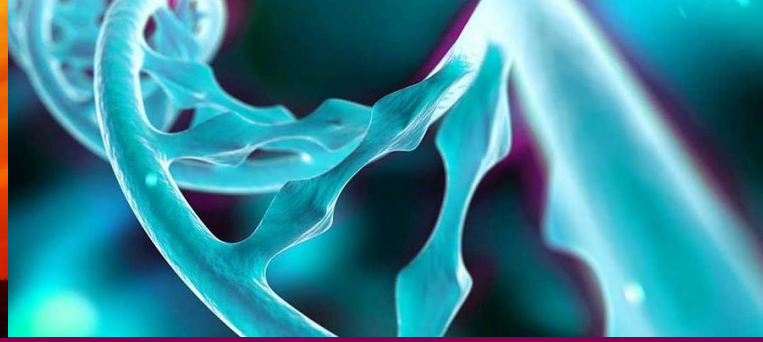




PAS-palveluiden aineistojen haku ja hallinta

12.6.2018 - Johan Kylander

CSC – Suomalainen tutkimuksen, koulutuksen, kulttuurin ja julkishallinnon ICT-osaamiskeskus

REST-rajapinta

- REST = Representational state transfer
- Arkkitehtuurimalli, perustuu HTTP-protokollaan
  - Pyyntö - vastaus
  - Resurssipohjainen, resurssit määritellään URI:na pyynnöissä
  - Hyperlinkkejä viesteissä
  - Vakiintuneita metodeita (GET, POST, PUT, DELETE)
- REST määrittele ominaisuuksia ja rajoitteita rajapinnoille, jotka ovat ”RESTful”

REST-yhtenäisen rajapinnan vaatimus

- REST on teknologiariippumaton, mutta asettaa RESTful-rajapinnalle seuraavat vaatimukset:
- REST on resurssipohjainen
  - Vs. toimintapohjainen
  - Pyyntöjen kohde on resurssi, esimerkiksi tietokanta, jota manipuloidaan HTTP-komennoilla
  - Pyynnöt koostuvat URI-jonoista, jotka yksilöivät resurssin
- REST-viestit kuvaavat itseään
  - esim. MIME-tyypeillä viestin otsikossa

REST-yhtenäisen rajapinnan vaatimus

- Dataa käsitellään representaationa
  - Esimerkiksi metatietoa JSON-muodossa
  - Ei riippuvuutta taustajärjestelmän teknologiasta
- REST-viestit ovat hypertekstiä
  - Viestit sisältävät otsikon, rungon, kyselyn, palautuskoodin, hyperlinkkejä
- Esimerkki: GET http://myrest/people/johankylander

RESTful-rajapinta on tilaton

- Tilattomassa arkkitehtuurissa palvelin ei tallenna dataa istunnoista
  - Pyynnön tulos ei ole riippuvainen edellisistä pyynnöistä
  - Pyyntö on itsenäinen ja tuottaa aina saman tuloksen
  - Asiakas hallinnoi tilaa, haluttu tila ilmaistaan viesteissä
- Jokainen viesti sisältää kaiken tarvittavan tiedon mitä palvelin tarvitsee voidakseen käsitellä viestiä
  - Ei riippuvuutta ulkoiseen tietoon
- Esimerkiksi GET http://myrest/people/johankylander/email

RESTful-rajapinta perustuu palvelin - asiakasmalliin

- Palvelin
  - Järjestelmä, joka sisältää resursseja, esim. tietokannan ja aineistoa
  - Kuuntelee pyyntöjä, vastaa niihin
- Asiakas
  - Järjestelmä, joka lähettää pyyntöjä palvelimelle
  - Ei ole jatkuvasti yhdistettynä palvelimeen
  - Pollaaminen, ei jatkuva kuunteleminen
  - Muutokset palvelimen koodiin ei yleensä vaikuta asiakkaiden rajapintoihin
- Keskustelu tapahtuu yhtenäisen rajapinnan vaatimuksien määrätyllä tavalla

RESTful-rajapinnan vaatimukset

- Lisäksi REST-rajapinnan viestit sopivat välimuistiin, mahdollistavat kerrostetun arkkitehtuurin ...

Eli:

- Asiakkaat lähettävät REST-rajapintaan kyselyitä, johon palvelin vastaa
- Vastaukset johtavat asiakkaita eteenpäin
  - Vastaukset sisältävät linkkejä eteenpäin, tai aineistoa
- Tilan muutos, esimerkiksi jakelupaketin muodostaminen, selvitetään pollaamalla palvelinta
- Asiakkaan ei tarvitse kuunnella PAS-palvelinta
  - Omaa palvelinta ei tarvita
  - Asiakas on aina aloitteentekijä keskustelussa
- Asiakkaan ei tarvitse tietää rajapinnan kaikkia ominaisuuksia etukäteen

Hyötyjä

- Tilattomuus mahdollistaa sisäänrakennetun virhekäsittelyn
  - Jos jokin ei toimi, lähetään pyyntö uudestaan
  - Pyynnöt ja pollaamiset eivät riko rajapintaa eivätkä vaikuta performanssiin
- PAS-palveluiden rajapinnat perustuvat lähinnä GET-pyyntöihin, jotka eivät muokkaa resursseja
- Poikkeuksena jakelupaketin luontipyyntö, joka on POST
- Muita metodeita ei ole käytössä

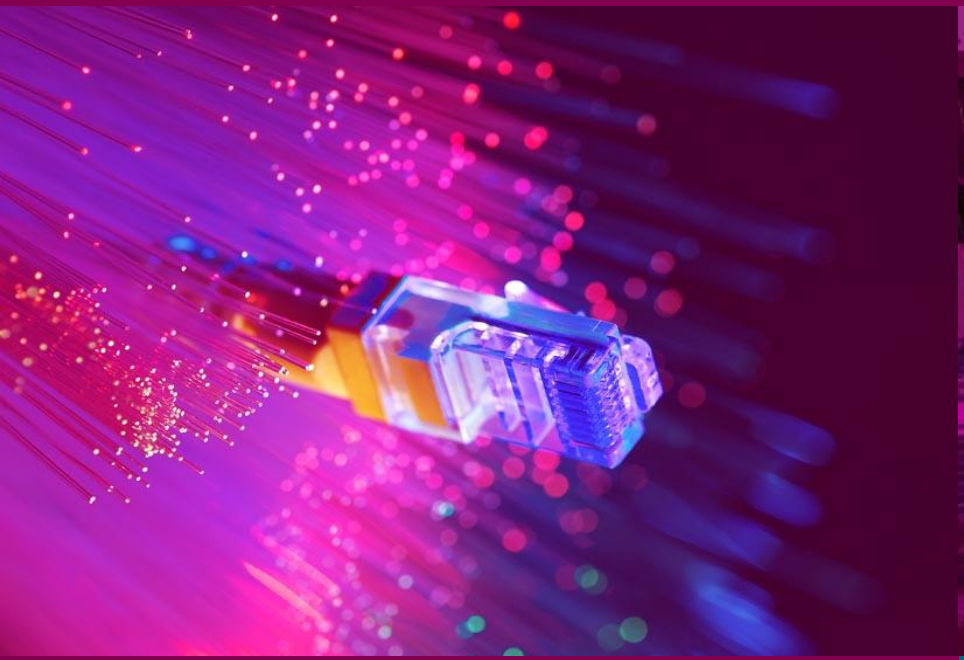
Hyötyjä

- Tilattomuus mahdollistaa automaattisen käsittelyn
  - Kaikki tarvittavat tiedot resurssien muokkaamiseen ja palauttamiseen sisältyvät viesteihin
  - Pyyntö ja vastaus liittyvät automaattisesti toisiinsa
  - Viestit sisältävät valmiiksi metatietoja otsikossa HTTP-protokollan myötä, kuten keskustelun osapuolet HTTPS-yhteydessä

REST

- REST on suosittu ja yleisesti käytössä koska:
  - Sitä on helppoa kehittää koska se perustuu olemassa olevaan protokollaan, HTTP
  - Se ei vaadi asiakasjärjestelmiltä erillistä jatkuvasti ylläpidettävää softaohjelmaa
  - HTTP-metodit (GET, POST jne.) ovat yksiselitteisiä
  - REST on joustava, koska jokainen resurssi on yksilöity
  - REST skaalautuu hyvin ja on tehokas viestinnässä koska sisältö ei ole riippuvainen käytetystä taustajärjestelmien teknologista (representaatio)
  - REST-rajapinta on helposti muokattavissa, laajennettavissa

HTTP-viestit






HTTP-viestien rakenne

- Viesti/statusrivi ilmaisee viestin
  - Metodi, URL ja palautuskoodi
- Otsikko kuvaa viestiä
  - Sisältää tietoja viestistä kuten viestin sisällön tyyppi, koko, aikaleima
  - Tiedot ilmaistaan avain-arvo-parina
- Tyhjä rivi
- Runko sisältää palautusviestin
  - Yleisiä formaatteja on JSON, HTML, XML
  - GET-pyyntö ei sisällä viestirunkoa

HTTP-viestien rakenne, esimerkki

```
HTTP/1.1 200 OK
Date: Mon, 23 May 2005 22:38:34 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 138
Last-Modified: Wed, 08 Jan 2003 23:11:55 GMT
Server: Apache/1.3.3.7 (Unix) (Red-Hat/Linux)
ETag: "3f80f-1b6-3e1cb03b"
Accept-Ranges: bytes
Connection: close

<html>
<head>
  <title>An Example Page</title>
</head>
<body>
  Hello World, this is a very simple HTML document.
</body>
</html>
```

HTTP metodeja

- CRUD-pohjaisia metodeita resurssien manipulointiin
- GET – resurssin haku, pidetään yleensä harmittomana metodina koska se ei muuta resurssia
- POST – tietojen lähettäminen, päivittäminen, palveluun
- Lisäksi DELETE, PUT, HEAD, OPTIONS jne.

HTTP – yleisiä vastauskoodeja

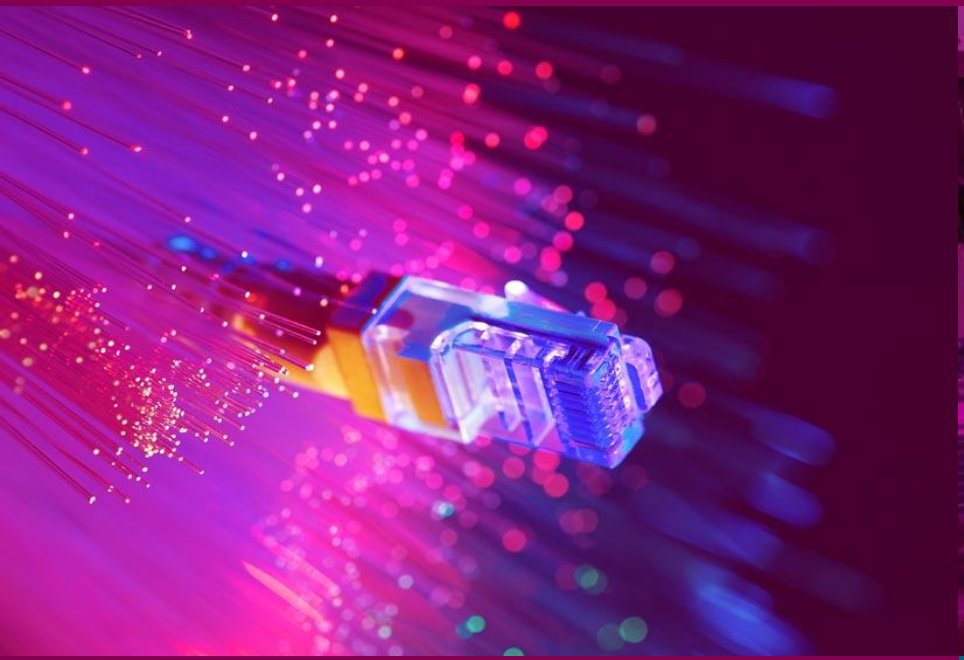
- Viestit sisältävät kolminumeroisen koodin, joka kertoo miten pyyntöä on vastaanotettu ja käsitelty
  - 1xx-alkuiset koodit – palvelin on vastaanottanut viestin
  - 2xx-alkuiset koodit – onnistunut pyyntö
    - 200 – OK, 202 – hyväksytty
  - 3xx-alkuiset koodit – edelleenohjaus
  - 4xx-alkuiset koodit – virhetilanne
    - 400 – vääränlainen pyyntö, 401 – ei auktorisoitu pyyntö, 404 – resurssia ei löydy, 405 – väärä metodi
  - 5xx-alkuiset koodit – palvelinvirhe (palvelu tai sen osa on alhaalla)

PAS-palveluiden rajapinnan viestit ovat JSON-muodossa

- JSON on kompakti ja avoin standardi tiedonvälitykseen
- Syntaksi perustuu avain-arvo-pareille
  - Tietueet erotellaan aaltosulkeilla
  - Avain-arvo-parit erotellaan pilkulla
  - Listat ympyröidään hakasulkeilla
- Esimerkki:

```
{
 ”etunimi”: ”Johan”,
 ”sukunimi”: ”Kylander”,
 ”tarvikkeet”: [”tietokone”, ”hiiri”, ”kännykkä”]
}
```

PAS-palveluiden REST-rajapinta – määritykset





REST-rajapinnan toiminnot

- PAS-palveluiden REST-rajapinnan kautta asiakas voi hakea, hallinnoida ja noutaa omaa aineistoaan
- Haku:
  - Haun avulla asiakas löytää aineistonsa, jota hän haluaa hallinnoida
- Hallinnointi:
  - Asiakas pystyy muodostamaan aineistostaan jakelupaketin
- Noutaminen:
  - Asiakas voi ladata muodostettuja jakelupaketteja

PAS-palveluiden REST-rajapinnan kyselyt

- Resurssi yksilöidään pyynnön URIlla, joka muodostetaan kauttaviivoilla erotetuilla määreillä
- Monille pyynnöille on mahdollista antaa tarkentavia parametreja avain-arvo-pareina
  - ? –merkki ilmaisee parametrien osuuden pyynnöstä
  - & -merkki käytetään parametrien erotinmerkkinä
  - Esimerkki http://myrest/people?find=all&limit=100

PAS-palveluiden REST-rajapintakyselyiden muoto

- PAS-palveluiden REST-rajapinnan kyselyt ovat muotoa:

{GET, POST} <base>/<contract>/<term> …

- Kyselyissä:
  - <base> ilmaisee REST-rajapinnan
  - <contract> ilmaisee sopimustunnuksen, joka toimii kyselyiden aineiston rajauksena
  - <term> ilmaisee resurssin tai työkalun

PAS-palveluiden REST-rajapinnan kyselyiden muoto

{GET, POST} **<base>**/<contract>/<term> …

- <base> on URL-merkkijono, osoite, joka on ohjaa kyselyt PAS-palveluiden rajapinnalle:
  - https://pas.csc.fi/api/2.0
- Hyväksytyt metodit kyselyviesteille ovat GET (useimmat kyselyt) ja POST (jakelupaketin muodostus)

Sopimustunnus

{GET, POST} <base>/**<contract>**/<term> ...

- <contract> on aineiston sopimustunnus
  - Aineistoa haetaan ja hallinnoidaan REST-rajapinnassa sopimustunnuksen, ei käyttäjätunnuksen, perusteella
  - Jokainen säilytyksessä oleva AIP on osa aineistoa, jota määritellään PAS-palvelusopimuksissa
  - Sopimuksen ja hyödyntävän organisaation suhde ei ole 1:1
  - Hyödyntäville organisaatioille ilmoitetaan heidän sopimuksensa tunnukset
  - Sopimustunnus, contractID, on pakollinen metatieto PAS-skeemakatalogin versiosta 1.7.0 lähtien
  - Kaikille ”vanhoille” AIPeille on lisätty sopimustunnus hakuindeksissä

PAS-palveluiden REST-rajapinnan resurssit

{GET, POST} <base>/<contract>/**<term>** ...

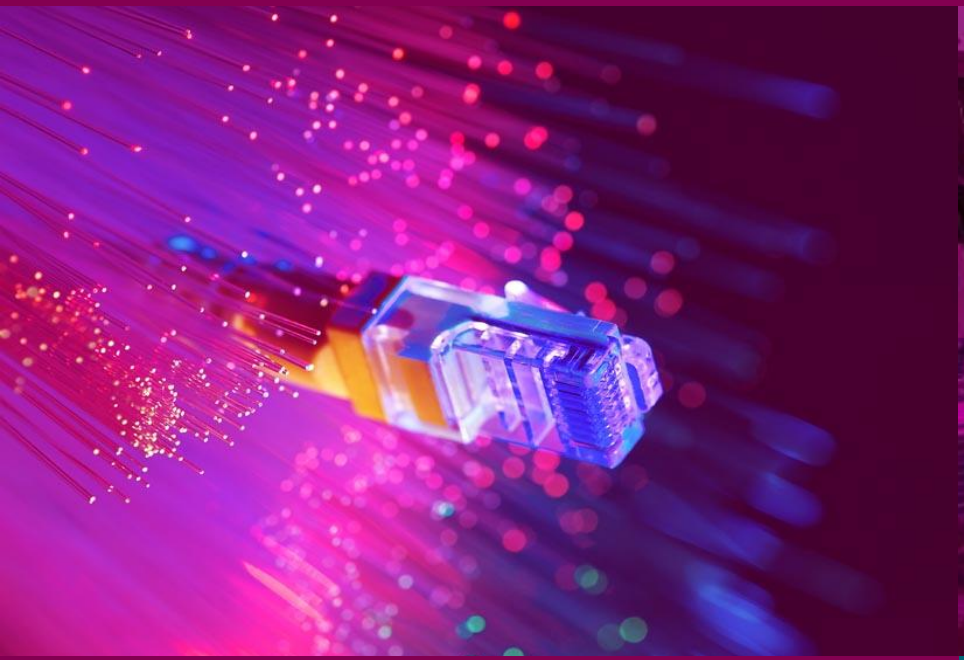
- <term> on olemassa oleva resurssi tai työkalu:
  - search – aineiston haun työkalu
  - preserved – säilytyspaketin resurssi
  - disseminated – jakelupaketin resurssi
- Esimerkki:
  - https://pas.csc.fi/api/2.0/urn:minunsopimukseni/search

PAS-palveluiden REST-rajapinnan palautusviestit ovat JSONia

```
{
  ”status”: ”success”,
  ”data”: { ”message”: ”…” }
}
```

- status – mahdolliset arvot: ”success” ja ”fail”
- message – sisältää palautusviestin

Autentikointi ja auktorisointi



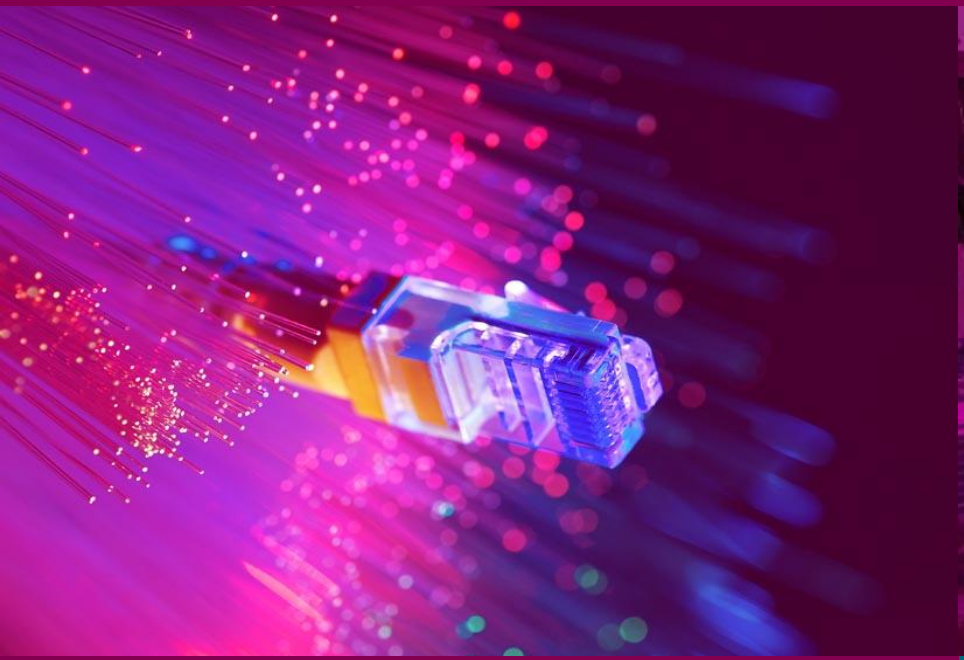
REST-rajapinta on avoin vain PAS-palveluiden käyttäjille

- Autentikointi = käyttäjän tunnistautuminen
- PAS-palveluiden REST-rajapinnassa käytetään TSL/SSL-suojausta ja HTTP Basic Access –autentikointia
  - Vain rekisteröityneille käyttäjille on pääsy REST-rajapintaan
  - IP-rajattu pääsy
  - Vaaditaan käyttäjätunnusta ja salasanaa REST-rajapinnan pyynnöille

REST-rajapinnan kautta on pääsy vain omalle aineistolle

- Auktorisointi= käyttäjän oikeuksien hallinta resursseihin
- PAS-palvelussa tarkastetaan jokaisen pyynnön yhteydessä onko kyseisellä käyttäjällä oikeus hakea kyseisellä sopimustunnuksella aineistoa
  - Pyynnöt hyväksytään vain jos käyttäjällä on oikeuksia sopimustunnukselle
  - Pyynnöissä tarkastetaan kuuluuko haettu säilytyspaketti kyseiselle sopimustunnukselle
- Poikkeuksena jakelupaketin julkisen allekirjoitusavaimen nouto

Aineiston hakutyökalu



Aineiston haku

GET <base>/<contract>/**search**?<parameters>

- Aineistoa pystyy hakemaan metatietotietokannasta METS dokumentin sisällön perusteella
- Hakuindeksinä toimii Solr-tietokanta
  - Myös Finna käyttää Solria
- Tyhjä haku (ei parametreja) palauttaa kaiken kyseisen sopimustunnuksen alaiset paketit luettelona

Aineiston haku

GET <base>/<contract>/search?**<parameters>**

- Pyynnölle pystyy antamaan valinnaisia parametreja, joista "q" ("query") ilmaisee hakuehdon
- Esimerkki:
  - https://pas.csc.fi/api/2.0/mycontract/search?q=title:MyTitle

Aineiston haku – kyselyt metatietotietokantaan

GET <base>/<contract>/search?**q=XXX:XXX**

- Kyselyt ovat muotoa q=<hakukenttä>:<arvo>
  - Hakukenttä on METS-dokumentin elementin nimi
  - Mahdollisuus antaa koko XML-polku haun tarkkaan rajaukseen
  - Apache Lucene-syntaksi käytettävissä
- Esimerkkejä:
  - q=subject:food
  - q=OBJID:mets1
  - q=mets_dmdSec_mdWrap_xmlData_subject:food

Aineiston haku – palautusviestit

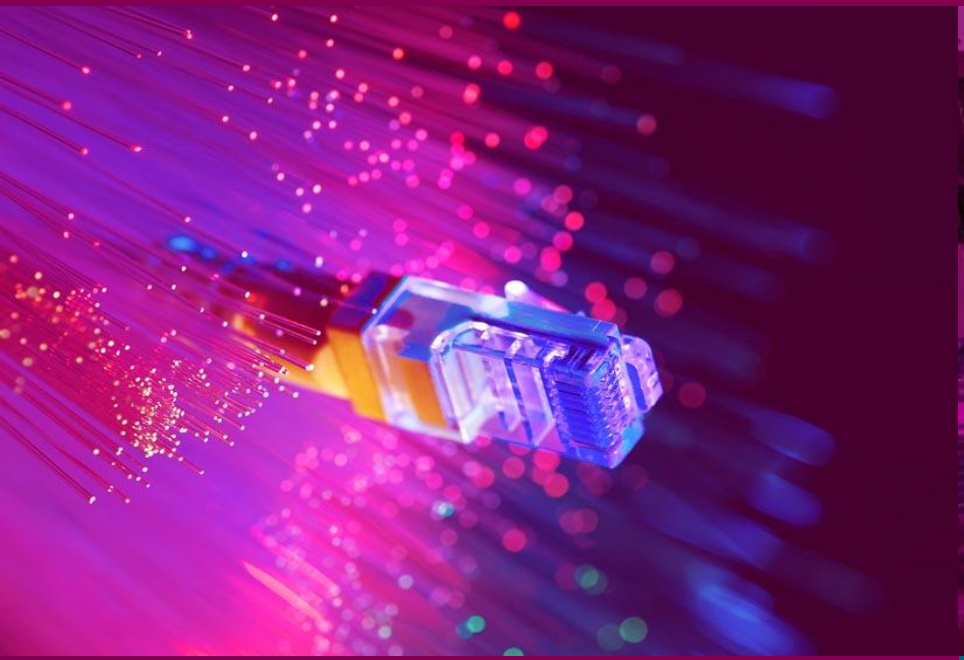

```
{
    "data": {
        ”links": {
            "self": "/api/2.0/mycontract/search?q=OBJID%3Ab82112ce-f3ea-4789-afd7-43651a374bof&limit=20&page=1"
        },
        "results": [
            {
                "createdate": "2018-04-10T11:48:20Z",
                ”id": "aip-001",
                "location": "/api/2.0/mycontract/preserved/aip-001",
                "match": {
                    "mets_OBJID": [ ”b82112ce-f3ea-4789-afd7-43651a374bof” ]
                },
                "pkg_type": "AIP”
            }
        ]
    },
    ”status": "success”
}
```


<- Paketin tunniste

<- Osoite aineiston hallintaan

<- Hakuehdon osuma

Säilytyspakettien resurssi



Säilytyspakettiin kohdistuvat komennot

GET <base>/<contract>/**preserved**/<aip-id>

- Haku palauttaa halutun aineistokokonaisuuden ja linkit sitä vastaavien säilytyspakettien hallintaan
- Säilytyspaketteja hallinnoidaan preserved-termillä
  - <aip-id> on pakollinen määre joka yksilöi säilytyspaketin

Säilytyspakettiin kohdistuvat komennot

- Palautusviestissä ilmoitetaan säilytyspaketille kohdistuvat komennot linkkinä
  - Tällä hetkellä vain jakelupaketin muodostus

```
{
        ”status”: ”success”,
        ”data”: {
              ”disseminate”: ” <base>/<contract>/preserved/<aip-id>/disseminate”
        }
}
```

Säilytyspaketin muodostus jakelupaketiksi

POST <base>/<contract>/preserved/<aip-id>/**disseminate?**<parameters>

- Säilytyspaketista muodostetaan jakelupaketti POST-komennolla
- Valinnaiset parametrit ovat:
  - catalog – haluttu skeemakatalogin versio, johon jakelupaketti muodostetaan
  - format – haluttu jakelupaketin formaatti

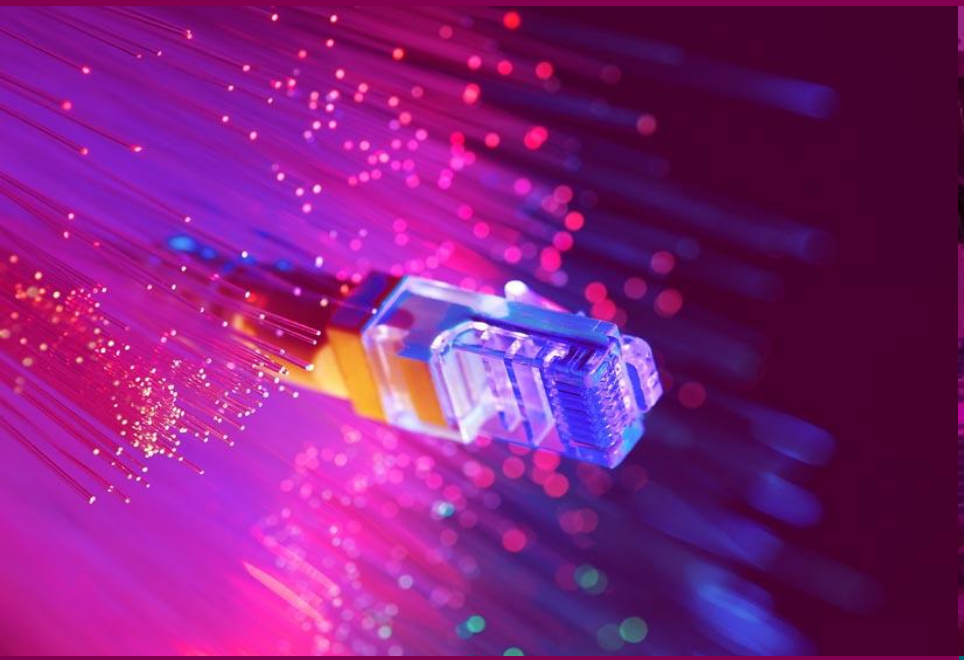
Jakelupaketin muodostus

- Palautusviestissä ilmoitetaan jakelupaketin hallinnan osoite

```
{
        ”status”: ”success”,
        ”data”: {
              ”disseminated”: ” <base>/<contract>/disseminated/<dip-id>”
        }
}
```

- <dip-id> on muodostetun jakelupaketin tunniste

Jakelupaketit



Jakelupaketin seuranta ja niihin kohdistuvat komennot

GET <base>/<contract>/**disseminated**/<dip-id>

- Komennolla seurataan jakelupaketin muodostusta
- Palautusviestissä ilmoitetaan jakelupaketin muodostuksen status sekä jakelupakettiin kohdistuvat komennot
- Komento on tulos edeltävästä jakelupaketin muodostus-komennosta

Jakelupaketin muodostamisen seuranta

- Palautusviesti on seuraava

```
{
    "status": "success",
    "data": {
        "complete": <complete>,
        "actions": <actions>
    }
}
```

  - complete ilmaisee jakelupaketin muodostumisen statusta, arvo voi olla joko "true" (jakelupaketti ob muodostettu) tai "false" (jakelupaketin muodostus on käynnissä)
  - actions ilmaisee jakelupaketille kohdistuvat komennot
  - Mikäli jakelupakettin muodostus on kesken, actions on tyhjä { }

Jakelupakettiin kohdistuvat komennot

- Muodostuneen jakelupaketin palautusviesti on seuraava

```
{
    ”status”: ”success”,
    ”data”: {
        ”complete”: ”true”,
        ”actions”: {
            ”download”: "<base>/<contract>/disseminated/<dip-id>/download“
            ”metadata”: "<base>/<contract>/disseminated/<dip-id>/metadata“
            ”history”: "<base>/<contract>/disseminated/<dip-id>/history"
        }
    }
}
```

Jakelupaketin noutaminen

GET <base>/<contract>/disseminated/<dip-id>/**download**

- Komennolla noudetaan jakelupaketti
- Jakelupaketti annetaan palautusviestissä
- Muita komentoja on metatietojen ja tapahtumahistorian noutaminen

Allekirjoitusavaimen noutaminen

GET <base>/**public_key/dip**

- Komennolla noudetaan jakelupaketin allekirjoitusavaimen julkinen osa
- Avaintiedosto annetaan palautusviestissä

REST-rajapinnan komentoketju

- Komennot muodostavat katkeamattoman ketjun
- Koko ketjua ei ole tarpeen käydä läpi mikäli tarvittava osoite on tiedossa
- Esimerkiksi:
  - <base>/<contract>/preserved/<aip-id>/disseminate komennon voi antaa heti mikäli säilytyspaketin tunniste on tiedossa
  - <base>/<contract>/disseminated/<dip-id>/download komennon voi antaa heti mikäli jakelupaketti on olemassa
- Tämä on mahdollista koska REST on tilaton

REST-rajapinnan määritykset

- PAS-palveluiden REST-rajapinnan määritykset:
- http://digitalpreservation.fi/specifications/PAS-rajapinnat-2.0.2.pdf


facebook.com/CSCfi


twitter.com/CSCfi

youtube.com/CSCfi


linkedin.com/company/csc---it-center-for-science


github.com/CSCfi