Datalink - SODA

Pierre Le Sidaner
Observatoire de Paris

C’est quoi ?

* **Un standard de 2015**

- **Fichiers multiples par dataset**
  - **Images différents format**
  - **Multiples preview (taille)**
- **SODA**
  - **Cutout, découpage cube**
  - **Conversion de format**
  - **Service “standard” SSA …**

Partie des protocoles d'accès aux données

LINK
VOSI-availability
VOSI-capabilities

Le service

- Un lien
- Un ID
  - l'identifiant peut être ce que l'on veut

Ou utiliser Datalink ?

- Très utile avec ObsTAP EPN-TAP.
  - Table unique donc un seul lien décrit
  - Accès à des services synchrones
  - Accès à des services associés SSA, SIA, SLAP

SODA

- Fait pour cutout, passage 3D→1D, chagmt format
- Idée sync et async

SODA

- Passage de paramètres :
  - ID obligatoire dans le service qui propose un datalink
  - Les paramètres : pos, band, time, pol

Le lien n'est pas vers le service mais ça description (service descriptor)

- **VOTable :**
  - **Lien sur le service**
  - **Description des paramètres avec UCD, type, unité …**

  possibilité de valeurs prédéfinies, de limites

Dans DaCHS cas EPN-TAP

```
<column name="ds_id" type="text"
tablehead="Dataset ID" description="Identifier for
this object to use when calling datalink services"
ucd="meta.id"/>

<column name="datalink_url" type="text"
tablehead="Datalink"
description="Datalinks and services for this object"
ucd="meta.ref.url;meta.datalink"/>

<var
key="ds_id">@target_class+"/"+@target_name</var
>
<var key="datalink_url">("\getConfig{web}
{serverURL}/\rdId/epdl/dlmeta" +"?
ID="+urllib.quote(@ds_id))</var>
```

Dans DaCHS cas EPN-TAP

```
<setup>
        <code>
          class PlanetDescriptor(object):
            def __init__(self, pubDID):
              self.pubDID = pubDID
              self.targetClass, self.targetName = self.pubDID.split("/", 1)
              self.computeEphemerisId()

            def computeEphemerisId(self):
                shortClass = {
                    'planet': 'p',
                    'satellite': 's',
                }[self.targetClass]
                self.ephId = "{}:{}".format(
                    shortClass, self.targetName)
            description="computed ephemeris of the planet",
        </code>
</setup>
```

Dans DaCHS cas EPN-TAP

```
<code>
    return PlanetDescriptor(pubDID)
</code>
    </descriptorGenerator>
<dataFunction>
    <setup>
      <par name="upstream_service_url"> "
http://vo.imcce.fr/webservices/miriade/ephemcc_query.php?
-from=vespa&amp;"</par>
 <code>
    import urllib
    from gavo.svcs import WebRedirect
 </code>
 </setup>
 <code>
    parameters = {
      "-name": descriptor.ephId,
      "-ep": args["epoch"],
      "-observer": args["observatory"],
       }
```

Dans DaCHS cas EPN-TAP

```
                raise
WebRedirect(upstream_service_url+urllib.urlencode(parameters))
            </code>
        </dataFunction>
    </datalinkCore>
```

CONCLUSION

- Uniquement si on en a besoin
- Tirer parti des frameworks/Librairies :
  - DaCHS, Mantelet/Saada , CADC …
- Manque encore de clients pour en tirer parti