Centre de Calcul de l'Institut National de Physique Nucléaire et de Physique des Particules

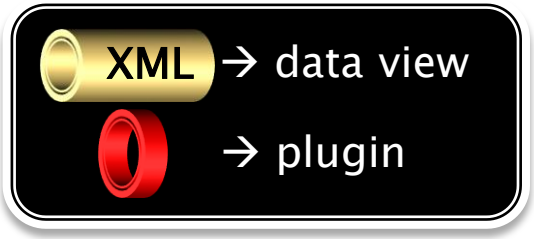
Nothing is lost, nothing is created,
everything is <xml:tranformed>

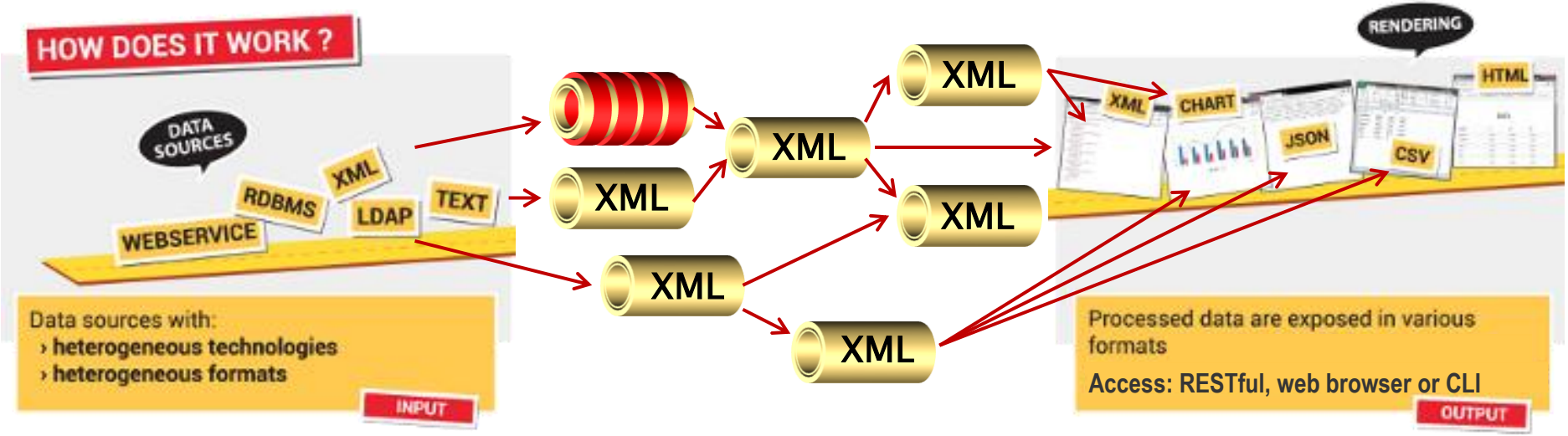
LAVOISIER

DATA AGGREGATION FRAMEWORK

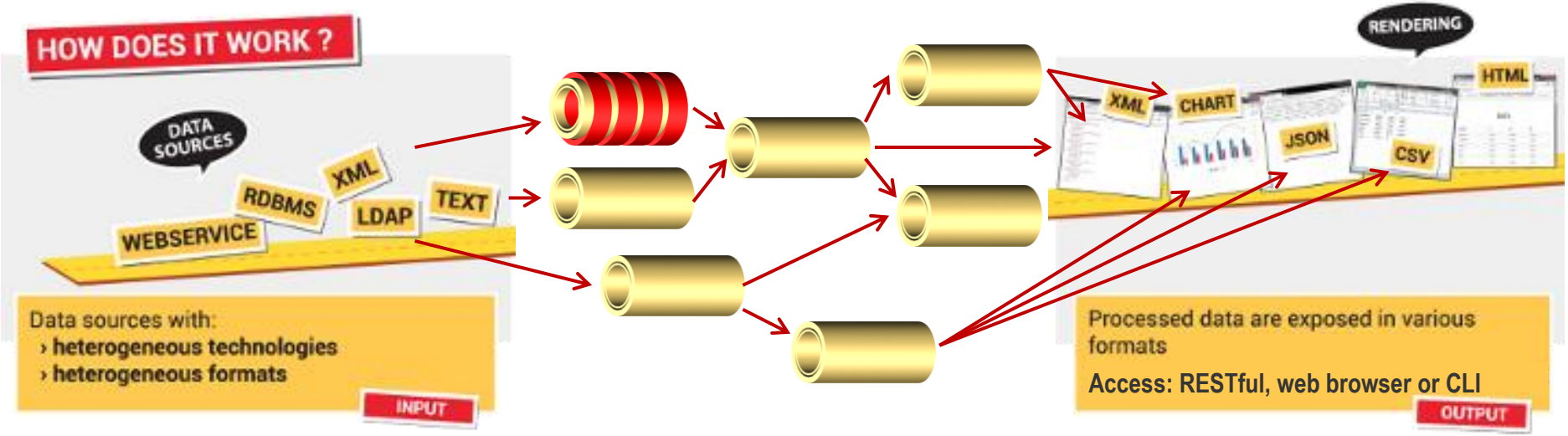
CCIN2P3

What is Lavoisier ?

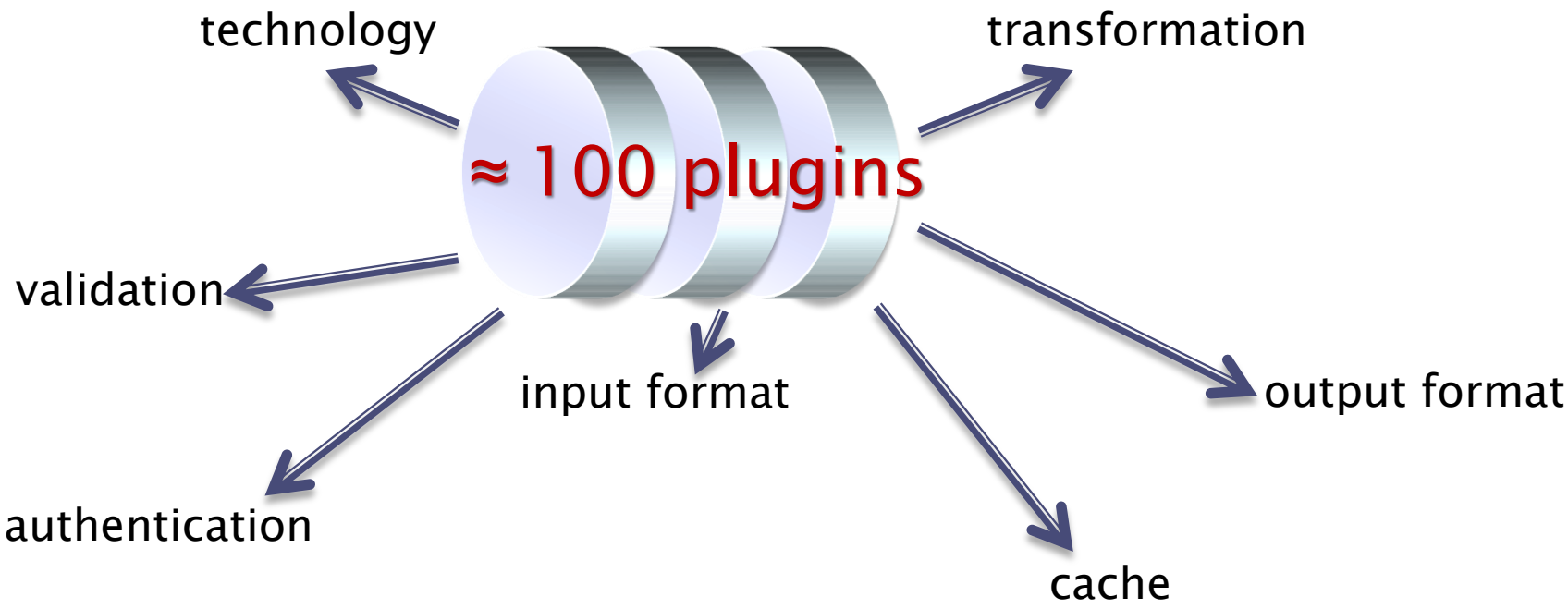




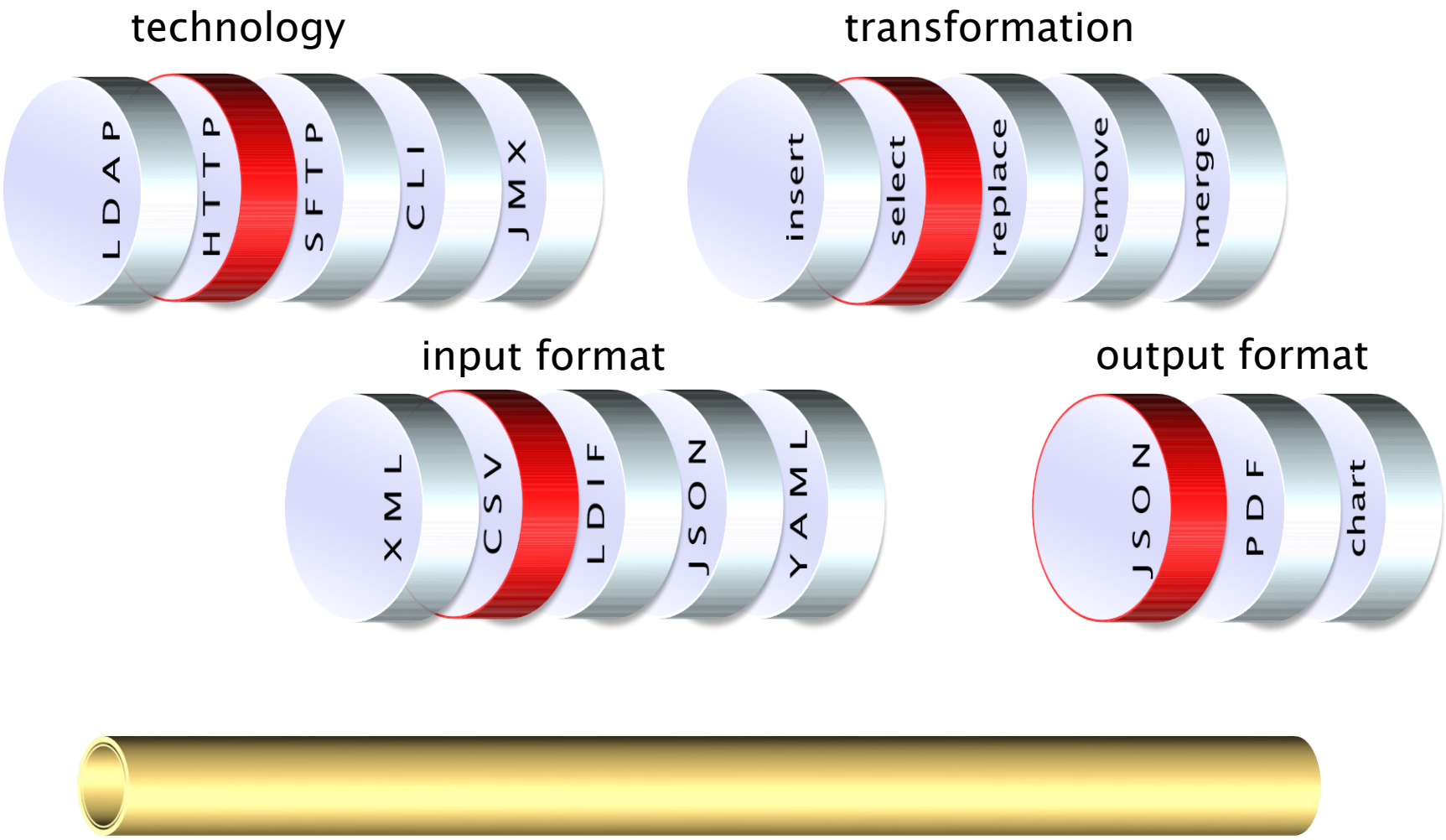
What is Lavoisier ?



What is a data view ?

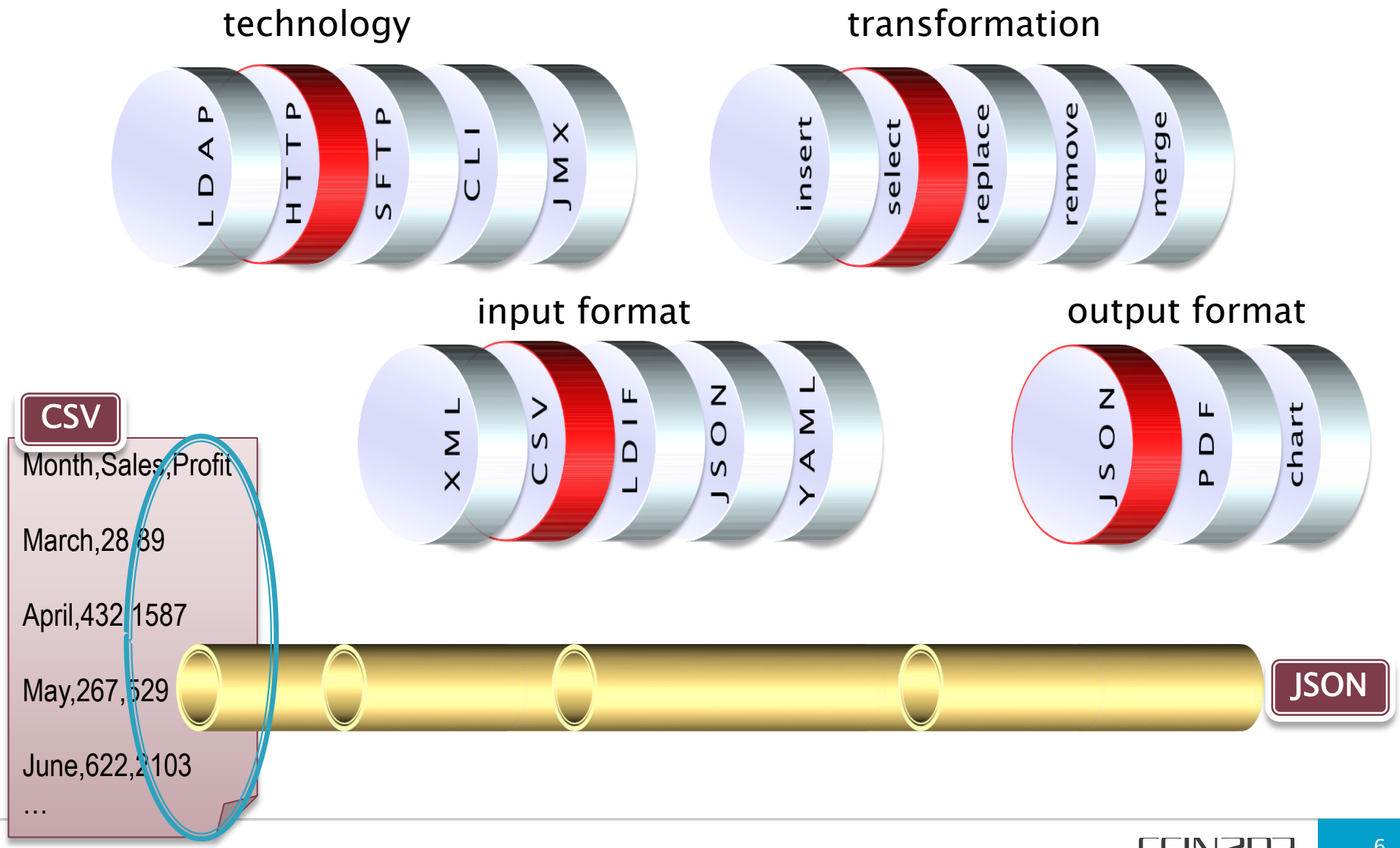


What is a data view ?



What is a data view ?

Example : download CSV data, extract column "Profit" and render to JSON



What is a data view ?

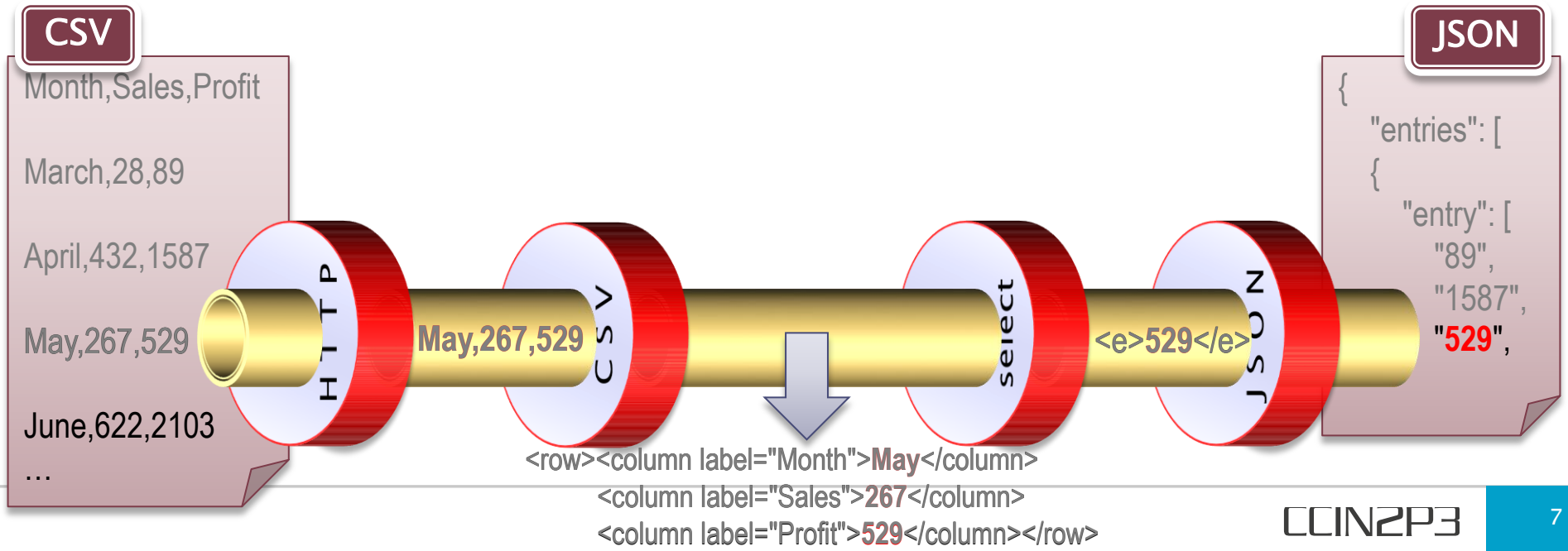
Example : download CSV data, extract column "Profit" and render to JSON

technology

transformation

input format

output format



Example : download CSV data, extract column "Profit" and render to JSON

profit

ccsyre7:8080/lavoisier/debug/profit

LAVOISIER | Monitoring | Configuration | Environment | Documentation

HTTPConnector

url	1 `concat('http://software.in2p3.fr/lavoisier/',$file)`

CSVSerializer

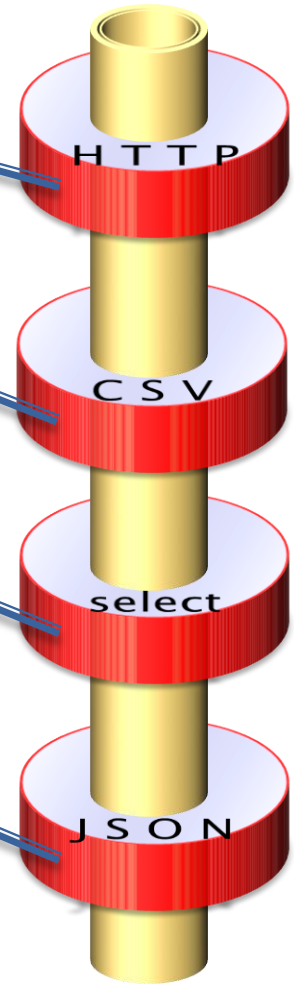
header	true

```
<rendering xmlns="http://software.in2p3.fr/lavoisier/rendering.xsd">
    <rows>
        <column_labels>
            <column_label label="Month">Month</column_label>
            <column_label label="Sales">Sales</column_label>
            <column_label label="Profit">Profit</column_label>
        </column_labels>
        <row>
            <column label="Month">March</column>
            <column label="Sales">28</column>
            <column label="Profit">89</column>
```

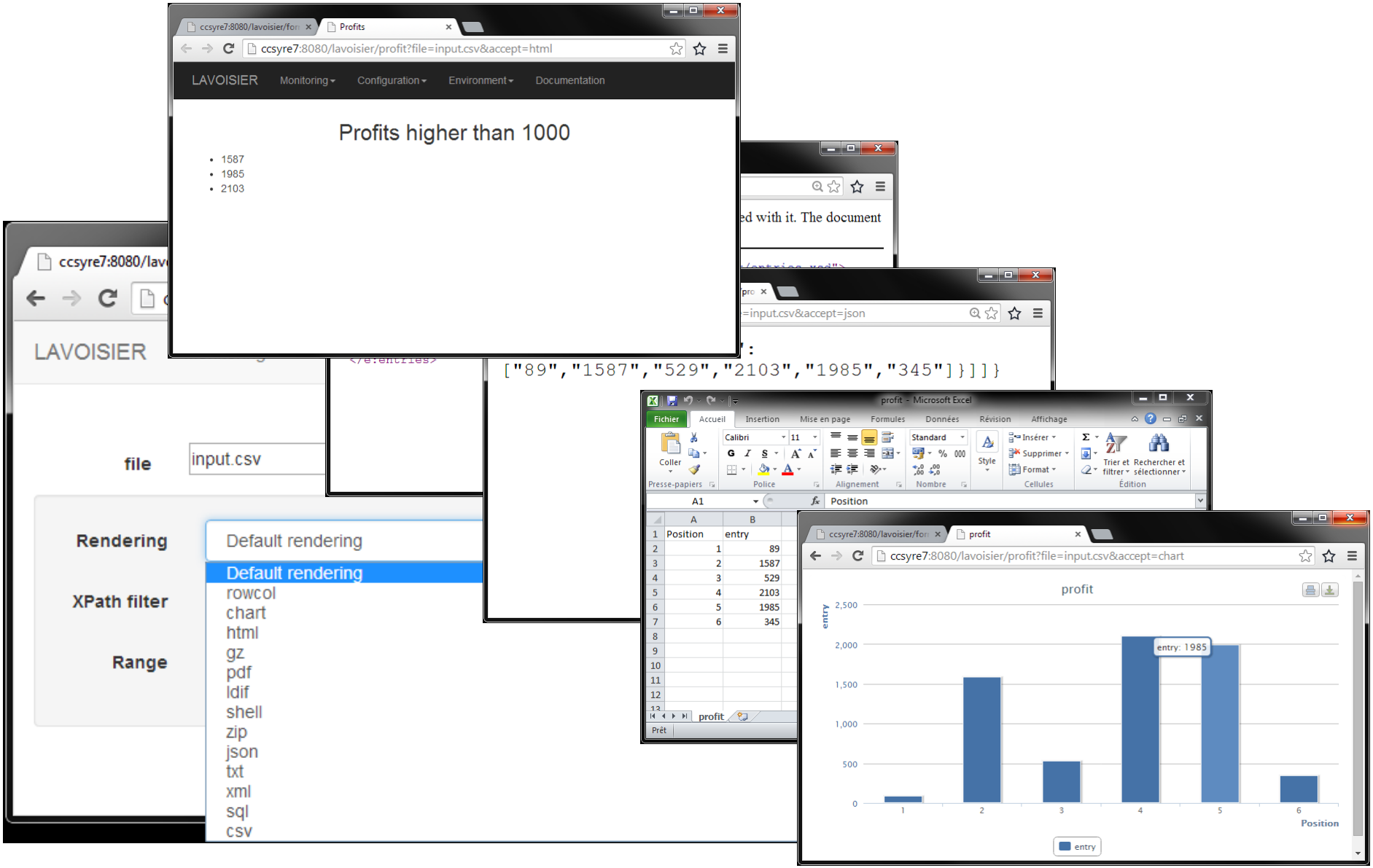
/ns_0:rendering/ns_0:rows/ns_0:row/ns_0:column/@label

SelectProcessor

match	1 `/ns_0:rendering/ns_0:rows/ns_0:row/ns_0:column[@label='Profit']/text()`

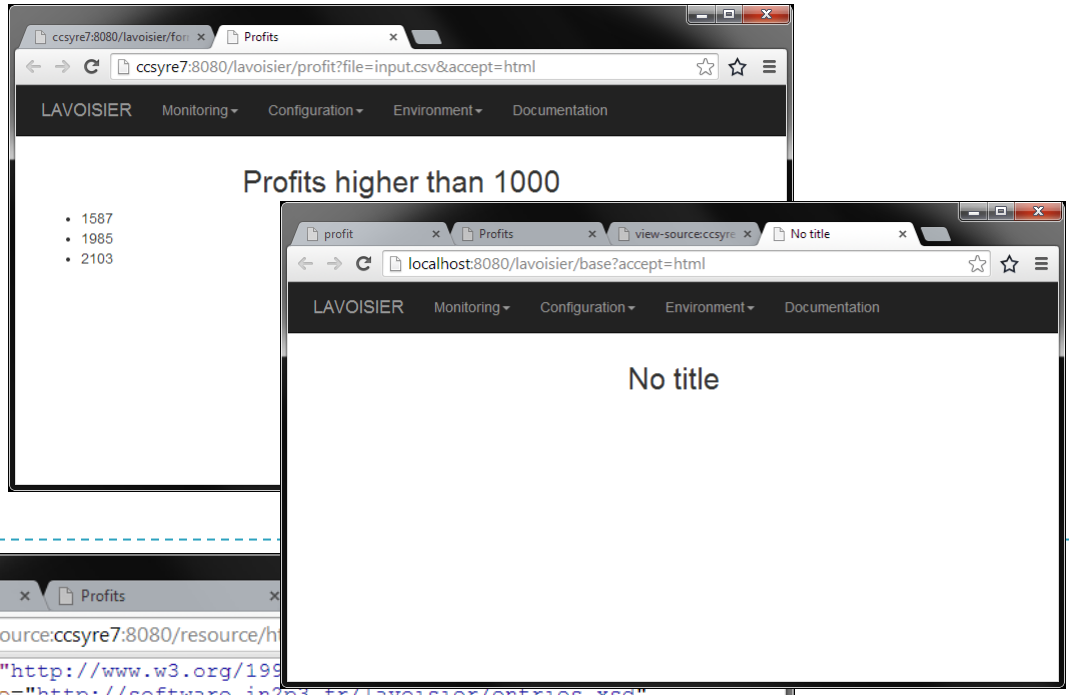


Example : download CSV data, extract column "Profit" and render to JSON



HTML template engine

Example : download CSV data, extract column "Profit" and render to JSON

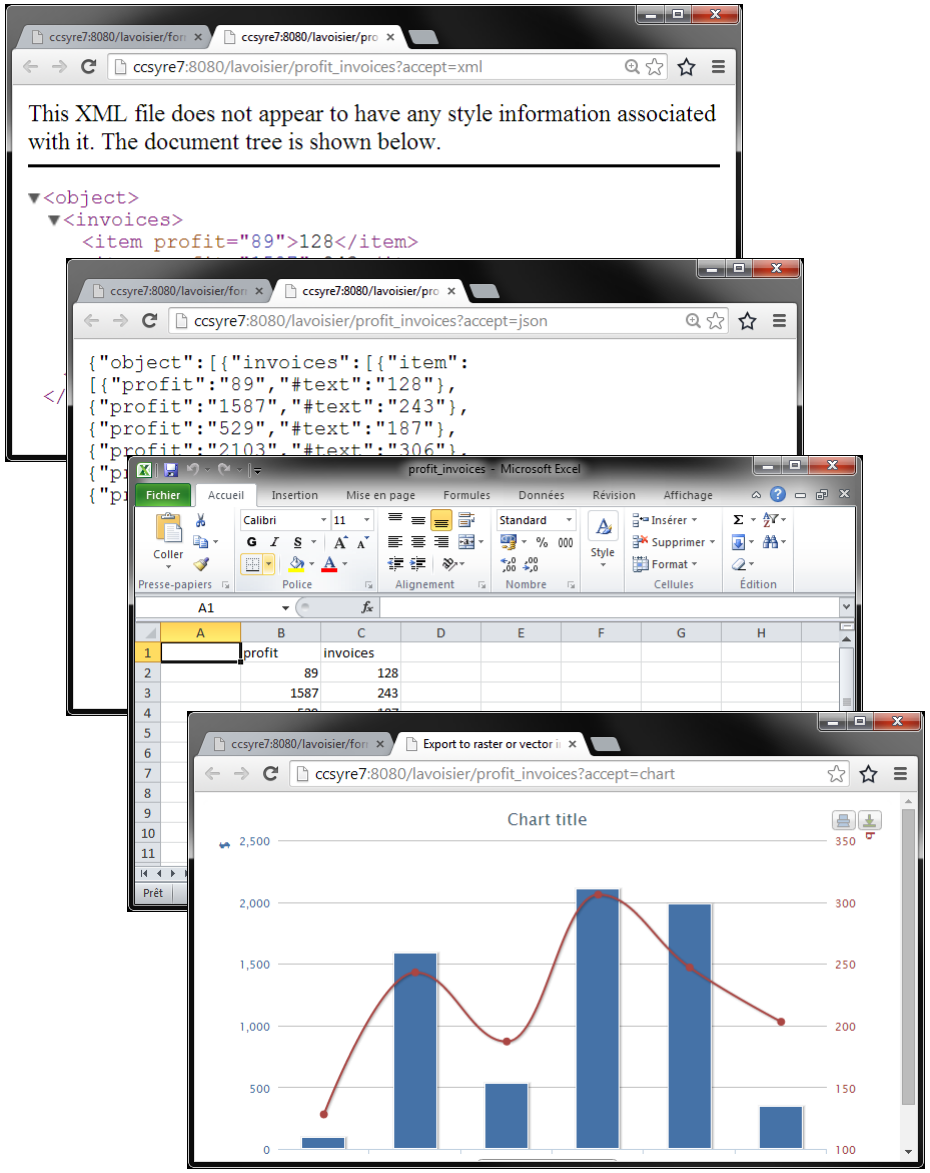


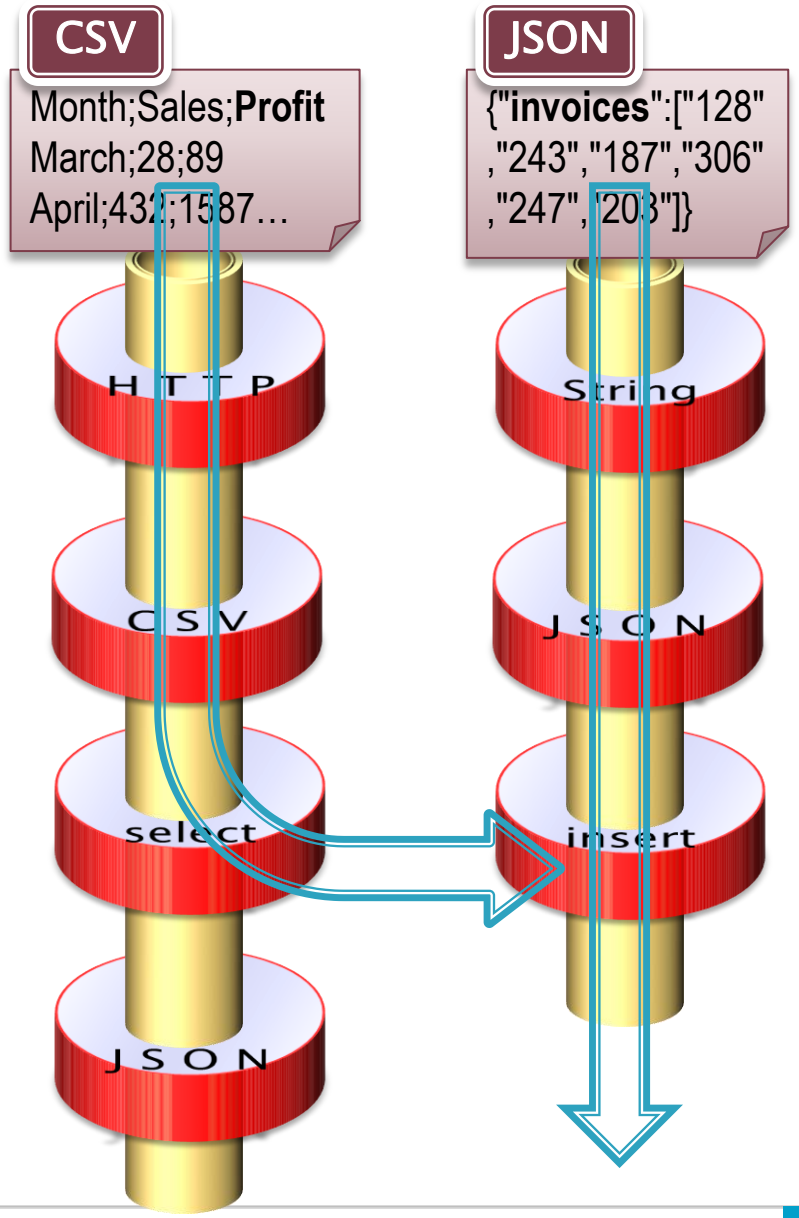
Generated

HTML Template

- XML/XPath support
- Client-side execution
- "Natural templating"
- Inheritance

Example : aggregate profit with number of invoices for each month





content	{"invoices":["128","243","187","306","247","203"]}

JSONSerializer

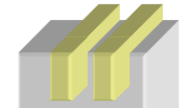
InsertProcessor

match	1 /object/invoices/item
nodes	1 new_attribute('profit', view('profit')/*/*[position()=position(match())])

```
<object>
    <invoices>
        <item profit="89">128</item>
        <item profit="1587">243</item>
        <item profit="529">187</item>
        <item profit="2103">306</item>
        <item profit="1985">247</item>
        <item profit="345">203</item>
    </invoices>
</object>
```

Click on XML to get XPATH

A real example :

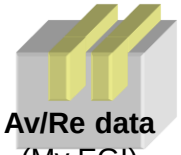


Downtimes
(GOC-DB)

- Previous example
  - helps to understand how to use Lavoisier
  - but Lavoisier is overkill for this !

VO IdCards
(Ops DB)

- A real use-case : the EGI Operations Portal
  - 15 heterogeneous data sources
  - 100 data views
  - 2000 users

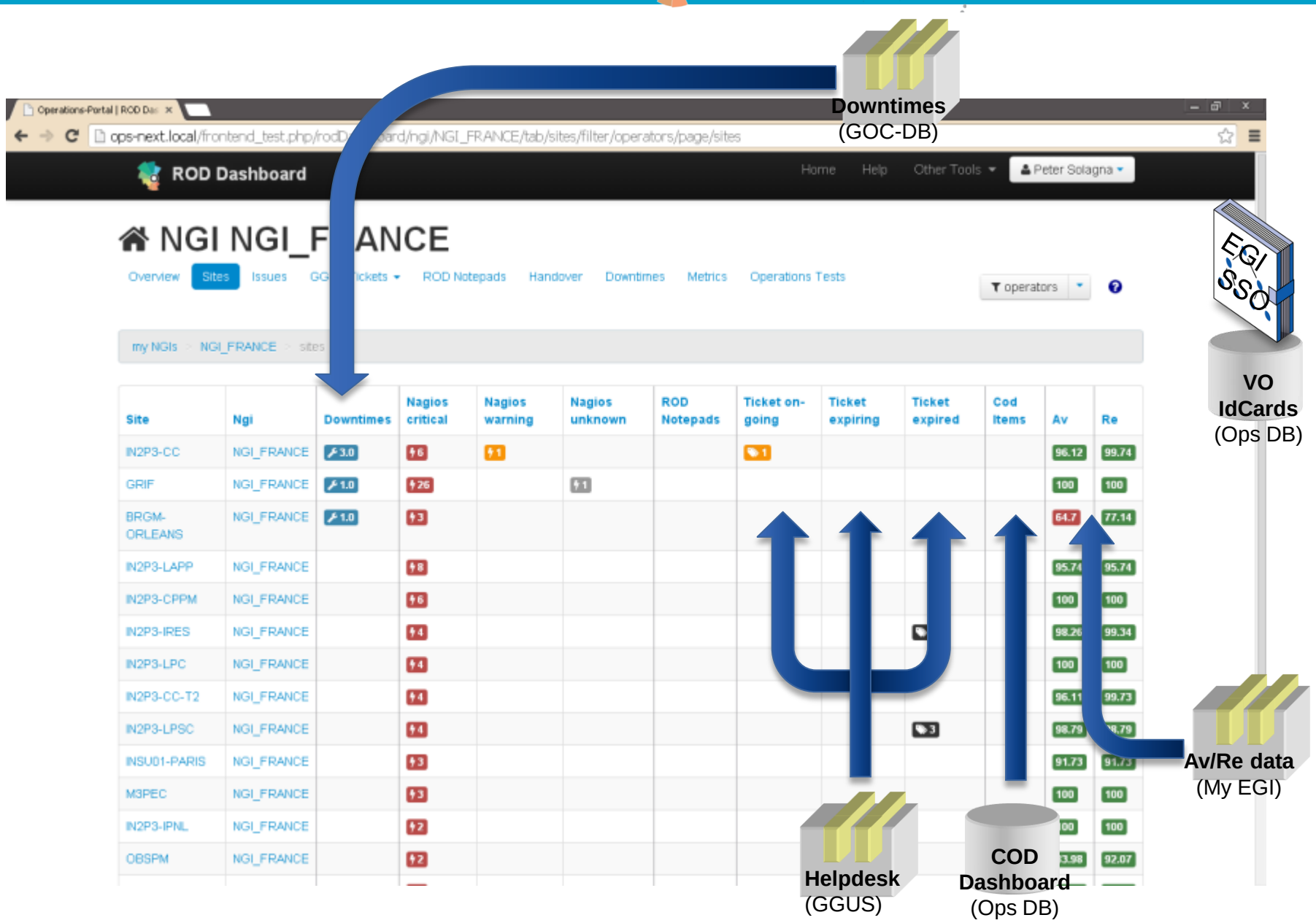


Av/Re data
(My EGI)

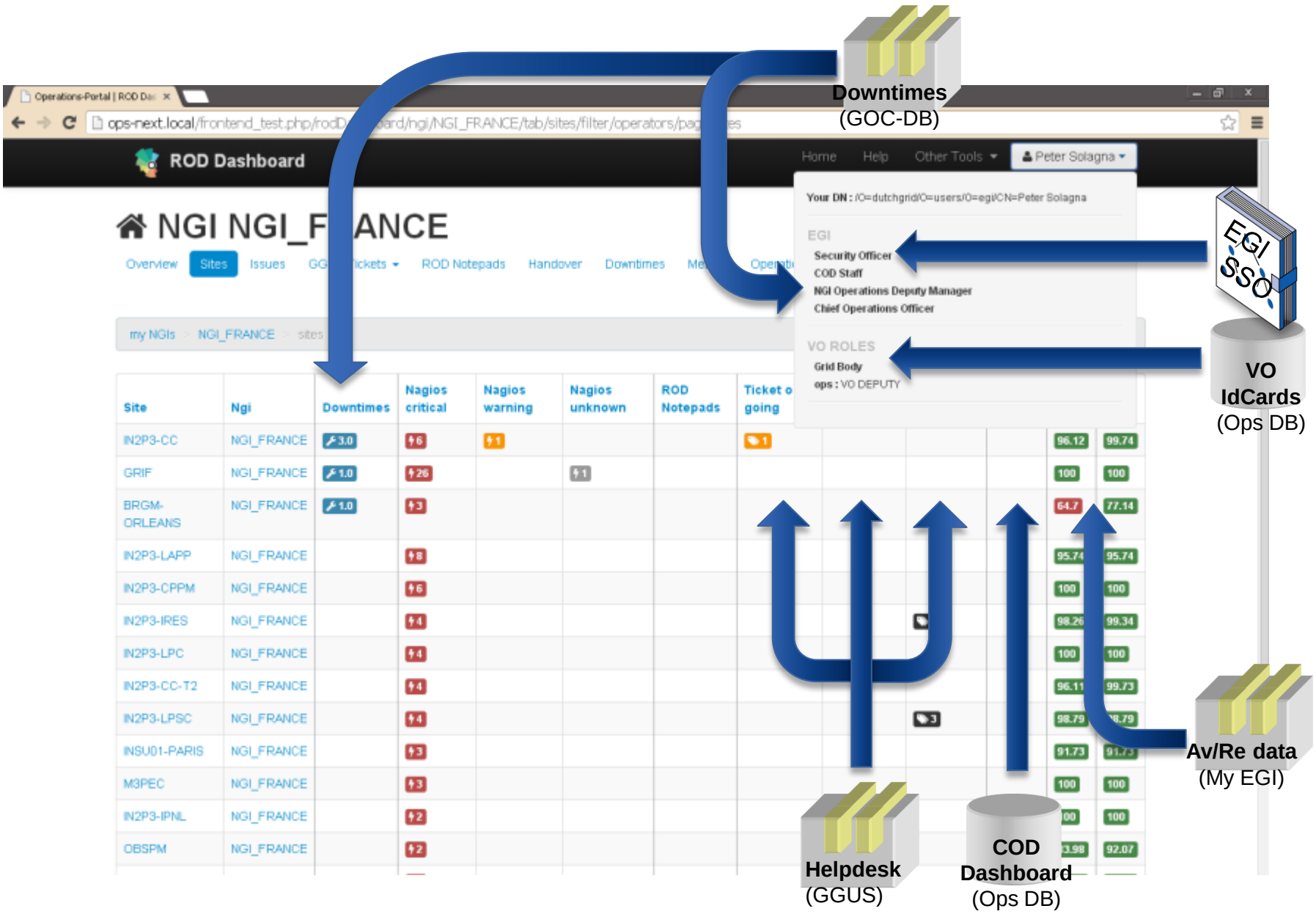
Helpdesk
(GGUS)

COD Dashboard
(Ops DB)

A real example :



A real example :



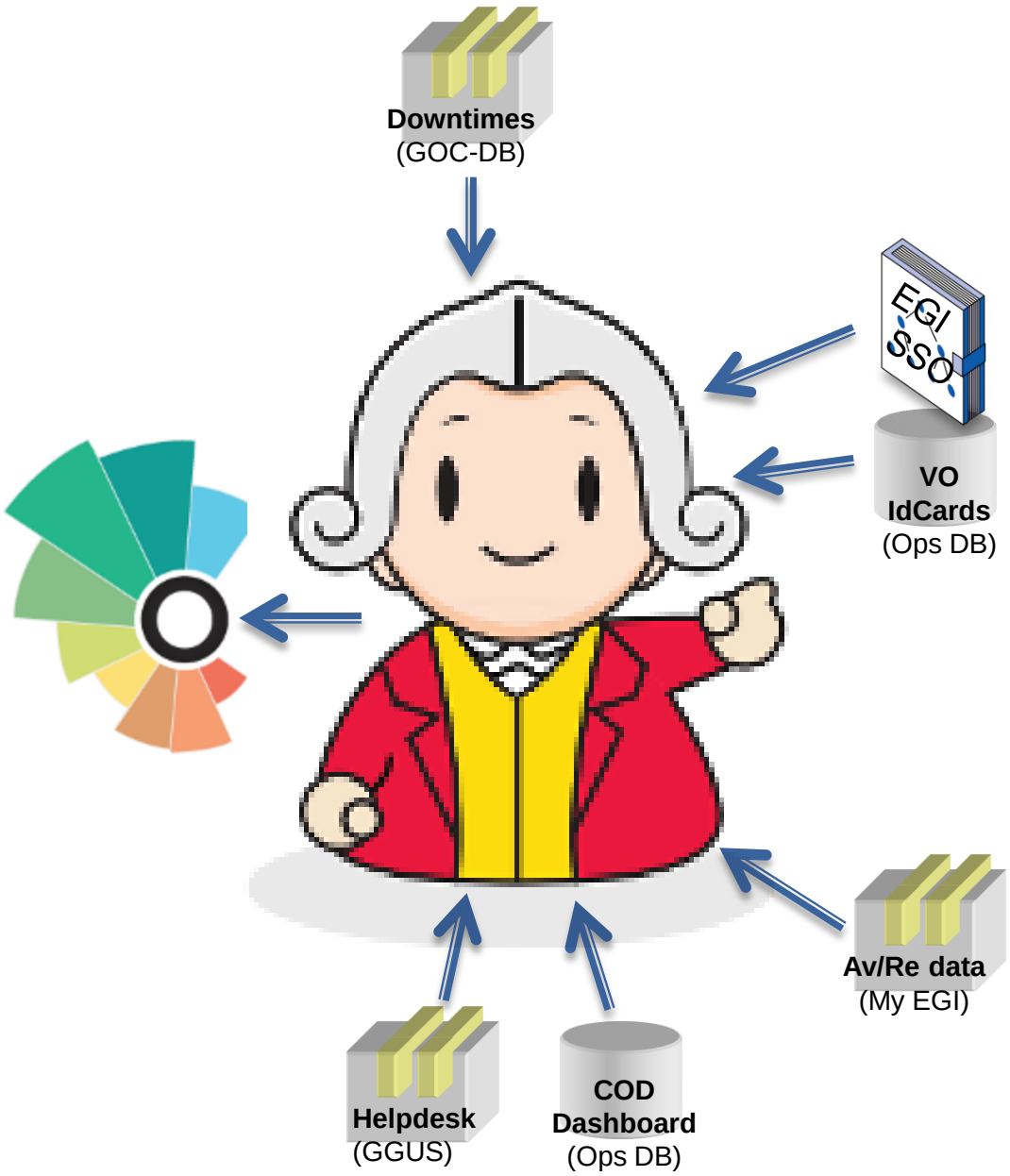
A real example :

Maintainability
- Modularity
- Single language

Performance
- Memory usage
- Latency

Availability
- Robustness
- Monitorability

Security
- Authentication
- Authorization



Lavoisier advantages

Maintainability
- Modularity
- Single language

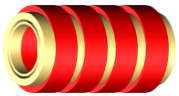
Performance
- Memory usage
- Latency

Availability
- Robustness
- Monitorability

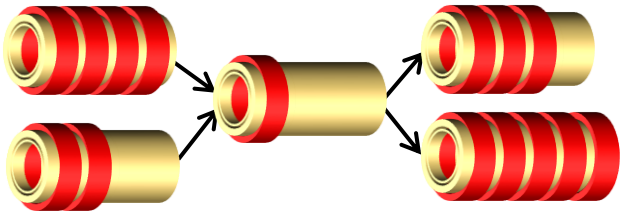
Security
- Authentication
- Authorization

Ensured by the configuration language

- technology/format integration
  - view = **chain of plugins**



- business code
  - application = **graph of data views**



Lavoisier advantages

Maintainability
- Modularity
- Single language

Performance
- Memory usage
- Latency

Availability
- Robustness
- Monitorability

Security
- Authentication
- Authorization

Single language (XPath) to :

- Define authorization rules
  - user()='127.0.0.1'
- Set parameter dynamically
  - concat('http://host:8080/lavoisier/', $file)
- Select parts of the data view
  - //row/column[@label='Profit']/text()
- Interact with
  - user request : arguments(), path()
  - environment : property(), document()
  - views : view(), match()
- Configure data rendering
  - row/columns
  - HTML template

Lavoisier advantages

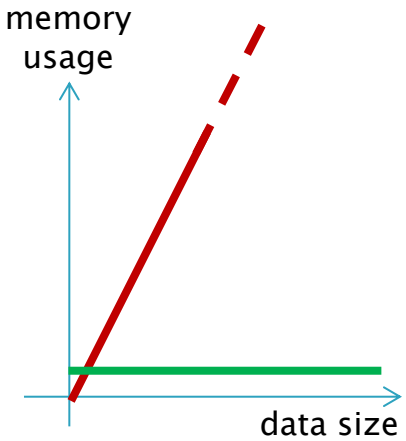
Maintainability
- Modularity
- Single language

Performance
- Memory usage
- Latency

Availability
- Robustness
- Monitorability

Security
- Authentication
- Authorization

- Plugins process streams
  - of bytes
  - of XML events



- XPath optimizer
  - avoid building big data structures
    - /absolute/xpath is transformed to a set of instructions processing the stream
    - ./relative/xpath is evaluated on small data structures
  - detect factorizable function calls

Transparent to developers

Lavoisier advantages

Maintainability
- Modularity
- Single language

Performance
- Memory usage
- Latency

Availability
- Robustness
- Monitorability

Security
- Authentication
- Authorization

Configure the cache mechanism:

- Per data view
  - choose cache plugin
  - combine refresh triggering rules
    - when view is notified by user
    - when dependency is refreshed
    - when time is elapsed
    - …
- According to the constrains of
  - data : size, time-to-live, dependencies
  - technology : latency, throughput, availability
  - users: patience, access frequency

Transparent to end-users

Lavoisier advantages

Maintainability
- Modularity
- Single language

Performance
- Memory usage
- Latency

Availability
- Robustness
- Monitorability

Security
- Authentication
- Authorization

Web console features

- Monitoring
  - Views build status, statistics…
- Configuration
  - Data views inter-dependencies
  - XPath expression optimizations
- Environment
  - OS, JVM, service…
- Debugging
  - Breakpoint on a XML stream
  - Breakpoint on a plugins chain
  - Skeleton of a data view

Developed with Lavoisier language

Lavoisier advantages

Maintainability
- Modularity
- Single language

Performance
- Memory usage
- Latency

Availability
- Robustness
- Monitorability

Security
- Authentication
- Authorization

Define security role per data view

- Authentication plugins
  - Chainable
  - Extensible
    - CAS, password, X509, IP, OAuth
  - Based on JAAS standard
- Authorization rules (XPath)
  - For each authentication plugin
  - Depending on argument value

```
view('authZ')/*/*[@user=user() and @key=$arg]
```

Conclusion

Maintainability
- Modularity
- Single language

Performance
- Memory usage
- Latency

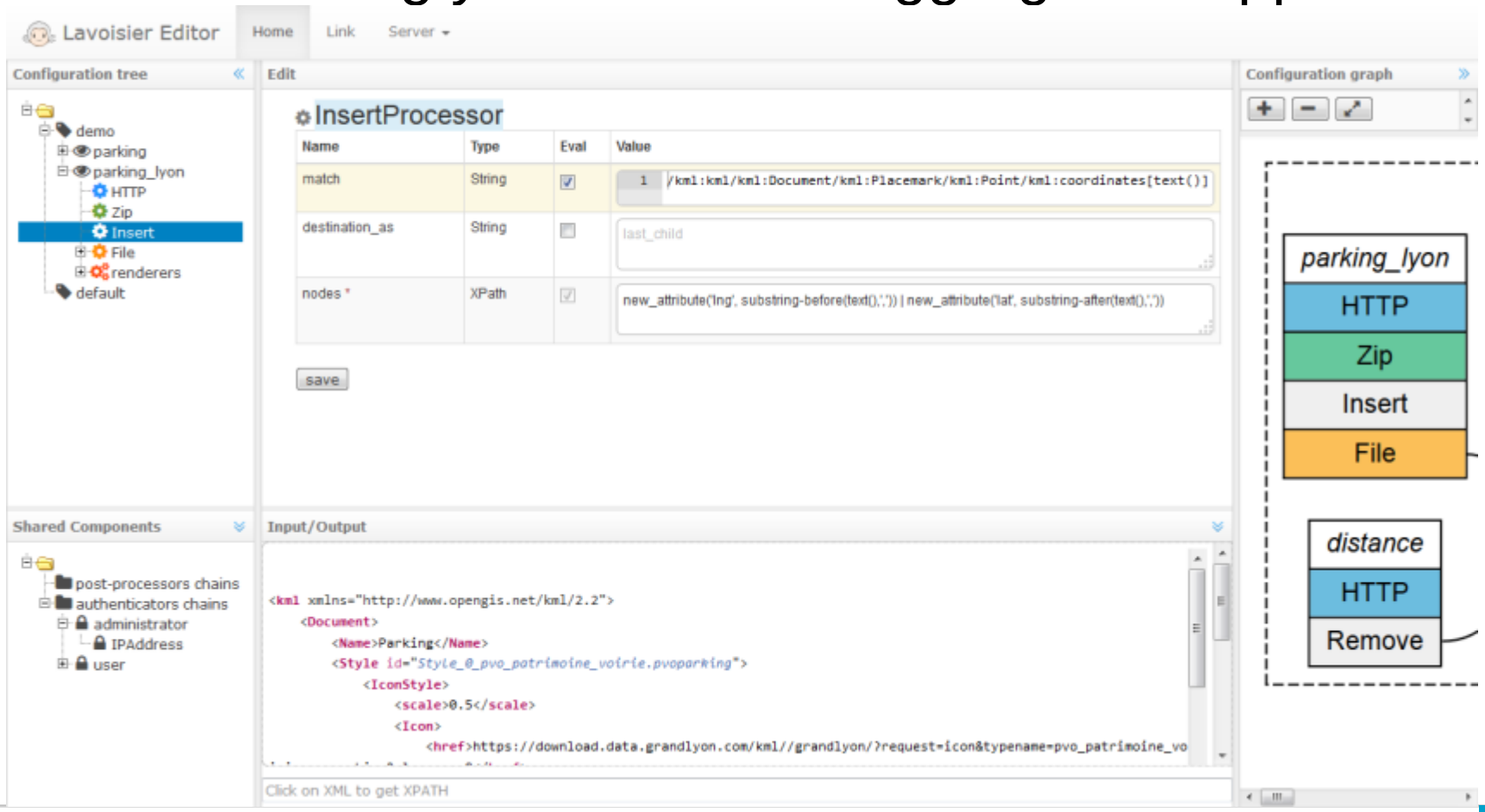
Availability
- Robustness
- Monitorability

Security
- Authentication
- Authorization

Lavoisier does all this for you…

…enabling you to focus on business code !

- New plugins
- On-the-fly partial reconfiguration of the framework
- GUI for building your own data aggregation application



Thank you !

Lavoisier 2 -

software.in2p3.fr/lavoisier/

CCIN2P3

LAVOISIER

DATA AGGREGATION FRAMEWORK

Last Published: 2014-05-06 | Version: 2.1.1-SNAPSHOT

CNRS | CC-IN2P3

LAVOISIER

- Project home
- Features
- Download
- Release notes
- Submit a bug

END USERS DOC

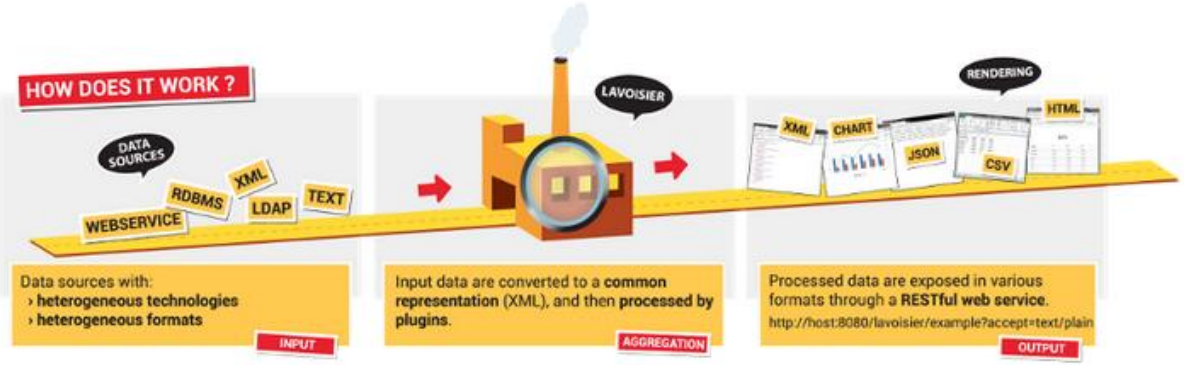
- Install guide
- User guide

ADVANCED USERS DOC

- Language guide
- Language reference
- Adaptors reference
- Configuration example

Nothing is lost, nothing is created, all is <xml:transformed>

Lavoisier is a framework which enables to retrieve, transform, merge and query heterogeneous data sources.



The resulting data is exposed as XML views through a RESTful web service interface (see **User guide**).

These XML data views are configured by defining chains of plugins (see **DataView Construction Language**