Application Working group

Discussion on standards

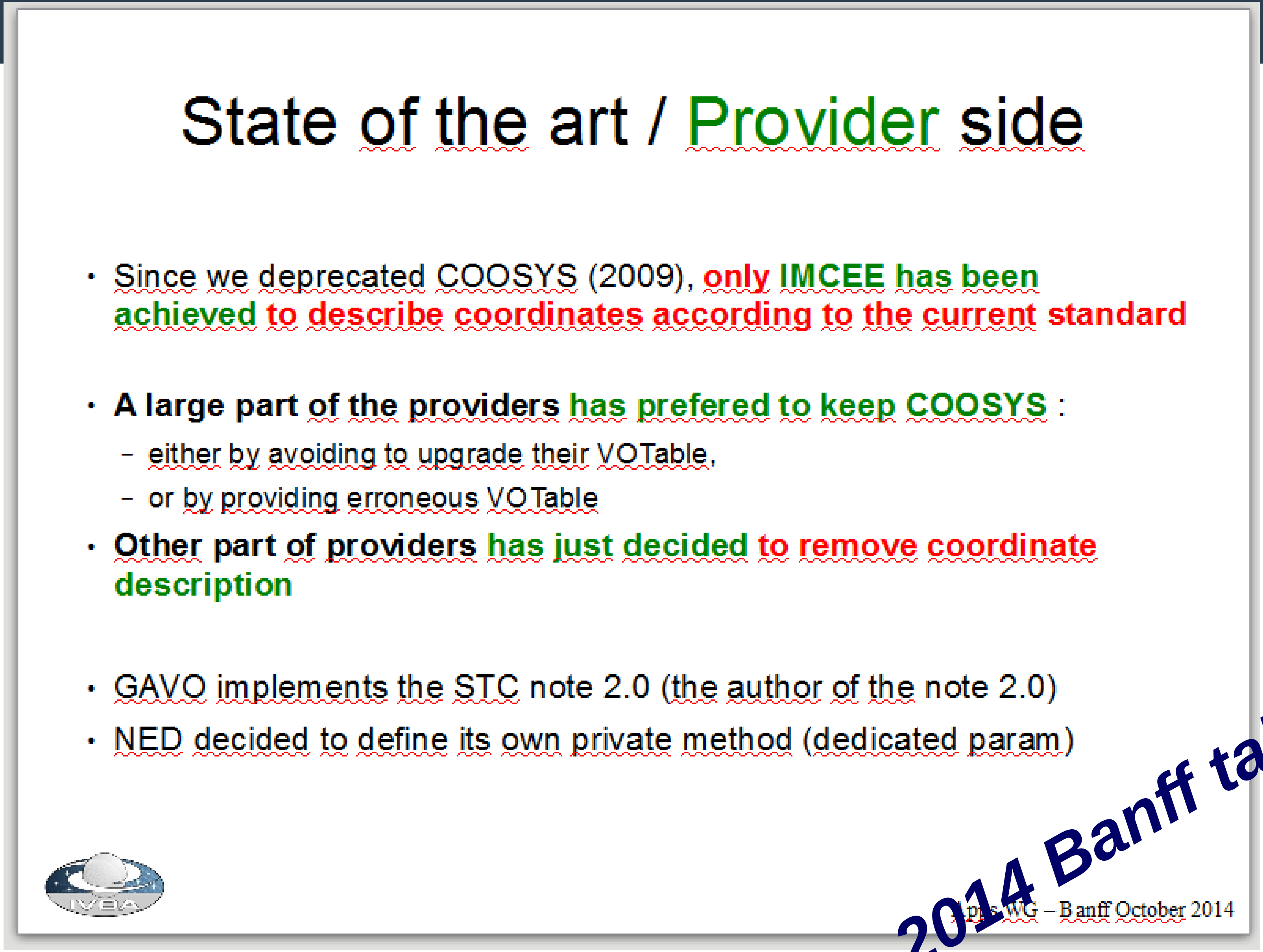
VOTable, HiPS & MOC

Pierre Fernique

VOTable

- First IVOA recommendation (2003)
- “THE” VO format used everywhere in VO
- 3 minor releases:
  - 1.1 (2004) → GROUP, FIELD/PARAMref, utype
  - 1.2 (2009) → xtype, COOSYS deprecated: alternate solution (by “STC in VOTable” note 1.1)
  - 1.3(2013) → BINARY2, new alternate COOSYS solution (new version of “STC in VOTable” note 2.0)
- The issue: Where are my coordinates ?

VOTable – The problem had been signaled !



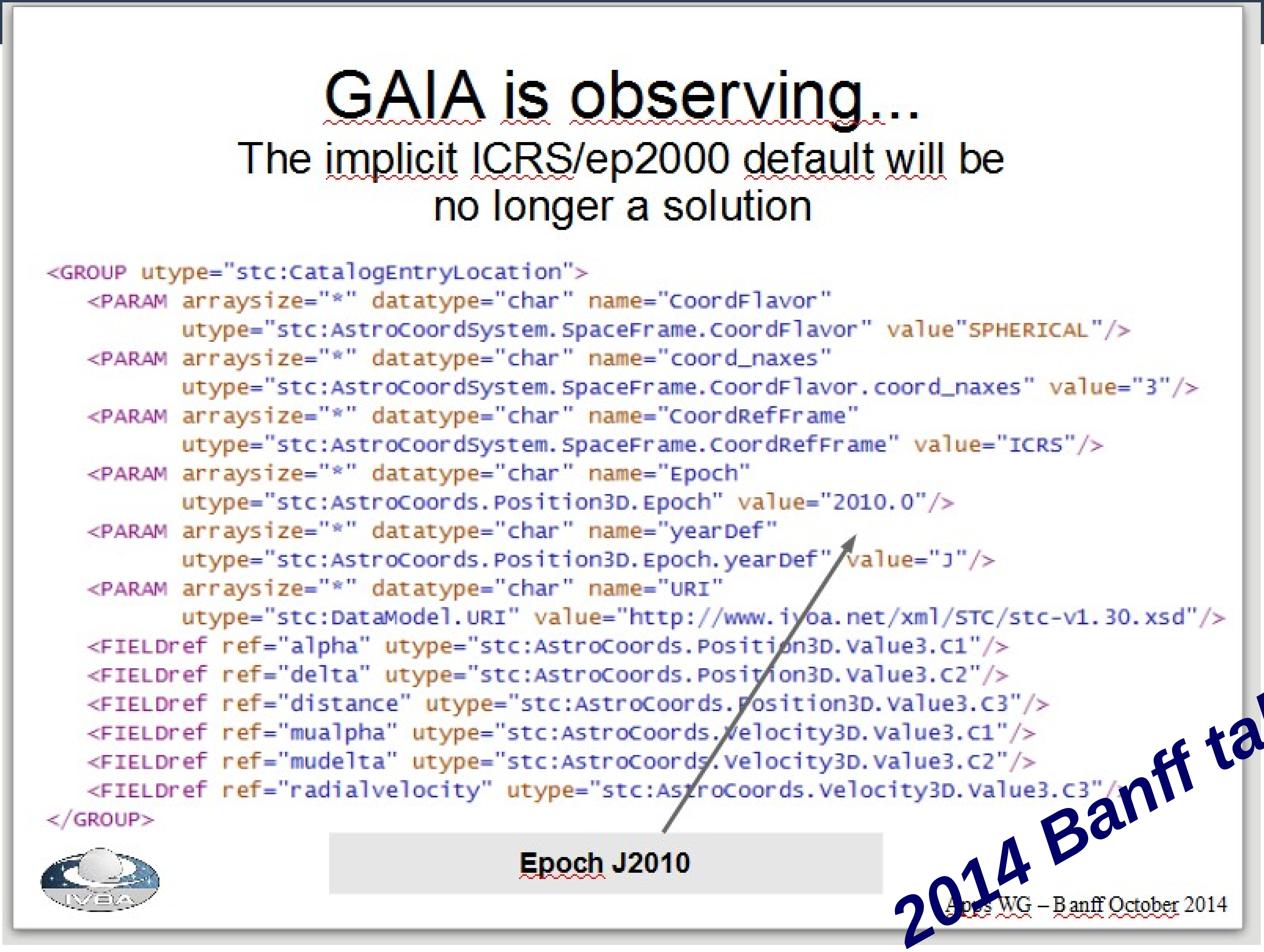
State of the art / Provider side

- Since we deprecated COOSYS (2009), **only IMCEE has been achieved to describe coordinates according to the current standard**
- **A large part of the providers has prefered to keep COOSYS** :
  - either by avoiding to upgrade their VOTable,
  - or by providing erroneous VOTable
- **Other part of providers has just decided to remove coordinate description**
- GAVO implements the STC note 2.0 (the author of the note 2.0)
- NED decided to define its own private method (dedicated param)

Apps WG – Banff October 2014

2014 Banff talk

VOTable – Gaia had been explicitly cited



GAIA is observing...

The implicit ICRS/ep2000 default will be no longer a solution

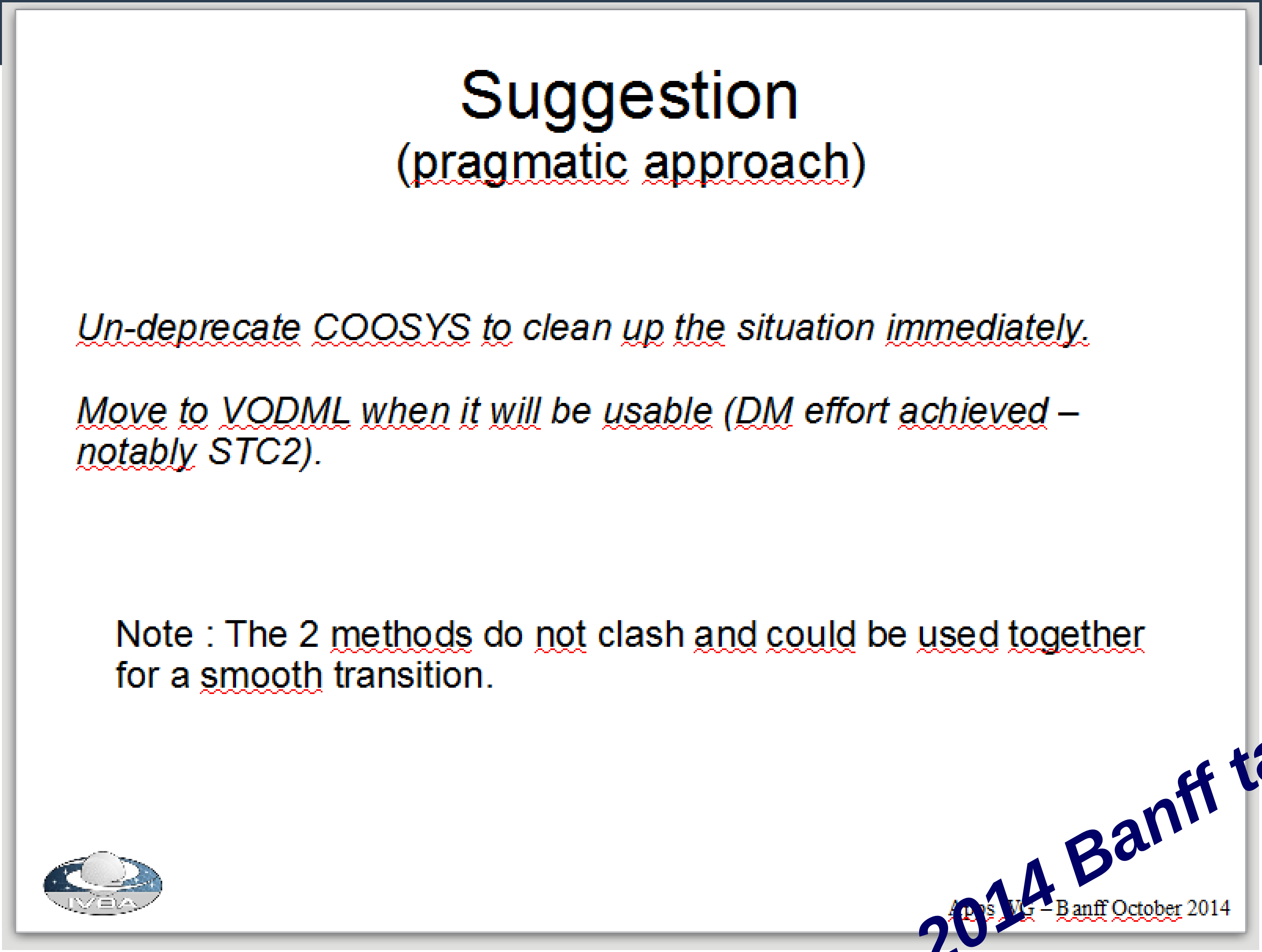
```
<GROUP utype="stc:CatalogEntryLocation">
   <PARAM arraysize="*" datatype="char" name="CoordFlavor"
          utype="stc:AstroCoordSystem.SpaceFrame.CoordFlavor" value"SPHERICAL"/>
   <PARAM arraysize="*" datatype="char" name="coord_naxes"
          utype="stc:AstroCoordSystem.SpaceFrame.CoordFlavor.coord_naxes" value="3"/>
   <PARAM arraysize="*" datatype="char" name="CoordRefFrame"
          utype="stc:AstroCoordSystem.SpaceFrame.CoordRefFrame" value="ICRS"/>
   <PARAM arraysize="*" datatype="char" name="Epoch"
          utype="stc:AstroCoords.Position3D.Epoch" value="2010.0"/>
   <PARAM arraysize="*" datatype="char" name="yearDef"
          utype="stc:AstroCoords.Position3D.Epoch.yearDef" value="J"/>
   <PARAM arraysize="*" datatype="char" name="URI"
          utype="stc:DataModel.URI" value="http://www.ivoa.net/xml/STC/stc-v1.30.xsd"/>
   <FIELDref ref="alpha" utype="stc:AstroCoords.Position3D.Value3.C1"/>
   <FIELDref ref="delta" utype="stc:AstroCoords.Position3D.Value3.C2"/>
   <FIELDref ref="distance" utype="stc:AstroCoords.Position3D.Value3.C3"/>
   <FIELDref ref="mualpha" utype="stc:AstroCoords.Velocity3D.Value3.C1"/>
   <FIELDref ref="mudelta" utype="stc:AstroCoords.Velocity3D.Value3.C2"/>
   <FIELDref ref="radialvelocity" utype="stc:AstroCoords.Velocity3D.Value3.C3"/>
</GROUP>
```

Epoch J2010

Apps WG – Banff October 2014

2014 Banff talk

VOTable – A solution had been adopted



Suggestion
(pragmatic approach)

Un-deprecate COOSYS to clean up the situation immediately.

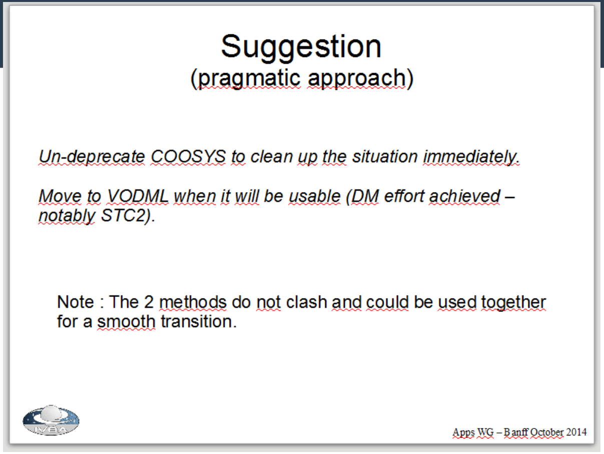
Move to VODML when it will be usable (DM effort achieved – notably STC2).

Note : The 2 methods do not clash and could be used together for a smooth transition.

Apps WG – Banff October 2014

2014 Banff talk

VOTable – But the situation is worst today


Suggestion
(pragmatic approach)

Un-deprecate COOSYS to clean up the situation immediately.

Move to VODML when it will be usable (DM effort achieved – notably STC2).

Note : The 2 methods do not clash and could be used together for a smooth transition.

Apps WG – Banff October 2014

- 2 years after, the VODML VOTable serialization is still in debate
- COOSYS has not been really readopted (reasons: just a mail announcement, TAP packages ? VOTable validators ?)

➡ We have a serious problem !

VOTable – The Apps chair's proposal 1

- **No chance to have a rapid solution from VO-DML+STC2 coordinate DM serialization**
- **In the meantime, re-enforce the pragmatic solution adopted in 2014:**
  - Write an short EN (endorsed note) for un-deprecating COOSYS “officially”
  - Adapt as fast as possible the TAP libs and VOTable validators according to
  - Convince providers to reuse COOSYS (notably Gaia providers)

VOTable – The Apps chair's proposal 2

- **VODML uses now dedicated <VODML>,<TYPE>,<ROLE> => there is no longer clash with the "STC in VOTable note" syntax (utype based).**
- **Standardize the principle of "STC in VOTable" for describing specifically coordinates:**
  - Adapt as fast as possible the TAP libs and VOTable validators according to
  - Convince providers to use this method (notably Gaia providers)
  - Write VOTable 1.4 according to (no longer reference to an external note)

VOTable – your point of views

HiPS – Hierarchical Progressive Survey

- First CDS demonstration (IVOA 2010)
- IVOA note (oct 2015) describing HiPS
- IVOA decision to standardize HiPS (nov 2015)
- IVOA Working Draft (june 2016)
- Already a great success:
  - 350+ HiPS, 12+ servers, 4 independent clients + derived clients, python and java HiPS toolkit...
- The question: what's the next step ?

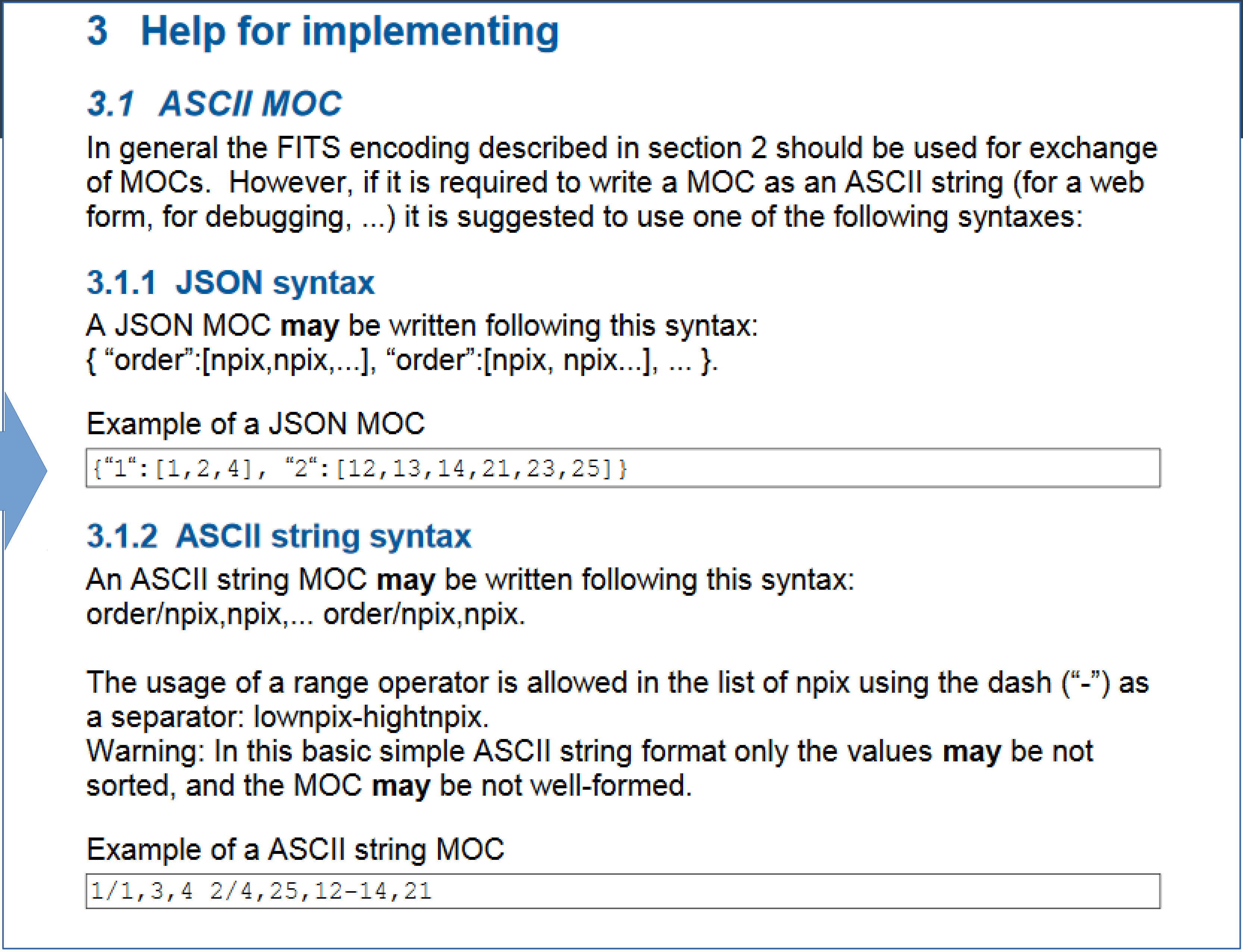
HiPS – App chair's proposal

- The HiPS author list is large and represents a good panel of data providers: CDS, JAXA, ESAC, MAST, CADC, ALMA
- All controversial points have been fixed (author's level, and external level)
- It seems that we are ready for the next step (PR)

HiPS – your point of views

MOC – MultiOrder Coverage map

- IVOA recommendation since 2 years (june 2014)
- Good success: more & more usages, libs, algos, and tools
- It has been adopted by developers as a generic tool for manipulating any kind of regions (even very accurate regions, observation footprints, spatial index, ...)
- One serialization: FITS (= binary table of HEALPix index)
- Altlernate JSON and ASCII serialization syntaxes just suggested



3 Help for implementing

3.1 *ASCII MOC*

In general the FITS encoding described in section 2 should be used for exchange of MOCs. However, if it is required to write a MOC as an ASCII string (for a web form, for debugging, ...) it is suggested to use one of the following syntaxes:

3.1.1 JSON syntax

A JSON MOC **may** be written following this syntax:
{ “order”:[npix,npix,...], “order”:[npix, npix...], ... }.

Example of a JSON MOC

```
{“1“:[1,2,4], “2“:[12,13,14,21,23,25]}
```

3.1.2 ASCII string syntax

An ASCII string MOC **may** be written following this syntax:
order/npix,npix,... order/npix,npix.

The usage of a range operator is allowed in the list of npix using the dash (“-”) as a separator: lownpix-hightnpix.
Warning: In this basic simple ASCII string format only the values **may** be not sorted, and the MOC **may** be not well-formed.

Example of a ASCII string MOC

```
1/1,3,4 2/4,25,12-14,21
```

MOC – The question ?

- Concretely: JSON MOC is more & more used
- The question: Is it required now to normalize this syntax in a MOC REC 1.1, or a EN (endorsed note) ?
- Pro
  - Stop the risk of divergences (various JSON implementations)
- Con
  - 2 serializations never help for interoperability

MOC – The App chair's point of view

- Presently there is no divergence
- The principle of "*suggested alternative syntaxes*" in the REC works fine: 1 REC format, but keeping the door open for specific alternatives if required.
- A new REC or EN is always a heavy process
- For me, no reason to change the MOC REC 1.0

MOC – your point of views