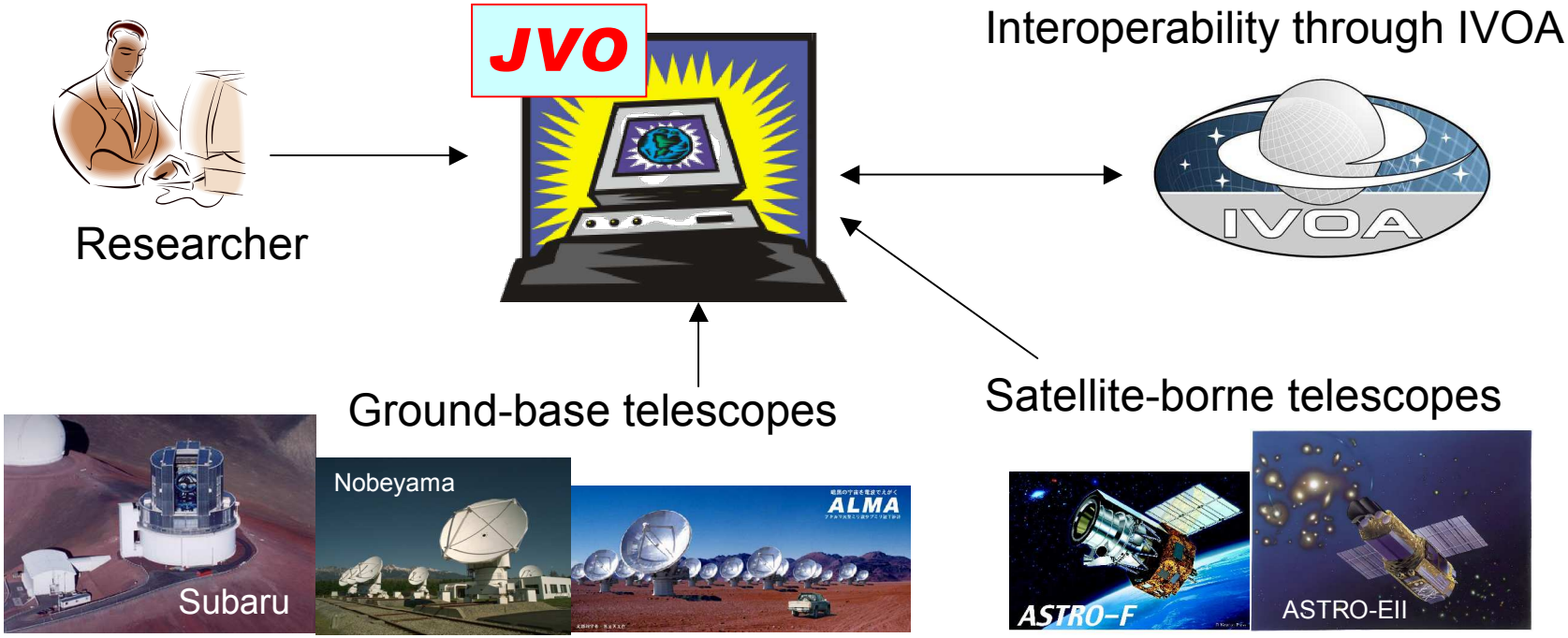
Development of JVO prototype system and its application to Astrophysics

Portal System : M. Tanaka
Data Service : Y. Shirasaki
Science Applications : S. Honda

JVO : Japanese Virtual Observatory

- Purpose:
  - Easy access to federated Astronomical databases
  - Interoperability through IVOA



JVO activities

- Development of JVO System
  - Federated Data Servers using Grid
- Development of Query Language for JVO
- Interoperability in IVOA
  - work on VOQL WG
  - chair: Prof. Ohishi

JVO collaborators

Project Scientists

NAOJ

- Mizumoto
- Ohishi
- Shirasaki
- Tanaka
- Honda
- Kawanomoto

ICRC

- Yasuda

Ochanomizu U.

- Masunaga

System Engineers

Fujitsu Ltd.

- Monzen
- Kawarai
- Ishihara
- Yanaka
- Yamaguchi
- Ishida
- Saito
- Abe
- Tsutsumi

SEC Ltd.

- Morita
- Nakamoto
- Kobayashi
- Yoshida

JVO Portal System

Masahiro Tanaka

National Astronomical Observatory of Japan

Development Strategy

- "Top-down" approach
  - set "Science Use Cases"
  - study and design "Overall System"
    - How to federate distributed computers?
  - build "Prototype System"
    - with minimal capabilities
    - to achieve use cases
- "Build-and-scrap" prototype

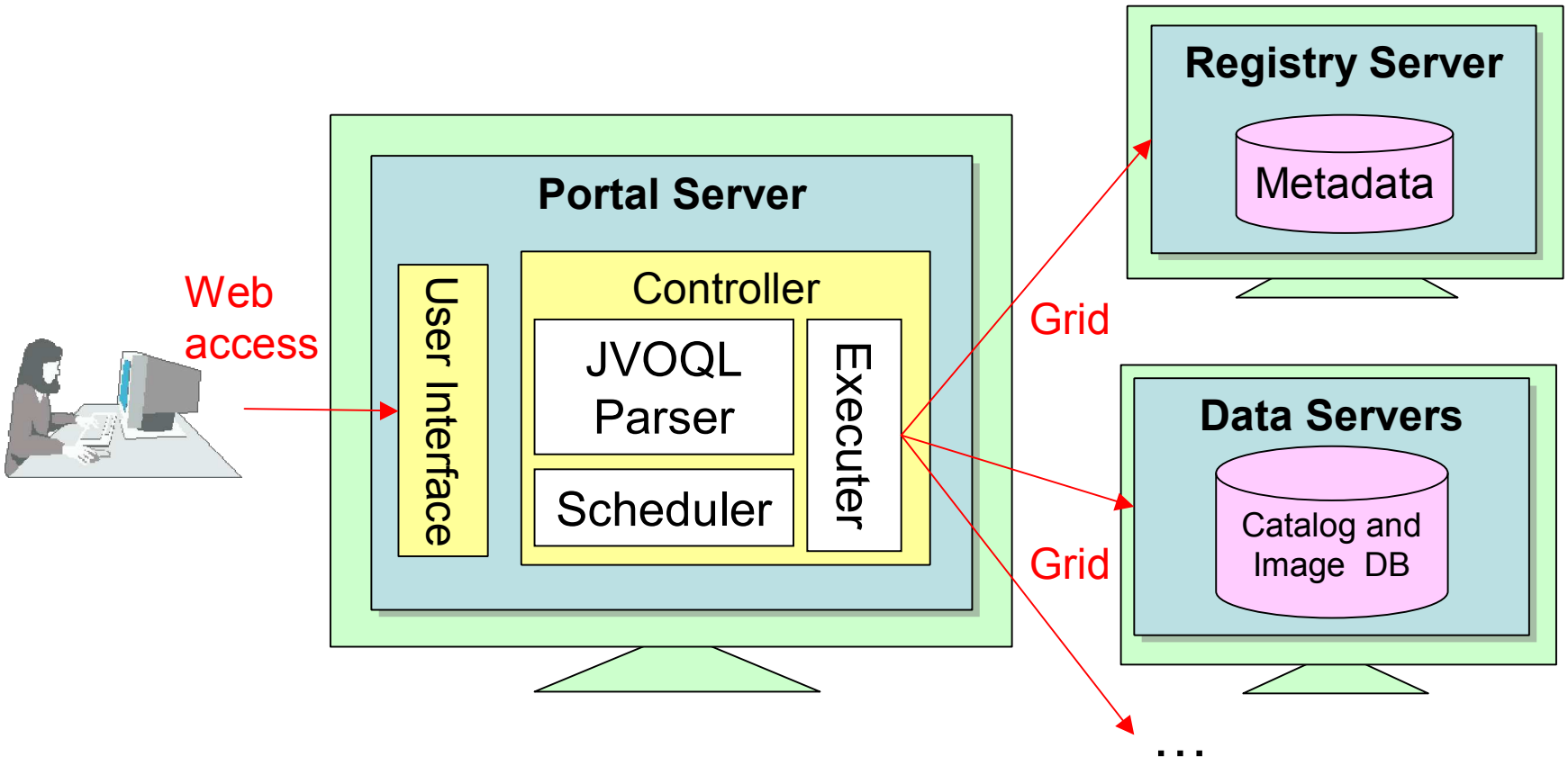
Development history

- JVO Project start — April 2002
- Prototype 1 finish — March 2003
- Prototype 2 finish — March 2004
- Prototype 3 — under development

JVO Portal Server

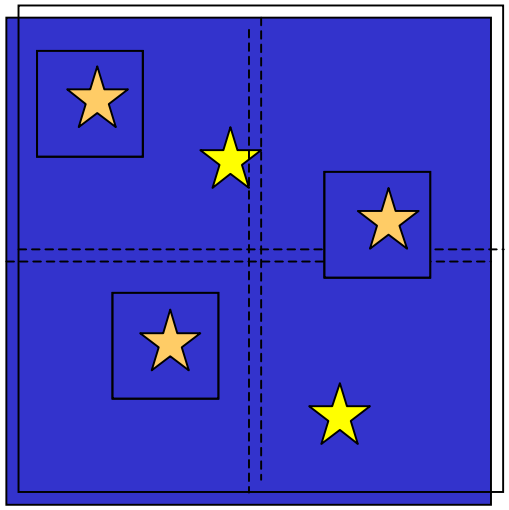
- Web Interface to users
- Interpret JVO Query Language
- Retrieve query result from Distributed Data Servers
- VOTable and FITS image browser

JVO components



Use Case

- Cross-matching z'- and i'-band catalogs of Subaru Deep Field
- Retrieving images for each object



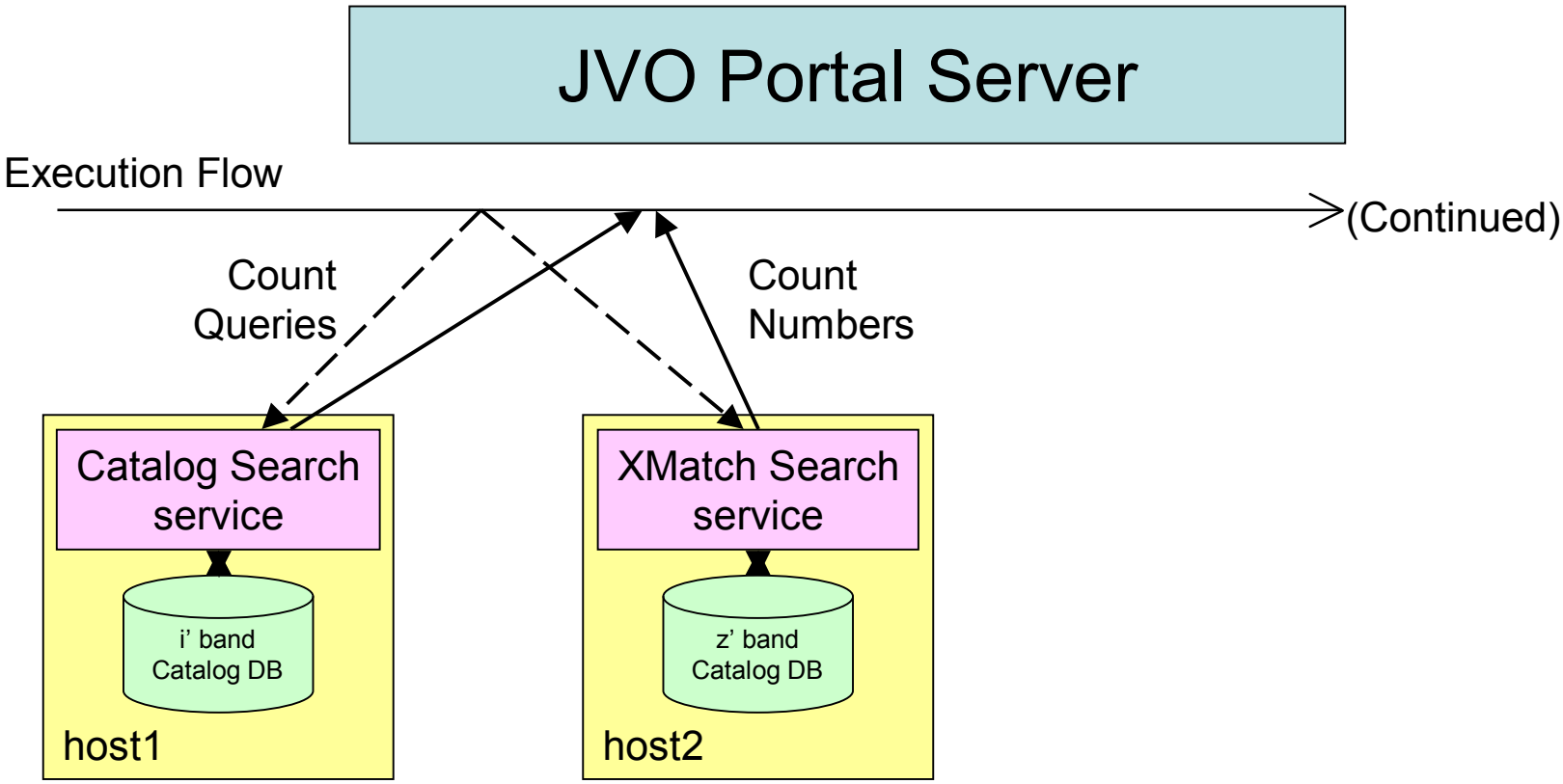
JVOQL


```
select
    i.POS_EQ_RA_MAIN as ra1,
    i.POS_EQ_DEC_MAIN as dec1,
    i.PHOT_SDSS_I,
    z.POS_EQ_RA_MAIN as ra2,
    z.POS_EQ_DEC_MAIN as dec2,
    z.PHOT_SDSS_Z,
    i.BOX(POINT(ra1,dec1), 20 arcsec, 20 arcsec),
    z.BOX(POINT(ra2,dec2), 20 arcsec, 20 arcsec)
from
    naoj.sdf.i i,
    naoj.sdf.z z
where
    XMATCH(i, z) < 10 arcsec NEAREST and
    BOX(POINT(201., 27.4), 5 arcmin, 5 arcmin)
```

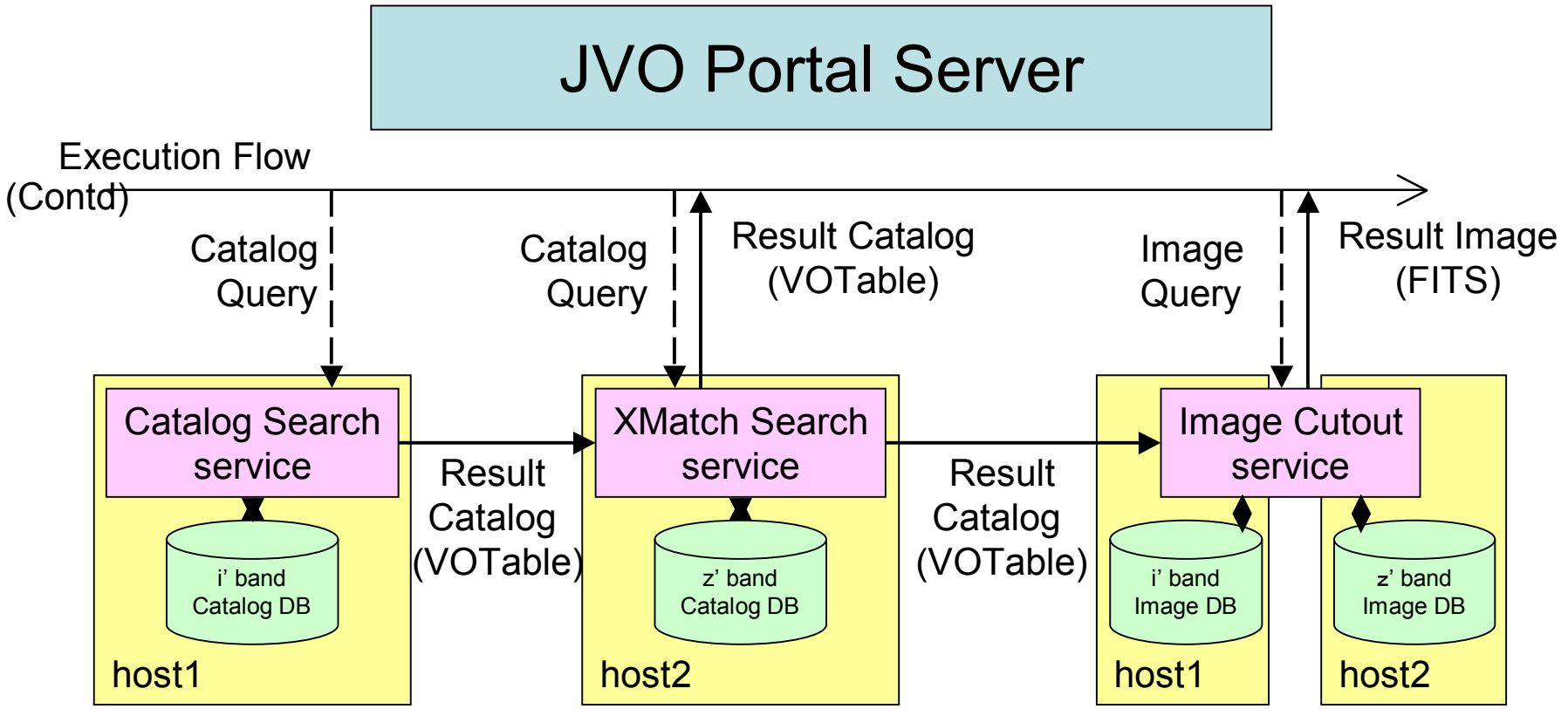
Parsing JVOQL and Generating Workflow

- “JVOQL Parser” generates query for each host
- “Scheduler” generates:
  - count query job for host1
  - count query job for host2
- “Executer” executes jobs on remote hosts
- “Scheduler” generates based on the result of execution
  - query job for host1
  - xmatch job for host2
  - image query for host1 and host2
- “Executer” executes jobs on remote hosts

Federateion of Distributed Servers 1



Federateion of Distributed Servers 2

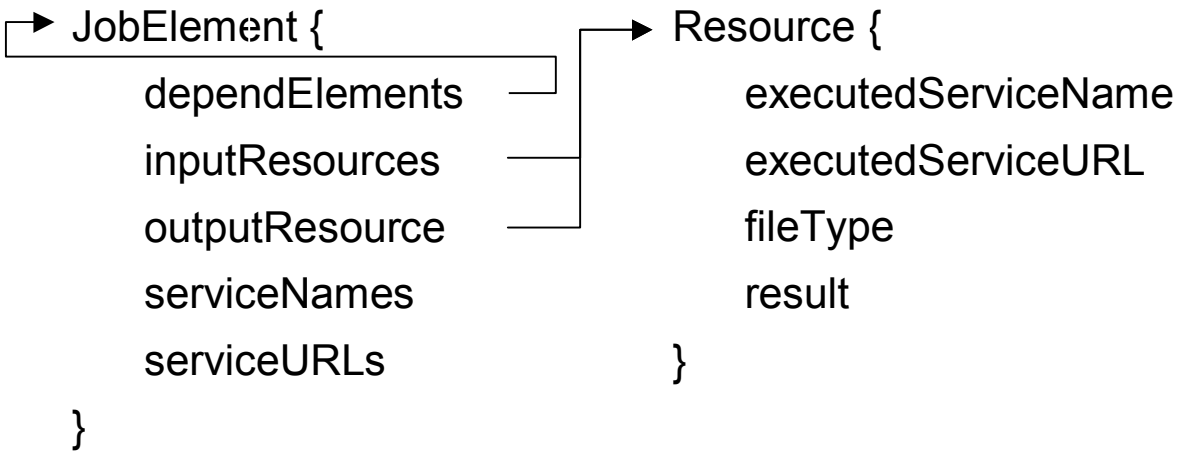


Description of Workflow

- Count queries can be executed in parallel.
- Search and XMatch service must be called sequentially.

→ "Dependency" is considered.

  – Parallel execution is to be implemented.

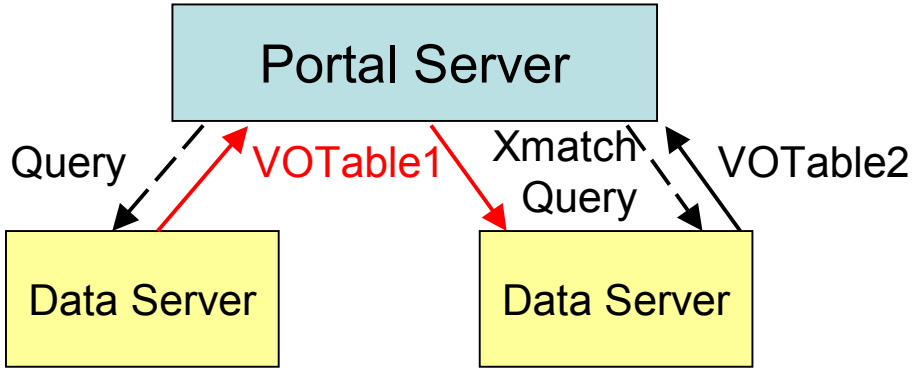


Remote execution

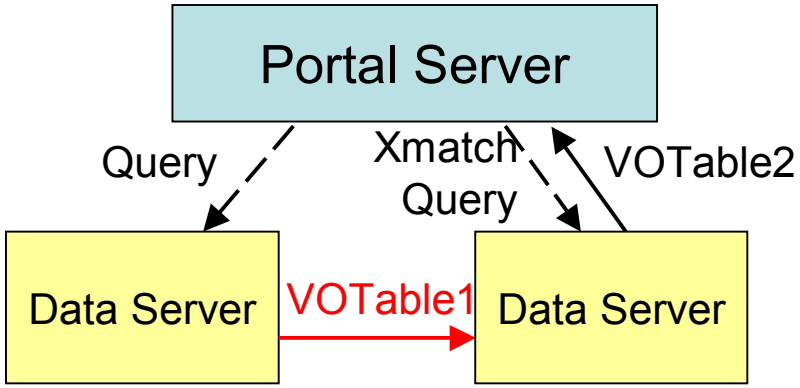
- Proto1:
  - Globus toolkit ver 2
  - using `globus-job-run` command
  - 1 call = 30 sec
    - 1 query ~10 min!
- Proto2:
  - Globus toolkit ver 3
  - using Grid Service
  - 1 call = 2-3 sec
    - overhead time is only ~ 30 ms

Data Transfer

- Web/Grid Service
  - Query result is always returned to Portal server



- GridFTP
  - Query result can be directly transferred to XMatch server



Data Transfer

- We tried:
  - GridFTP, RFT
  - GSI-SFS
- Learned:
  - SFS is not flexible
    - A server cannot be a client.
  - RFT is promising in Globus environment
  - need support for HTTP and Web Services

Data Discovery

- Proto1: UDDI
  - UDDI is for “Service discovery”, not for “Data discovery”
- Proto2: XMLDB
  - XMDB product “Karearea” is used.
  - XPath search
  - enables both “Data discovery” and “Service discovery”

Resource Metadata

Identity metadata | service metadata | column metadata

curation metadata | content metadata

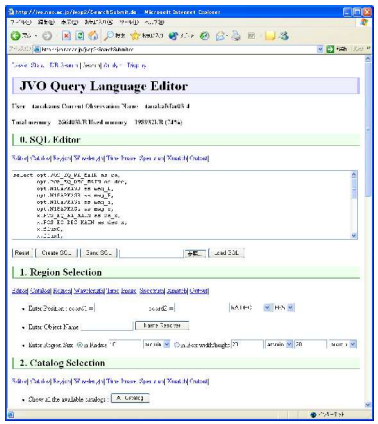
title	string
short_name	string
identifier	URI
publisher	string
publisher_id	URI
creator	string
creator_logo	URL
contributer	string
date	string
version	string
contact_name	string
contact_email	e-mail address
service_interface_url	URL
service_base_url	URL
service_http_result	MIME type
service_standard_uri	URI
service_standard_url	URL
service_msr	float,decimal degrees
ucd	string
unit	string
datatype	string
width	int
precision	string
arraysize	string

	identity	curation	service	content	column
catalog	○	○	○	○	×
table	○	○	○	○	×
column	○	○	○	○	○

subject	string,list
description	string,free text
source	string
reference_url	URL
type	string,list
coverage_spatial	string
coverage_region_of_regard	float,decimal degrees
coverage_spectral	string,list
coverage_spectral_bandpass	string,list
coverage_spectral_central_wavelength	float
coverage_spectral_minimum_wavelength	float
coverage_spectral_maximum_wavelength	float
coverage_temporal_start_time	string
coverage_temporal_stop_time	string
coverage_depth	float
coverage_depth_unit	string
coverage_object_density	float
coverage_object_count	int
coverage_sky_fraction	float
resolution_spatial	float
resolution_spectral	float
resolution_temporal	float
content_level	string,list
facility	string,list
instrument	string,list
format	string,list
right	string

User Interface

- Java Servlet
  - Tomcat
  - Struts
- Easy use by scientists (S. Honda's talk)
  - needs only Web browser, no extra installation.



Plans toward Prototype 3

- support SIAP, SSAP, SkyNode
- implement ADQL

} next talk by Y. Shirasaki

- improve registry, employing OAI-PMH architecture
- flexible workflow architecture
- introduce User management
  - LDAP
  - User Storage Area (support VOStore?)
- API to control JVO with SOAP

END