Exploitation Platforms Open Architecture

Salvatore Pinto

16/03/2016
ESA “earth Science Application Exploitation Platform” project, for the building and pre-operations of platforms to foster exploitation of Earth Observation and earth science related data

- **6 Thematic Exploitation Platforms** sub-projects (hydrology, forestry, polar, urban, costal, geo-hazard)
- **Open Architecture**, harmonised among the EPs
- Open interface **standards and profiles**
- Resources **interoperability**, between EPs
- Common Core Components (**Open Source Framework**, with the aim to maximise reuse of existing Open Source project)
What is an Exploitation Platform?

For eSAEP, an Exploitation Platform is a virtual workspace, with the final aim to bring the users to the data.

The Exploitation platform is providing a user community with:

- **large volume and variety of data** (EO and non-space data)
- an algorithm development and **integration environment**
- **processing services** and software (e.g. toolboxes, retrieval baselines, visualization routines)
- **computing resources** (e.g. hybrid cloud/grid)
- **collaboration tools** (e.g. forums, wiki, knowledge base, open publications, social networking...)
- general operation capabilities (e.g. user management and access control, accounting, billing, etc...).
**eSAEP EP Concept**

- **Data Provider**
- **Data**
- **Processing services**
- **Service integration**
- **Scientific User**
- **Service Provider**
- **Final User**
- **Collaboration Tools**
- **Computing resources**
The **Exploitation Platform Open Architecture** definition is an on-going activity, building on experience from the TEP and related projects. Currently, the **DRAFT2** has been released with Creative Commons Attribution-Share Alike 4.0 International License.

DRAFT2 available at:  
https://tep.eo.esa.int/webdav/guest/document_library/Public/Exploitation%20Platforms%20Open%20Architecture%20DRAFT2
Key points in EP Architecture Design

1. User friendly (for all functionalities, including software integration, etc...)
2. Clear separation between key functionalities (processing, data search, data access, ...)
3. Interoperability and standards
4. ICT infrastructure independence
5. Cost-effective ICT provisioning
6. Common authentication framework (e.g. EO-SSO)
7. Pay-per-use (accounting, quota management, sponsorship)
8. Open Source software, open Standards
Key points in EP Architecture Design

1. User friendly (for all functionalities, including software integration, etc...)

- Pre-defined processing (Workflow)
- Interactive processing (App)
- Integration environment (SDK & tools)
Key points in EP Architecture Design

1. User friendly (for all functionalities, including software integration, etc...)

2. Clear separation between key functionalities (processing, data search, data access, ...)

3. Interoperability and standards
Key points in EP Architecture Design

1. User friendly (for all functionalities, including software integration, etc...)
2. Clear separation between key functionalities (processing, data search, data access, ...)
3. Interoperability and standards
4. ICT infrastructure independence

**ICT resources are decoupled from the platform**, via virtualization technologies, cloud technologies and generic API support.
1. User friendly (for all functionalities, including software integration, etc.)

2. Clear separation between key functionalities (processing, data search, data access)

3. Interoperability and standards

4. ICT infrastructure independence

5. Cost-effective ICT provisioning

Cost Effectiveness Target:

\[
\frac{\text{Resources spent for Scientific Processing}}{\text{Total amount of resources spent}} \rightarrow 1
\]

**Key points in EP Architecture Design**

- Use small-size packages
- Use shared resources
- Pre-install processors packages
- Cache the data
Key points in EP Architecture Design

1. User friendly (for all functionalities, including software integration, etc...)
2. Clear separation between key functionalities (processing, data search, data access, ...)
3. Interoperability and standards
4. ICT infrastructure independence
5. Cost-effective ICT provisioning
6. Common authentication framework (e.g. EO-SSO)
7. Pay-per-use (accounting, quota management, sponsorship)
8. Open Source Software, open standards
EP Architecture Macro-components
### Key components of an EP

According to the current EP Architecture draft, the key common core user-facing components of an Exploitation Platform are listed in the table (with description and related functionality):

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Catalogue</td>
<td><strong>Resource search</strong> interface, for user to look for platform resources (eg. data, results, processing services), according to metadata, classification, ranking, etc...</td>
</tr>
<tr>
<td>Resource Access Gateway</td>
<td>Manage the <strong>Resources access</strong>, for the platform. Controls users access to binary resources representation (e.g. input EO data products) and provides access interface.</td>
</tr>
<tr>
<td>Execution Gateway</td>
<td>Provides the <strong>Processing service execution</strong> interface for submission of processing requests.</td>
</tr>
<tr>
<td>Workflow/App Manager</td>
<td>Imports, interprets and executes a service represented by a <strong>Processing service package</strong>.</td>
</tr>
<tr>
<td>Geo Resource Browser</td>
<td>Provides the possibility to display all resources involved in a processing (eg. data, results, processing services), package them into <strong>Processing Containers</strong> and replicate processing</td>
</tr>
<tr>
<td>AAI</td>
<td>Manage the <strong>Authentication ad Authorization Infrastructure</strong> for the users of the platform.</td>
</tr>
</tbody>
</table>
Full components list

Workspace: Provide functionalities to the user to manage processing services

Collaboration Bucket: provides a framework for sharing unvalidated processing services, processing results, etc.

Accounting and monitoring: provides a central integrated accounting and monitoring system

Execution Cluster: Provides the environment and ICT resources where processing services are executed (for normal operations or testing)

Resource Ingestion: provides the Resource ingestion interface into the platform catalogue, extracting metadata and other required information
For the EP, in conjunction with HMA/OGC, we plan to evolve the existing standards to address the new EP, Big Data and Web 2.0 requirements.

The current baseline for this evolutions is:

<table>
<thead>
<tr>
<th>Implementation BP</th>
<th>Based on</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resource search</td>
<td>OGC OpenSearch standards and CEOS BPs</td>
</tr>
<tr>
<td>Resource access</td>
<td>OGC Download Services for EO</td>
</tr>
<tr>
<td>Processing service execution</td>
<td>OGC WPS</td>
</tr>
<tr>
<td>Processing service packaging</td>
<td>BPEL/XPDL/Docker/RPM</td>
</tr>
<tr>
<td>Processing container</td>
<td>OGC OWS Context</td>
</tr>
<tr>
<td>AAI</td>
<td>OGC User Management Interfaces for EO</td>
</tr>
</tbody>
</table>
Key points for the standards selection in the EP:

- **Web/developer oriented API**
- M2M communication
- Simple client
- Efficient encoding

Which translates to preference to the following technologies:

- **JSON over XML**
- **REST over SOAP**
- Pre-defined namespace vs custom namespaces

Moreover, standards shall contain Best Practices for Implementation, with:

- Clear definition of **namespaces, nomenclature and conventions**
- Definition of End-to-end Scenario involving the interface/product
- **Samples for all messages exchange** in the end-to-end scenario
Future activities

- **EP Open Architecture evolution**: evolve the architecture with lessons learnt from the thematic TEPs parallel architectural design activities.

- **EP standardization**: sponsor new standards or evolution to EP-related standards, profiles and best practices to the relevant standardization bodies (e.g. OGC, OGC testbeds).

Thank you

Contacts:

salvatore.pinto@esa.int

The ESA TEPs:

tepcoreteam@esa.int

https://twitter.com/esa_teps