MocServer 2: What, Where and When in milliseconds.

Benefits for VO tools. Demonstration with Aladin Desktop

Interop Automn (virtual) – 2-4 November 2021

Pierre Fernique, Mathieu Baumann, Thomas Boch, Sébastien Derrière,
Daniel Durand, Gilles Landais, Ada Nebot, François-Xavier Pineau,
& all other contributors

Observatoire astronomique de Strasbourg | ObAS

For newcomers, a MOC...

Abstract

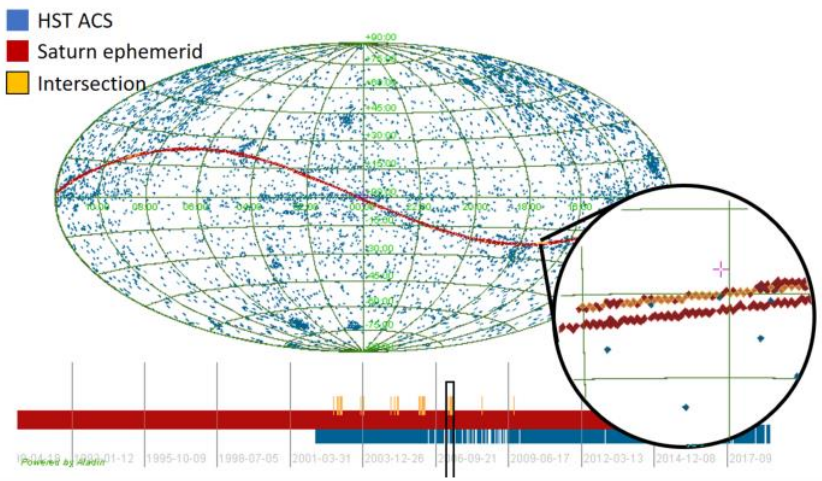
This document describes the Multi-Order Coverage map method (MOC) version 2.0 to specify arbitrary coverages for sky regions and/or time coverages and potentially other dimensions. The goal is to be able to provide a very fast comparison mechanism between coverages. The mechanism is based on a discretization of space and time dimensions. The system is based on the definition of a specific storage of the map coverage using predefined cells hierarchically grouped which makes it easy to produce and use for exploring astronomical collections. There are already a few applications and libraries which are taking advantage of this new standard.

… specifies arbitrary **coverages** for **sky regions** and/or **time coverages**…

… provides a **very fast comparison** mechanism…

… is based on a discretization of space and time dimensions…

… is based on specific storage of the map coverage using predefined cells hierarchically…



See the IVOA Moc2.0 document for details

The MOC2 time line

2018

Demonstration of the feasibility of adding a temporal dimension to MOCs

Standardisation process:

2019

- IVOA decision to add a time dimension to MOCs
- **Redaction** of a new version of the MOC document:
  WD -> **PR** -> REC

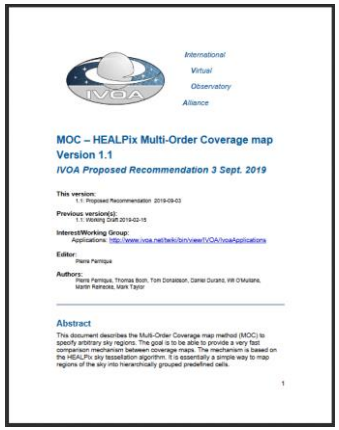
2020

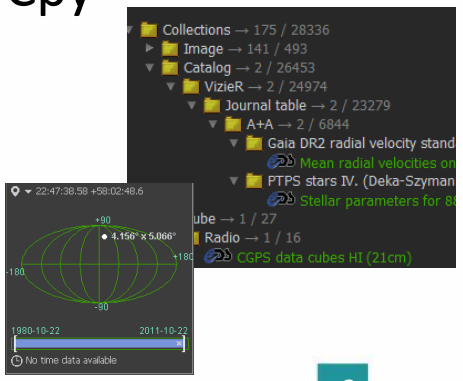
- Extension of **MOC libraries** javaMoc and MOCpy

Evolutions of **VO MOC tools**

2021

- MocServer, Aladin, VO registry ? TOPCAT? …





MocServer 2

- What is it exactly? What's new ?
- How does it work?
- What does it contain?
- How to use it? Who is using it?

What is MocServer 2?

A “small” Web **server** able to **answer in a few ms** to these 3 categories of questions:

- Which data collections coincide with a predefined space *and/or time* coverage?
- Which data collections meet a particular set of criteria? (other than spatial or temporal constraints)
- What is the coverage (space *and/or time*) of one or more data collections?

In other words, MocServer 2 uses the same operating principles as MocServer 1, but integrates now temporal and spatio-temporal MOCs

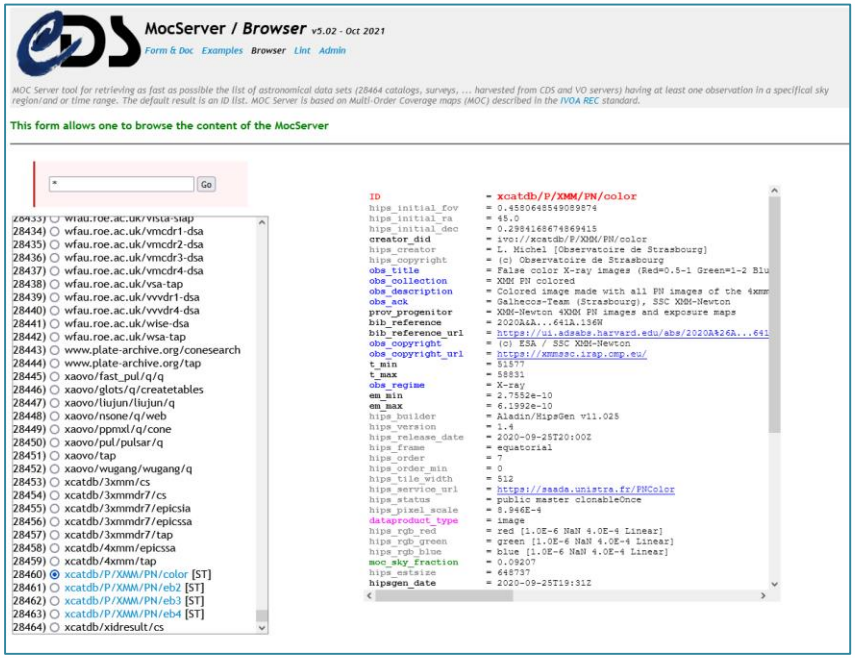
How does it work ?

- A **Tomcat servlet** hosted at CDS
- **Loading/managing in RAM thousands of pairs**
  => N x (MOC + properties)
- Pairs are coming from a special directory containing all the MOCs and their associated properties files
- This directory is **regularly "updated"** via a series of small scripts dedicated to VizieR, Simbad, HiPS servers, VO RegTAP...

=> MocServer2 uses the same architecture as MocServer1

What does it contain today?
(22/10/2021)

- 28,464 datasets descriptions (properties) and coverages (MOC):
  - 23,832 SMOC
  - 1,212 TMOC
  - 1,045 STMOC



***Big work** of the CDS team+ to get the temporal data from the VizieR catalogs and HiPS surveys & generate the T,STMOC notably with MOCpy*

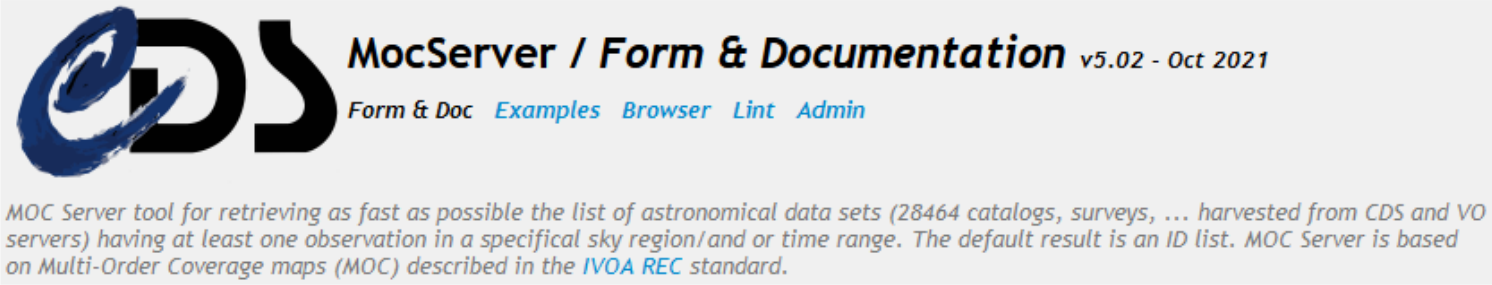
Evolution from 2015 to 2021

- **Better content**:
  - 15000 collections -> **28000** collections
  - SMOC only -> SMOC + **TMOC** + **STMOC**
- **Better accuracy**:
  SMOC Order 8 -> **11** => now 1.72' => 64x more accurate
- **Better capabilities**:
  - Basic queries -> Extended queries
  - Manual updates/checks => **Automatic updates/checks** (each day)

- **Size In RAM**: 300MB -> **3GB**
- **Performance**: Average time response: 95ms -> **110ms**
  (with an upgrade of the server)

How to use it?

- A simple web site: https://alasky.cds.unistra.fr/MocServer/...



- 5 forms:
  - …/**query**: **query form & docs**
  - …/example: examples & tests
  - …/browse: interactive browser
  - …/lint: MOC lint form
  - …/admin : administration form *(restricted access)*

Basic queries

1) Cone & time search

MOC server can be queried by SIMPLE CONE SEARCH syntax. The search region is a circle on the sky and/or a time range.

RA,DEC must be expressed in degrees (ICRS), SR in degrees, TIME as 2 MJD values space separatedintersect param determines if the region must overlap (default), enclosed, or cover the matching collection coverages:

RA= 10.8 & DEC= 32.2 & SR= 1.5 & TIME= & intersect= overlaps Go

New field

2) Basic space STC search

MOC server can be queried via a basic STC syntax. The search region is a STC string:

This version only supports Circle & Polygon in ICRS

stc= Polygon 57.376 24.053 56.391 24.622 56.025 24.049 56.616 24.29 & intersect= overlaps Go

3) Inline MOC search

MOC server can be queried by MOC string. The search coverage is a MOC (space, time or space-time) expressed as a string.

MOC must be a string following the ASCII or JSON syntax

moc= t16/6020-6022 s3/128 4/345-510 & intersect= overlaps Go

New capabilities

4) URL MOC search

MOC server can be queried by a remote MOC. The search coverage is a remote MOC uploaded via an URL.

MOC must be a HTTP stream (FITS, ASCII or JSON syntax)

url= http://alasky.u-strasbg.fr/SDSS/DR9/color/Moc.fits & intersect= overlaps Go

5) File MOC search

MOC server can be queried by a local MOC file. The search coverage is a local MOC uploaded via Multipart POST method (multipartID: moc).

MOC must be a local file (FITS, ASCII or JSON syntax)

Parcourir... Aucun fichier sélectionné. & intersect= overlaps Go

Advanced queries

i) Filtering by IDs or properties and output control

MOC server can also be queried/filtered by IDs(*) or any other property values. In this case, the query is not a sky region but a combination of ID mask and property values. The default ASCII output format can be replaced by JSON or JSONP. And the result list ca provide a full record for each data set rather than a simple ID. In this case, the ouptut fields can be designated explicitely. It is also possible to get the number of matching data sets, or to be HTTP redirected on the URL (only in case of an unique answer and url field

The ID selection must be an identifier or a template(**) of identifier or a comma separated template list. The field names are used for filtering the properties (***)
The associated value must be a template(**) or a a comma separated template list. The boolean logic is OR inside comma lists and AND between fields.
In "record" get mode, the list of output fields can be explicitely specified in "fields" as a comma separated template(**) list.

ID= CDS* & *moc_sky_fraction*= <0.01 & ...
& get= record & fields= !obs_description & fmt= ascii
& casesensitive= true & MAXREC= 3 & Go

Previous basic query modes can also accept any of these additional parameters in order to filter and/or control the result.

ii) Full set algebrica language control

Alternatively, MOC server can be queried via a full set algebra expression.

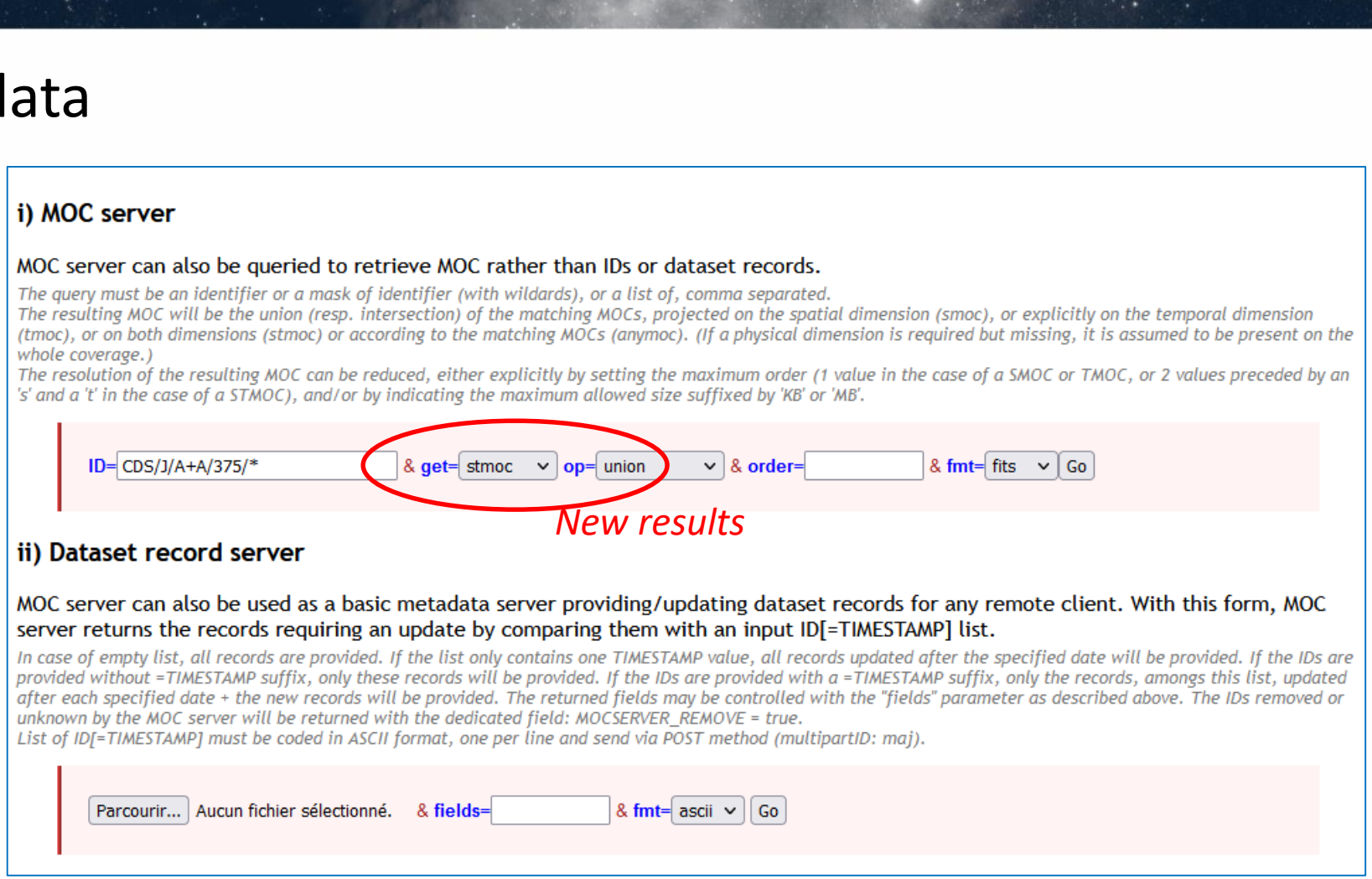
Each operand is a filter rule expression as described above. The operators are the union (→ ||), the intersection (→ &&) and the subtraction (→ &!).Parenthesis m be used to change the default operator priority. Expression values containing control characters (,),|,& must be quoted (" or '), and the possible internal quotes mus be backslashed.Output control fields described above may be added for controlling the output content and format.

expr= ((ID=xcatdb* || obs_regime=X-ray,UV) && hips_service_url=*) &! obs_*=*CONSTEL*
& get= id & fmt= ascii Go

New query language

Previous query modes can also accept the MOC retrievial mode. In case of Cone Search or other query by region, the resulting MOC will be the intersection with the union (or intersection) of all matching MOCs

Meta data server



Examples/Bench – space related

82ms
3626 rec

. **Around M31** - IDs of all data sets in 5 deg around M31
Returns the ID list of all data sets overlapping a circle of 10 degree diameter cent
→ http://.../MocServer/query?RA=10.67305&DEC=41.26875&SR=5

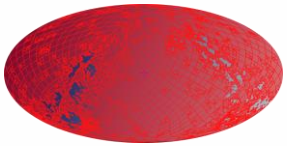
```
CDS/C/CGPS-HI
CDS/C/HIPASS
CDS/J/A+A/374/227/table4
CDS/J/A+A/384/650/table
CDS/J/A+A/501/519/table7
CDS/J/A+A/520/A89/rv
CDS/J/A+A/596/A116/table2a
CDS/J/A+A/601/A31/table2
CDS/J/A+A/616/A118/table6
CDS/J/A+A/616/A7/rvstdcat
CDS/J/A+A/616/A7/rvstdmes
CDS/J/A+A/619/A1/apt55cnc
CDS/J/A+A/619/A1/rv55cnc
CDS/J/A+A/619/A81/rv-mease
CDS/J/A+A/619/A85/table1
CDS/J/A+A/619/A97/tablea5
CDS/J/A+A/619/L10/tablea1
CDS/J/A+A/622/A199/tablea1
CDS/J/A+A/622/A22/list
CDS/J/A+A/624/A123/tablea1
CDS/J/A+A/625/A1/table1
CDS/J/A+A/625/A1/table2
CDS/J/A+A/625/A59/rvdat1
CDS/J/A+A/625/A71/rv
```

120ms
1328 rec

. **Overlapping a STC polygon** - IDs of all data sets overlapping a polygon
→ http://.../MocServer/query?stc=Polygon+57.376+24.053+56.391+24.622+56.025+24.049+56.616+24.290

80ms

'. **SCUBA2 spatial coverage** - Spatial MOC of SCUBA2 observations
Provides the spatial MOC of the SCUBA2 850em observations. The MOC is provided in FITS format.
→ http://.../MocServer/query?ID=CDS/P/SCUBA2/850em&get=smoc



112ms
2smoc

GALEX & HERSCHEL spatial union - Spatial MOC of the GALEX and HERSCHEL coverage
Provides the union of spatial MOCs of GALEX-NUV and HERSCHEL-PACS observations
→ http://.../MocServer/query?ID=CDS/P/GALEXGR6/AIS/NUV,ESAVO/P/HERSCHEL/PACS-color&get=smoc

Examples/Bench – time related

8ms
233 rec

. **In 2000** - IDs of all data sets in 2000
Returns the ID list of all data sets having on observation during the year 2000
→ http://.../MocServer/query?TIME=51544+51910

5ms
134 rec

. **Overlapping an STMOC** - IDs of all data sets overlapping an inline STMOC
Returns the ID list of all data sets overlapping the inline MOC t16/6020-6022 s3/128 4/345-510
→ http://.../MocServer/query?moc=t16%2F6020-6022+s3%2F128+4%2F345-510

```
CDS/C/CGPS-HI
CDS/C/HIPASS
CDS/J/A+A/374/227/table4
CDS/J/A+A/384/650/table
CDS/J/A+A/501/519/table7
CDS/J/A+A/520/A89/rv
CDS/J/A+A/596/A116/table2a
CDS/J/A+A/601/A31/table2
CDS/J/A+A/616/A118/table6
CDS/J/A+A/616/A7/rvstdcat
CDS/J/A+A/616/A7/rvstdmes
CDS/J/A+A/619/A1/apt55cnc
CDS/J/A+A/619/A1/rv55cnc
CDS/J/A+A/619/A81/rv-mease
CDS/J/A+A/619/A85/table1
CDS/J/A+A/619/A97/tablea5
CDS/J/A+A/619/L10/tablea1
CDS/J/A+A/622/A199/tablea1
CDS/J/A+A/622/A22/list
CDS/J/A+A/624/A123/tablea1
CDS/J/A+A/625/A1/table1
CDS/J/A+A/625/A1/table2
CDS/J/A+A/625/A59/rvdat1
CDS/J/A+A/625/A71/rv
```

869ms
3691 tmocs

. **AJ temporal coverage** - Temporal MOC of all Aj tables
Provides the union of temporal MOCs all all tables published in AJ.
→ http://.../MocServer/query?ID=CDS/J/AJ*&get=tmoc

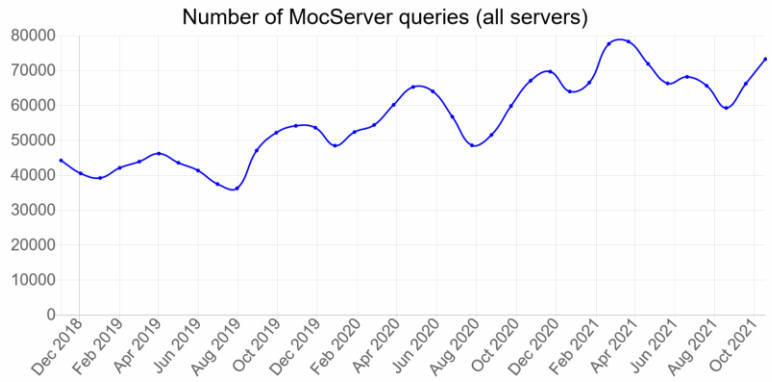
70ms

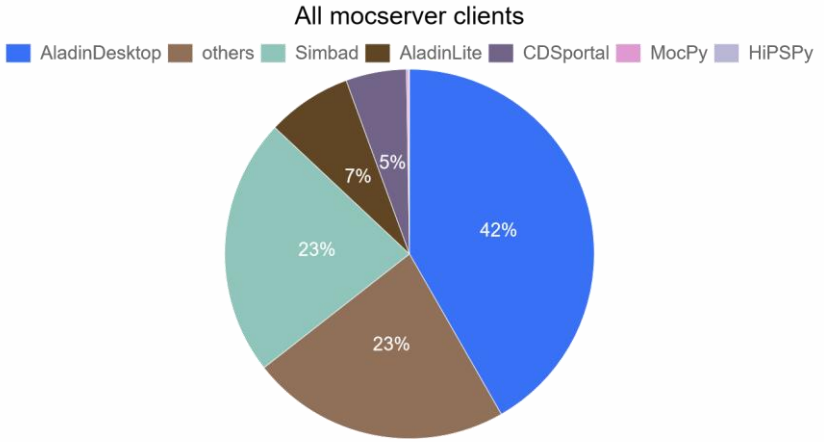
. **Spatio-temporal coverage of LAMOST** - Spatio-temporal MOC of the LAMOST DR4 observations
Provides the spatio-temporal MOC of all coincidental observations in time and space of LAMOST & ALMA calibrator cat observations
→ http://.../MocServer/query?ID=CDS/V/153/dr4&get=stmoc

75ms

. **LAMOST & ALMA calibrator cat co-coincidental coverage** - Spatio-temporal MOC of the LAMOST & ALMA calibrator cat coincidental observations
→ http://.../MocServer/query?
ID=CDS/V/153/dr4,CDS/J/MNRAS/485/1188/acccat&get=stmocc&op=intersection

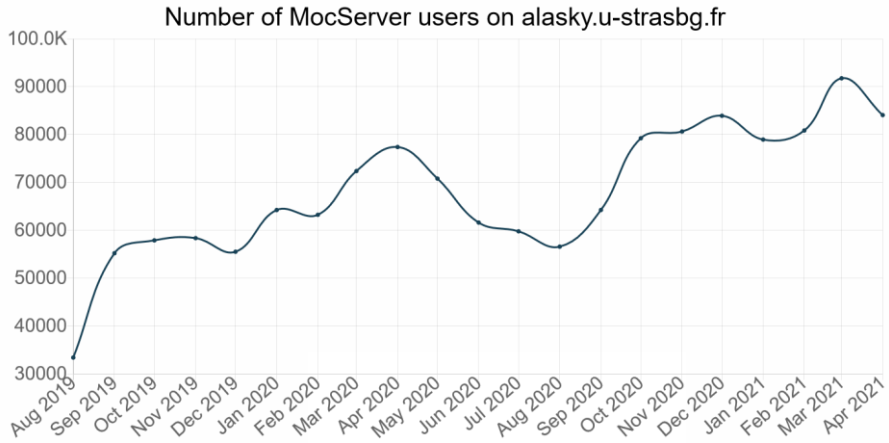
Who is using MocServer? (v1)



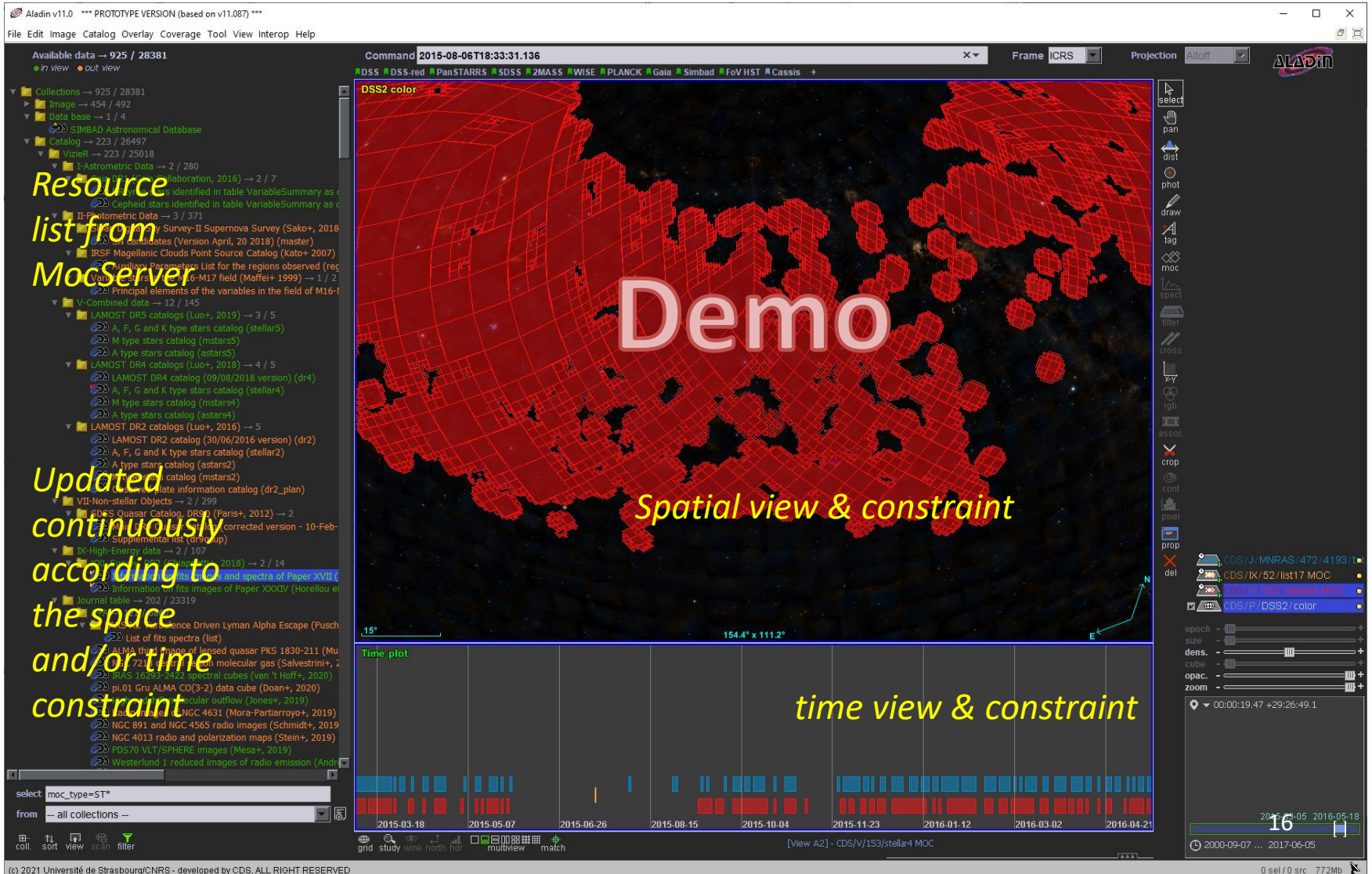


- Around **70,000 queries per day**
- From **84,000 IP** (last month)
- **A quarter by non CDS clients** through dedicated python libs

(MocServer 1 has been deployed in 2015)



Aladin Desktop beta as MocServer v2 client



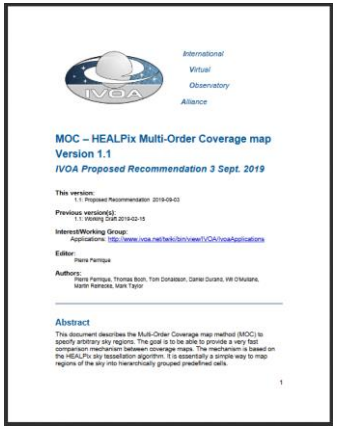
Lessons learned

- The addition of the temporal dimension does **not alter significantly the performance**.
- The temporal extension of the MocServer **required some adjustments**:
  - Acceleration of hierarchical MOC <-> MOC interval transformations
  - Internal hybrid range coding to save RAM
  - Parallelization of the loading procedure

- Finally, it works, and it even **works surprisingly well**

MOC 2.0 without delay

- Mocserver 2 is **already operational** (2 duplicated sites) since 2 months
- Fully **compatible with MocServer 1 clients**
  (including all Aladin Desktop non beta)


International Virtual Observatory Alliance

MOC – HEALPix Multi-Order Coverage map
Version 1.1
IVOA Proposed Recommendation 3 Sept. 2019

This version:
1.1: Proposed Recommendation 2019-09-03
Previous version(s):
1.1: Working Draft 2019-02-15
Interest/Working Group:
Applications. http://www.ivoa.net/twiki/bin/view/IVOA/IvoaApplications
Editor:
Pierre Fernique
Authors:
Pierre Fernique, Thomas Boch, Tom Donaldson, Daniel Durand, Wil O'Mullane, Martin Reinecke, Mark Taylor

Abstract
This document describes the Multi-Order Coverage map method (MOC) to specify arbitrary sky regions. The goal is to be able to provide a very fast comparison mechanism between coverage maps. The mechanism is based on the HEALPix sky tessellation algorithm. It is essentially a simple way to map regions of the sky into hierarchically grouped predefined cells.

1

⇒ **MOC 2.0 PR phase** started last week

Next steps:

- Evolution of the **python lib** for supporting MocServer time extension
- **VO needs more MOC** (S,T,ST) available in/from the VO registry:
  - Use "Hipsgen.jar … STMOC" action => to generate a STMOC from a HiPS
  - Use MOCpy/rust/wasm (FX Pineau's tools&lib)
    => https://github.com/cds-astro/cds-moc-rust