EGI Repositories

Description about the EGI repositories

- Architecture
- Requirements
  - Data Structure
  - Json Schemas
  - GPG Sign RPMs
- Documentation
  - Deployment of new releases
  - Add new product to an existing distribution
  - Add new distribution

Architecture

Requirements

Data Structure

NEW implementation

- **name** : Software name
- **version** : version of the software name
- **contact** : contact name of the software
- **technicalContact**: technical contact name of the developper
- **description** : Software shor descripion
- **releaseDate** : Date of the release deployed in UMD/CMD etc.
- **majorVersion** : Major version of the release
- **minorVersion** : Minor version of the release
- **revisionVersion** : Revision version of the release
- **releaseNotes** : Descriptiopn or link to the relase notess
- **changeLog** : Change log description or link to change log
- **documentationLinks** : Documentation link for the software
- **distributionType** ": [retrieved from distribution list]
- **distributionVersion**: [retrieved from distribution distributed version list]
- ~~**MajordistributionVersion**: [retrieved from distribution distributed version list]~~
- **capabilities** : [retrieved from capabilities definition list]
- **target**:
  ["platform": "centos7"];
  ["arch": "x86_64"];
  [repositoryURL": "sw/production/umd/4/centos7/x86_64/updates]
- **softwareConfigurationLink** : link for the UMD/CMD software configuration
- ~~**softwareConfigurationManagement** : link for the UMD/CMD software configuration~~

target

platform

- centos 7
- centos 8
- centos 9
- ubuntu 18
- ubuntu 20
- Docker

arch

- X86_64

Capabilities

(list of possible capabilities)

- **Accounting**
- **Authorization**
- **Attribute Authority**
- **Client Tools**
- **Credential Management**
- **Data Access**
- **File Access, File Transfer**
- **Interactive Job Management**
- **Information Model**
- **Information Discovery**
- **Job Execution**
- **Metadata Catalogue**
- **Parallel Job**
- **Storage Management**

distributionType

- **UMD**
- **CMD-OS**
- **CMD-ONE**
- **Community**

MajordistributionVersion

- 1
- 2
- 3
- 4
- 5

OLD

- **currentState** : State of the "Production",

- **additionalDetails** : Wiki link for Knon issues
- **repositoryURL** : "sw/production/umd/4/centos7/x86_64/updates",
- **keywords** :

Json Schemas

Product


```
{
"name" : "htcondor",
"version" : "9.0.1",
"currentState" : "Production",
"technologyProvider" : "CHTC (UW-Madison)",
"contact" : "jfrey.at.cs.wisc.edu",
"technicalContact" : "jfrey.at.cs.wisc.edu",
"description" : "  HTCondor is a specialized workload management system for compute-intensive jobs. Like other full-featured batch systems, HTCondor provides a job queueing mechanism, scheduling policy, priority scheme, resource monitoring, and resource management. Users submit their serial or parallel jobs to HTCondor, HTCondor places them into a queue, chooses when and where to run the jobs based upon a policy, carefully monitors their progress, and ultimately informs the user upon completion.
",
"releaseDate" : "2021-06-01",
"majorVersion" : "9",
"minorVersion" : "0",
"revisionVersion" : "1",
"releaseNotes" : "https://htcondor.readthedocs.io/en/v9_0/",
"changeLog" : "N/A",
"documentationLinks" : ["http://research.cs.wisc.edu/htcondor/manual/","link2"],
"repositoryURL" : "sw/production/umd/4/centos7/x86_64/updates",
"distributionType": "UMD",
"distributionVersion": "4.15.0",
"capabilities": ["File Access", "Workload Management System"],
"targets": [
   {
      "platform": "centos7",
      "arch": "x86_64",
      "repositoryURL": "sw/production/umd/4/centos7/x86_64/updates",
      "gpgkey": "http://repository.egi.eu/sw/production/umd/UMD-RPM-PGP-KEY",
```

```
    "rpms": [
        "https://research.cs.wisc.edu/htcondor/repo/9.0/el7/release/condor-9.0.1-1.el7.x86_64.rpm",
            "https://research.cs.wisc.edu/htcondor/repo/9.0/el7/release/blahp-2.0.1-1.el7.x86_64.rpm",
            "https://research.cs.wisc.edu/htcondor/repo/9.0/el7/release/condor-all-9.0.1-1.el7.x86_64.rpm",
            "https://research.cs.wisc.edu/htcondor/repo/9.0/el7/release/condor-annex-ec2-9.0.1-1.el7.x86_64.rpm",
            "https://research.cs.wisc.edu/htcondor/repo/9.0/el7/release/condor-boinc-7.16.16-1.el7.x86_64.rpm",
            "https://research.cs.wisc.edu/htcondor/repo/9.0/el7/release/condor-bosco-9.0.1-1.el7.x86_64.rpm",
            "https://research.cs.wisc.edu/htcondor/repo/9.0/el7/release/condor-classads-9.0.1-1.el7.x86_64.rpm",
            "https://research.cs.wisc.edu/htcondor/repo/9.0/el7/release/condor-classads-devel-9.0.1-
1.el7.x86_64.rpm",
            "https://research.cs.wisc.edu/htcondor/repo/9.0/el7/release/condor-credmon-oauth-9.0.1-
1.el7.x86_64.rpm",
            "https://research.cs.wisc.edu/htcondor/repo/9.0/el7/release/condor-devel-9.0.1-1.el7.x86_64.rpm",
            "https://research.cs.wisc.edu/htcondor/repo/9.0/el7/release/condor-externals-9.0.1-1.el7.x86_64.rpm",
            "https://research.cs.wisc.edu/htcondor/repo/9.0/el7/release/condor-kbdd-9.0.1-1.el7.x86_64.rpm",
            "https://research.cs.wisc.edu/htcondor/repo/9.0/el7/release/condor-procd-9.0.1-1.el7.x86_64.rpm",
            "https://research.cs.wisc.edu/htcondor/repo/9.0/el7/release/condor-tarball-9.0.1-1.el7.x86_64.rpm",
            "https://research.cs.wisc.edu/htcondor/repo/9.0/el7/release/condor-test-9.0.1-1.el7.x86_64.rpm",
            "https://research.cs.wisc.edu/htcondor/repo/9.0/el7/release/condor-vm-gahp-9.0.1-1.el7.x86_64.rpm",
            "https://research.cs.wisc.edu/htcondor/repo/9.0/el7/release/python2-condor-9.0.1-1.el7.x86_64.rpm",
            "https://research.cs.wisc.edu/htcondor/repo/9.0/el7/release/python3-condor-9.0.1-1.el7.x86_64.rpm",
            "https://repo.opensciencegrid.org/osg/3.5/el7/release/x86_64/python2-scitokens-1.3.1-
1.osg35.el7.noarch.rpm"
        ]
    }
],
"softwareConfigurationLink": ""
}
```


- **name** : Software name
- **version** : version of the software name
- **contact** : contact name of the software
- **technicalContact**: technical contact name of the developper
- **description** : Software shor descripion
- **releaseDate** : Date of the release deployed in UMD/CMD etc.
- **majorVersion** : Major version of the release
- **minorVersion** : Minor version of the release
- **revisionVersion** : Revision version of the release
- **releaseNotes** : Descriptiopn or link to the relase notess
- **changeLog** : Change log description or link to change log
- **documentationLinks** : Documentation link for the software

- **distributionType** : [retrieved from distribution list]
- **distributionVersion**: [retrieved from distribution distributed version list]
- **capabilities** : [retrieved from capabilities definition list]
- **target**:
  ["platform": "centos7"
  "arch": "x86_64"
  "repositoryURL": "sw/production/umd/4/centos7/x86_64/updates
  "gpgkey": ""
  "rpms": [
    "https://...1",
    "https://...n"
  ]
  ]
- **softwareConfigurationManagement** : link for the UMD/CMD software configuration

Distributions


```
{
   "title": "Channel: EGI Repository - Production level releases",
   "description": "This is the EGI Repository Releases RSS channel.",
   "item": [
      {
         "distributionIdentifier": "UMD-5",
         "description": "UMD 5 is a new release of UMD...",
□□□"currentState": "Production"
      },
□□{
         "distributionIdentifier": "UMD-4.15.0",
         "description": "UMD 4.15.0 is an update release of UMD 4. It includes StoRM 1.11.21, lcmaps-plugins 1.8.1, CERN Frontier 4.15.2, dmlite 1.15.0, APEL SSM 3.2.1, DDNS Nagios probe 1.0.1, dCache 6.2.24, Infrastructure Manager Nagios probe 1.0.1",
□□□"currentState": "Production"
      },
      {
         "distributionIdentifier": "UMD-4.14.0",
         "description": "UMD 4.14.10 is an update release of UMD 4. It includes ARC 6.12.0, xrootd 5.2.0, xrootd 5.2.0, davix 0.7.6, gridftp 13.24.1, gfal2 2.18.2",
□□□"currentState": "Production"
      }
   ]
```

```
}
```

If we need to add dates to help user control the info:

```
"pubdate": "2020-11-26 11:44:25 EEST",
"updated": "2020-11-26 11:45:02 EEST",
```

Software Catalog

```
{
   "title": "Channel: EGI Repository - Production level releases",
   "description": "Here you can find the complete list of EGI Software products.",
   "item": [
      {
         "name" : "nagios-probe-im",
         "version" : "1.0.2",
         "distroIdentifier": "UMD-5",
□□□"capabilities": ["client tools", "monitoring"],
         "description" : "probe for Infrastructure Manager service to ARGO-SAM release",
         "pubdate": "2021-12-02"
      },
□□{
         "name" : "nagios-probe-im",
         "version" : "1.0.1",
         "distroIdentifier": "UMD-4.15.0",
      □"capabilities": ["client tools", "monitoring"],
         "description" : "probe for Infrastructure Manager service to ARGO-SAM release",
         "pubdate": "2021-07-21"
      }
□□,{
□□□...
□□}
□]
}
```

Other info

target

platform

- centos 7
- centos 8
- centos 9
- ubuntu 18
- ubuntu 20
- Docker

arch

- X86_64

Capabilities

- **Accounting**
- **Authorization**
- **Attribute Authority**
- **Client Tools**
- **Credential Management**
- **Data Access**
- **File Access,File Transfer**
- **Interactive Job Management**
- **Information Model**
- **Information Discovery**
- **Job Execution**
- **Metadata Catalogue**
- **Parallel Job**
- **Storage Management**

distributionType

- **UMD**
- **CMD-OS**
- **CMD-ONE**
- **Community**

MajordistributionVersion

- 1
- 2
- 3
- 4
- 5

GPG Sign RPMs

GPG key generation

A gpg key was generated with the following parameters:

```
gpg --full-generate-key

Please select what kind of key you want:
   (1) RSA and RSA (default)
Your selection? 1
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (3072) 4096
Please specify how long the key should be valid.
         0 = key does not expire
Key is valid for? (0) 0

Real name: RPM sign UMD/CMD
Email address: grid.admin@lip.pt
Comment:
You selected this USER-ID:
    "RPM sign UMD/CMD <grid.admin@lip.pt>"
public and secret key created and signed.
pub   rsa4096 2022-02-21 [SC]
      FDA5272E8C8A03597BFA253511339AA2D60A5E99
uid                      RPM sign UMD/CMD <grid.admin@lip.pt>
sub   rsa4096 2022-02-21 [E]
```

The passphrase is in monica (under NCG site).

The public and private keys where exported:

```
gpg --export -a 'RPM sign UMD/CMD'  > RPM-GPG-KEY-umd-cmd
gpg --export-secret-key 'RPM sign UMD/CMD' > RPM-GPG-KEY-umd-cmd.key
```

Copy the keys to monica, and import them:

```
gpg --import RPM-GPG-KEY-umd-cmd
gpg --import --allow-secret-key-import RPM-GPG-KEY-umd-cmd.key
```

On the host that will sign the RPMs:

```
rpm --import RPM-GPG-KEY-umd-cmd
```

To list all rpm gpg keys:

```
rpm -q gpg-pubkey --qf '%{name}-%{version}-%{release} --> %{summary}\n'
```

Documentation

Deployment of new releases

Process to deploy a new release manually:

- 1.**Create the json files using one of the following options:**
  - 1.1. **yaml templates** - this option needs to use yaml2json converter to generate json file
  - 1.2. **json templates** - use the json SW/templates/distribution.template.json or product.template.json
- 2. **Prepare distribution and product**
  - **a)** If it's a product from a **new distribution** we need to add the distribution first:
    - ***add-new-distribution*** using the templates in SW/templates/distribution.template.json
      - Install/Copy the template into json_dir/distributions/distribution-name-version.json
    - ***add new product*** to the distribution file json_dir/distributions/distribution-name-version.json products array
      (example of a json product item to add to the products array)

      ```
      { "name" : "emi.apel", "display" : "APEL publisher", "version" : "3.2.7", "apicallback" : "/static/SW/APEL publisher3.2.7.json", "target" : [ { "platform" : "sl5", "arch" : "x86_64", "repositoryURL" : "sw/production/umd/1/sl5/x86_64/base" } ] }
      ```

  - **b)** If it's a **product from an existing distribution** we need to:
    - ***add "productname.json" full data || "name"=name+version + capabilities=[] to the distribution file***
      - json_dir/distributions/distribution-name-version.json
    - ***add product to the catalog***
      - json_dir/software-catalog.json
    - ***Finally add product.json to the respective directory and create the release.json manually***
      - json_dir/products/distribution-name/distribution-version/
      - and add the same product file to the root of the json_dir/ with the name release.json

Add new product to an existing distribution

Ways to add a new product to an existing distribution (Work in progress)

dir structure:

```
/var/www/be/
data/  (The latest json data for the FE)      
├── config.js
├── distributions/
│   ├── CMD-ONE
│   │   ├── 0.0.0.json
│   │   └── ...
│   ├── CMD-OS
│   │   ├── 0.0.0.json
│   │   └── ...
│   ├── UMD
│   │   ├── 1.0.0.json
│   │   └── ...
├── products/
│   ├── CMD-ONE
│   │   ├── 0.0.0/
│   │   │   └── productname.json
│   │   │   └── ...
│   ├── CMD-OS
│   │   ├── 0.0.0/
│   │   │   └── productname.json
│   │   │   └── ...
│   ├── UMD
│   │   ├── 1.0.0/
│   │   │   └── productname.json
│   │   │   └── ...
├── software-catalog.json  (Complete list of products)
├── software-distributions.json  (Complete list of distributions)
│
templates/ (All the templates needed to convert yaml to json or using json directly)
├── product.yaml
├── distribution.yaml
```

```
├── product.json
└── distribution.json
```

For both approaches the first step is always convert/prepare the release.json

```
cd /var/www/be
```

Copy the template:

```
cp SW/templates/product.yaml  ./release.yaml
```

Convert the yaml to json:

```
yaml2json release.yaml (this will create a release.json file in the same dir)
```

Now we have a json file:

```
be$ release.json
```

1st approach - using the Database (DB)

Add the data manually to the DB:

- Product
- Distribution product data Commands...
  Examples:

```
cat create_table.sql |sqlite3 db/egi-repos-fe-test.db

sqlite3 db/egi-repos-fe-test.db ".read insert_data.sql"

sqlite3 db/egi-repos-fe-test.db "select MAX(f1) from tbl2;" ".exit"
sqlite3 db/egi-repos-fe-test.db "INSERT INTO tbl2 ('f1','f2','f3') VALUES('001','teste','ttttt');" ".exit"
```

```
INSERT INTO "main"."products" ("id", "name", "productName", "version", "majorVersion", "minorVersion", "revision
', 'deployed', 'parak@cesnet.cz', 'parak@cesnet.cz', 'sw/production/cmd-one/1', 'CESNET', 'CESNET', '2018-11-12 1
```

Export/prepare all the data from DB (replacing the data in data/*)

```
$ export-full-dataset.sh
```

.....
php export-distributions.php

php export-distributions.php --individual-distributions
php export-products.php
php export-products.php --individual-products

2nd approach - add the data manually to the json files

Manually add product

```
cp release.json data/
cp release.json data/products/distribution-type/distribution-version/$productname(change this).json
```

add product data (to the products list in distribution):
data/distributions/distribution-type/$distribution-version.json (products[] array)
ex. of product item:

```
 {
    "productId": 32,
    "name": "emi.apel",
    "version": "3.2.7",
    "capabilities": ["Accounting"],
    "status": 1
  }
```

add product full data (to the software catalog):
data/software-catalog.json (item[] array)
ex. of product item: (DON'T FORGET TO ADD A COMMA , BEFORE THE BRACKET)

```
,
{
 "name": "bdii.site-bdii",
 "version": "1.2.1",
 "documentationLinks": [],
 "target": [
  {
   "platform": "centos7",
   "arch": "x86_64",
   "repositoryURL": "sw\\/production\\/cmd-one\\/1\\/centos7\\/x86_64\\/updates",
   "gpgkey": "",
   "rpms": [],
   "status": 1
  }
 ],
 "capabilities": ["Client Tools"],
 "currentState": "deployed",
 "technologyProvider": "BDII",
```

```
  "contact": "email of contact",
  "technicalContact": "email of contact",
  "description": "text description...",
  "releaseDate": "2018-11-12 12:34:40",
  "majorVersion": 1,
  "minorVersion": 2,
  "revisionVersion": 1,
  "releaseNotes": "http://gridinfo.web.cern.ch/developers/resource-bdii",
  "changeLog": "Same version as in SL6\n",
  "repositoryURL": "sw/production/cmd-one/1",
  "distributionType": "CMD-ONE",
  "distributionVersion": "1.1.0",
  "softwareConfigurationLink": ""
}
```

Add new distribution

Ways to add a new distribution (Work in progress)

```
cd /var/www/be
```

Create distribution folder:

```
mkdir data/distributions/%distribution-name%
```

Use/Copy the template:

```
cp SW/templates/distribution.yaml  ./
```

Convert the yaml to json: (need to have node.js installed)

```
yaml2json distribution.yaml (this will create a distribution.json file in the same dir)
```

Now we have a json file:

```
distribution.json
```

Rename it to the distribution-name:

```
mv distribution.json %distribution-version%.json
mv %distribution-version%.json data/distributions/%distribution-name%/
```

1st approach - using the Database (DB)

Add the data manually to the DB:

- Product

- Distribution product data Commands...
  Examples:

```
cat create_table.sql |sqlite3 db/egi-repos-fe-test.db

sqlite3 db/egi-repos-fe-test.db ".read insert_data.sql"

sqlite3 db/egi-repos-fe-test.db "select MAX(f1) from tbl2;" ".exit"
sqlite3 db/egi-repos-fe-test.db "INSERT INTO tbl2 ('f1','f2','f3') VALUES('001','teste','ttttt');" ".exit"
```

```
INSERT INTO `distributions`(
        distributionType, distributionVersion, currentState,
        majorVersion, minorVersion, revisionVersion,
        releaseDate, created,
        lastUpdated, description,
        repositoryURL, releaseNotes,
        additionalDetails, installationNotes,
        contact, technicalContact,
        knownIssues, changeLog
        )
VALUES ('','','','','','','','','','','','','','','','');
```

Export/prepare all the data from DB (replacing the data in data/*)

```
$ export-full-dataset.sh
```

.....
php export-distributions.php
php export-distributions.php --individual-distributions (Each distribution) php export-products.php
php export-products.php --individual-products (Each product)

2nd approach - add the data manually to the json files

Manually add distribution

add distribution data (to the software distributions catalog):
data/software-distributions.json (item[] array)
ex. of distribution item: (DON'T FORGET TO ADD A COMMA , BEFORE THE BRACKETS)

```
,
{
  "distributionType": "",
  "distributionVersion": "%version%0.0.0",
  "currentState": "deployed",
```

```
  "releaseDate": "",
  "description": "",
  "products": [
      {
        "name": "product-name",
        "version": "0.0.0",
        "capabilities": ["Client Tools"],
        "status": 1
      }
    ]
}
```