

# Plans for Operations Dashboard regionalization and distribution

1	Context .....	2
1.1	Current status .....	2
1.2	Evolution .....	2
2	Architecture proposal .....	3
2.1	Proposed solution .....	3
2.2	Presentation of the work .....	4
2.2.1	Work 1: Re-Organisation of the resources around Lavoisier .....	6
2.2.2	Work 2 : Adaptation to Nagios. ....	8
2.2.3	Work 3 : Packaging and Distribution in the regions .....	10
2.2.4	Work 4 : Provide for every regional model .....	10
3	Timeline, requirements and limitations .....	11
3.1	Timeline .....	11
3.2	Important considerations .....	13
4	Lavoisier . ....	13
4.1	Global Presentation .....	13
4.2	Features : .....	14
4.3	References : .....	14

# 1 Context

We give here plans on how to put in place a model enabling the distribution of the information for operations while keeping a central way of gathering it. We address the regionalization issue described in operations automation strategy MSA1.1 : <https://edms.cern.ch/document/927171>

## **1.1 Current status**

The model currently adopted by the operations dashboard is based on a central architecture based on a central database and a global view of all alarms raised against all federations based on a central SAM DB of results from sites' monitoring. The operations dashboard relies also on GGUS as the central operational ticketing system.

Operational tools currently rely on central instances . "Regional customization" is so far provided through views stored centrally.

## **1.2 Evolution**

Requirements onto the Operations Dashboard can be summarized as follows:

- define specifications of interfaces between the dashboard and the information sources to ensure interoperation between the various regional operational tools and models adopted according to the operational automation strategy through the use of web services.
- interact with third party info sources such as GOCDB/GGUS according to their future evolution and regional strategies ( Regional Helpdesks and Regional GOC DB).
- and the use of future versions of regional monitoring tools such as Nagios.

Provide adaptable tools to regional operational scenarios handled by NGIs and define the interaction between an EGI and NGIs operational tools i.e. provide a way to have potentially fully distributed operations and gather the relevant data at the central level.

This central level already in use will be kept in order to:

- address the continuity of work into the project
- propose a central instance for Regions not interested by the regionalisation
- propose a tool for Central Operations .

## 2 Architecture proposal

### *2.1 Proposed solution*

- The dashboard has to evolve from a central dashboard to a distributed model where every region will have the responsibility to apply an operational model of their choice to their sites.
- The resulting architecture has to allow regional instances of operational data to communicate with one another and with a central instance.
- In case a region is not willing to host its dashboard, we will continue to provide a “catch-all” instance to host the relevant information which will run with the information sources and the operational tools.
- Regional instances should be customizable and adaptable to local needs:
  - Integration of their « GOCDB » scenario into the dashboard (whether standalone, regional instance or catch-all instance of GOCDB).
  - Integration of their regional helpdesk scenario (through the use of Lavoisier).
  - National granularity to face EGI needs.

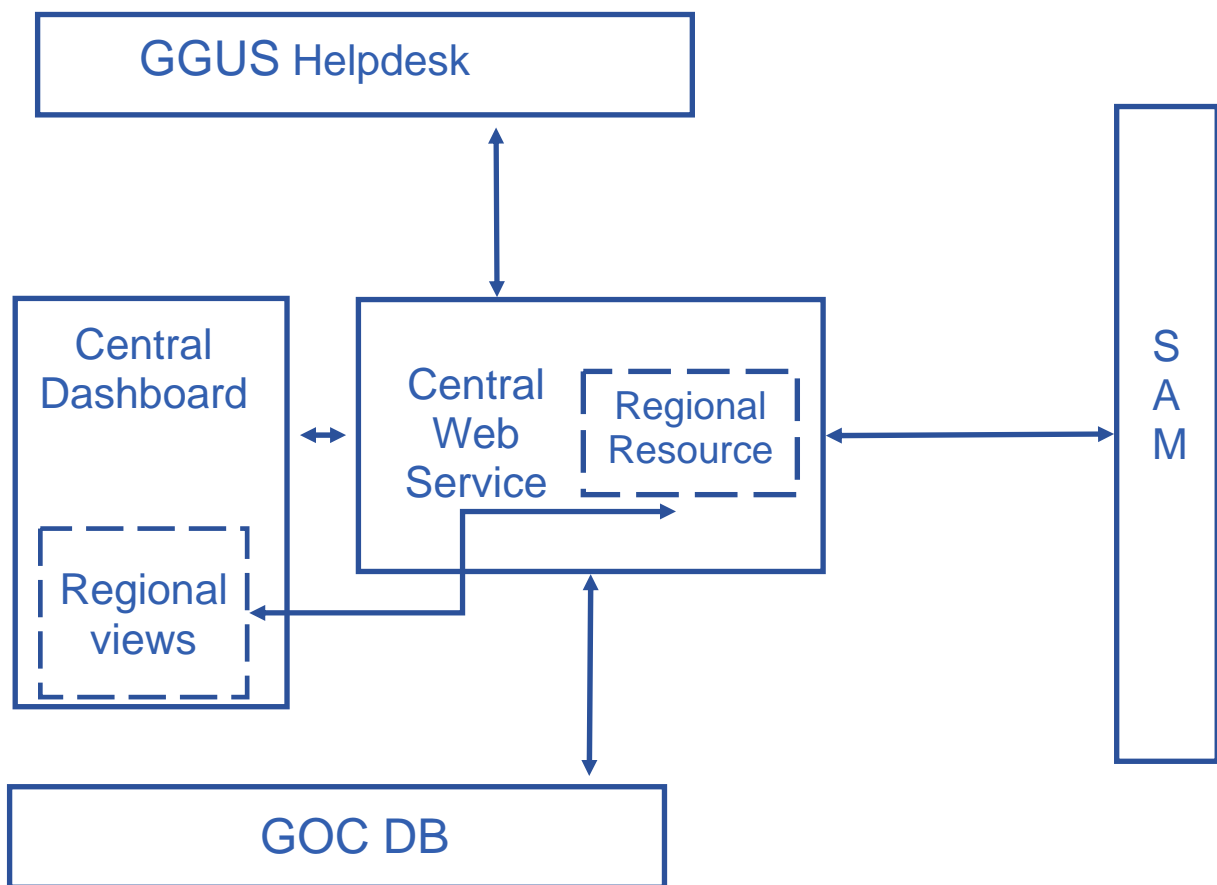
Firstly, we will rely on a generalization of the web services approach. This solution is already in use for some of the information sources: plug-ins have been developed in our web Service Lavoisier such as SAM , GOC DB , GGUS , BDII adapters.

We are highly dependent from the evolution of other tools: GOC DB, GGUS, SAM, Nagios. But the technical solution is organized around Lavoisier, which will assume the integration of these changes. The main idea is to have the integration of each resource via Lavoisier. Our different components will work in a standardized way with the outputs of the Lavoisier Web Service. The translation of resource in these standardized outputs will be assumed with different plug-ins.

## ***2.2 Presentation of the work***

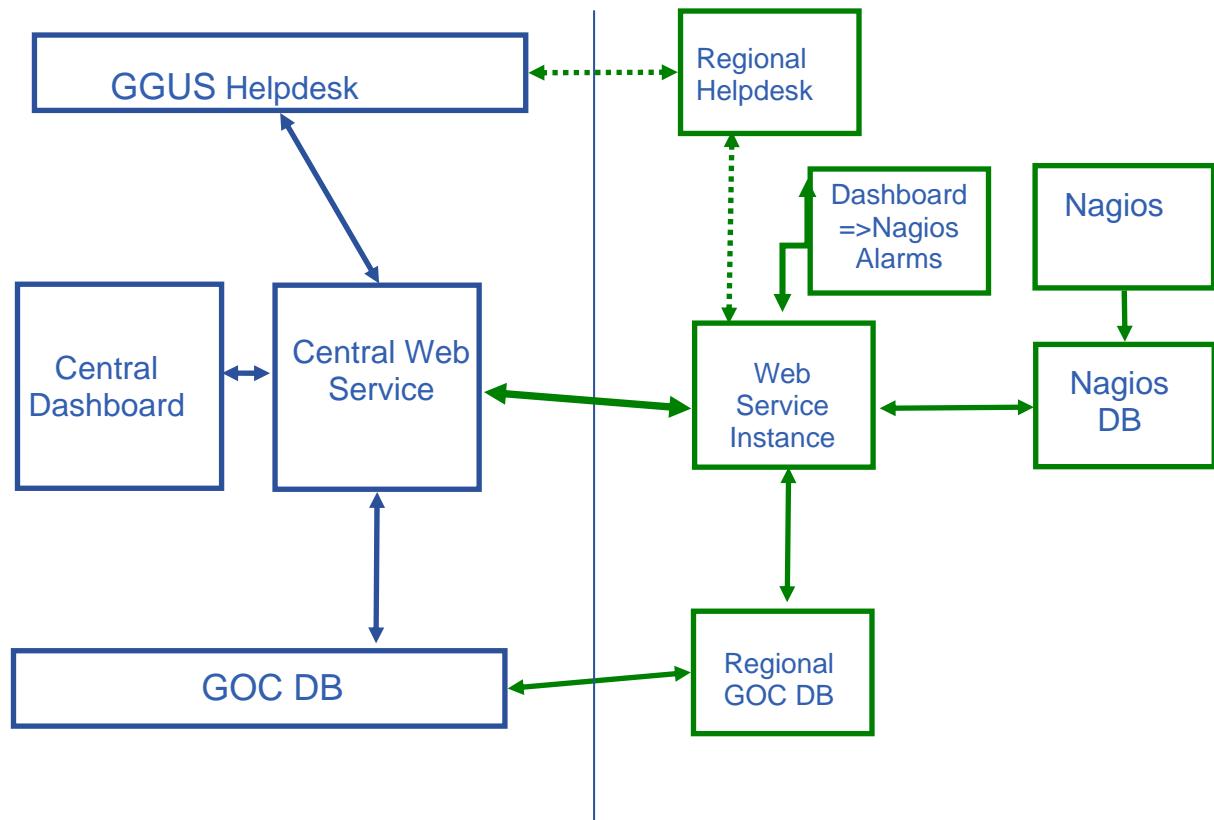
Current framework in production will undergo enhancements according to experience.

Initial situation:



Here is the schema of the current dashboard with all instances at a central level. Regional views are provided but at a central level.

Final situation in the most distributed case:



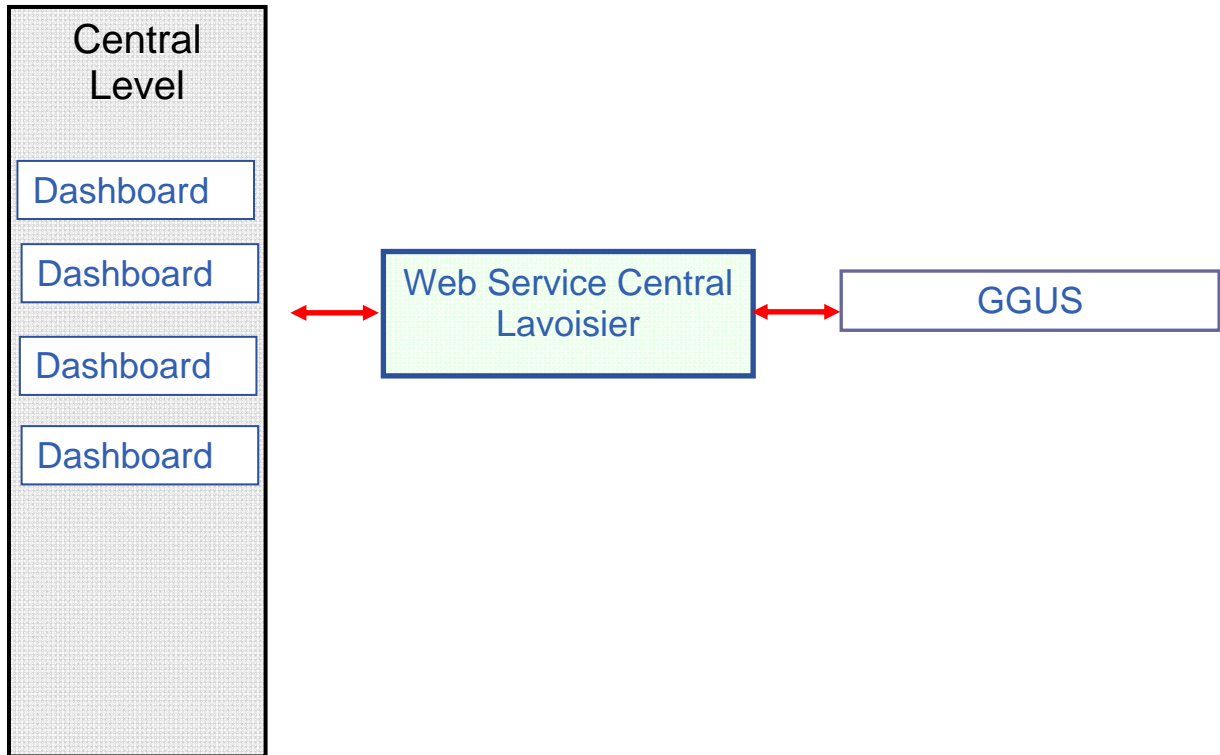
This schema presents the final situation in the most distributed case :

- the nagios monitoring is in place in Region
- GOC DB is deployed at a regional level
- GGUS is interfaced with a Regional Helpdesk.

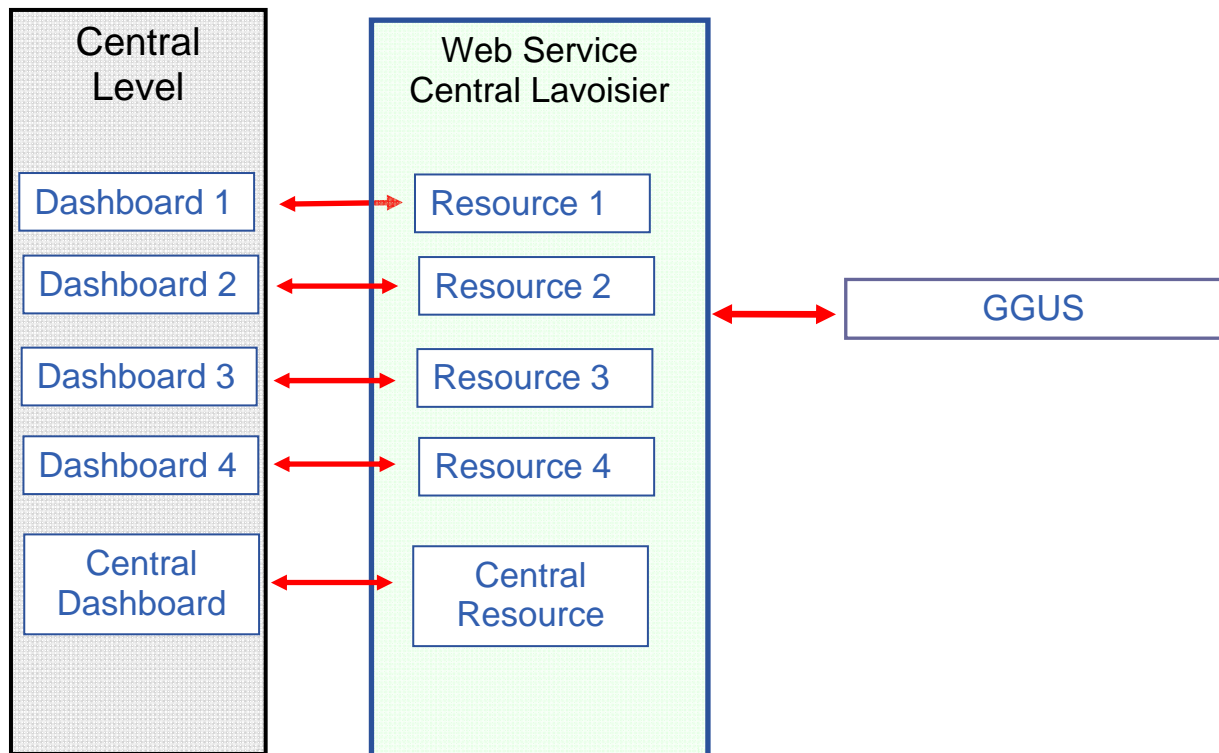
On the right you have the instances setup at a regional level for a given region. Every region will be interfaced with our central web service via a regional one. But this situation will be simplified by the fact that the instance will be the same at Regional Level and at Central Level. And consequently the interaction will request no extra development effort to be operational.

### 2.2.1 Work 1: Re-Organisation of the resources around Lavoisier

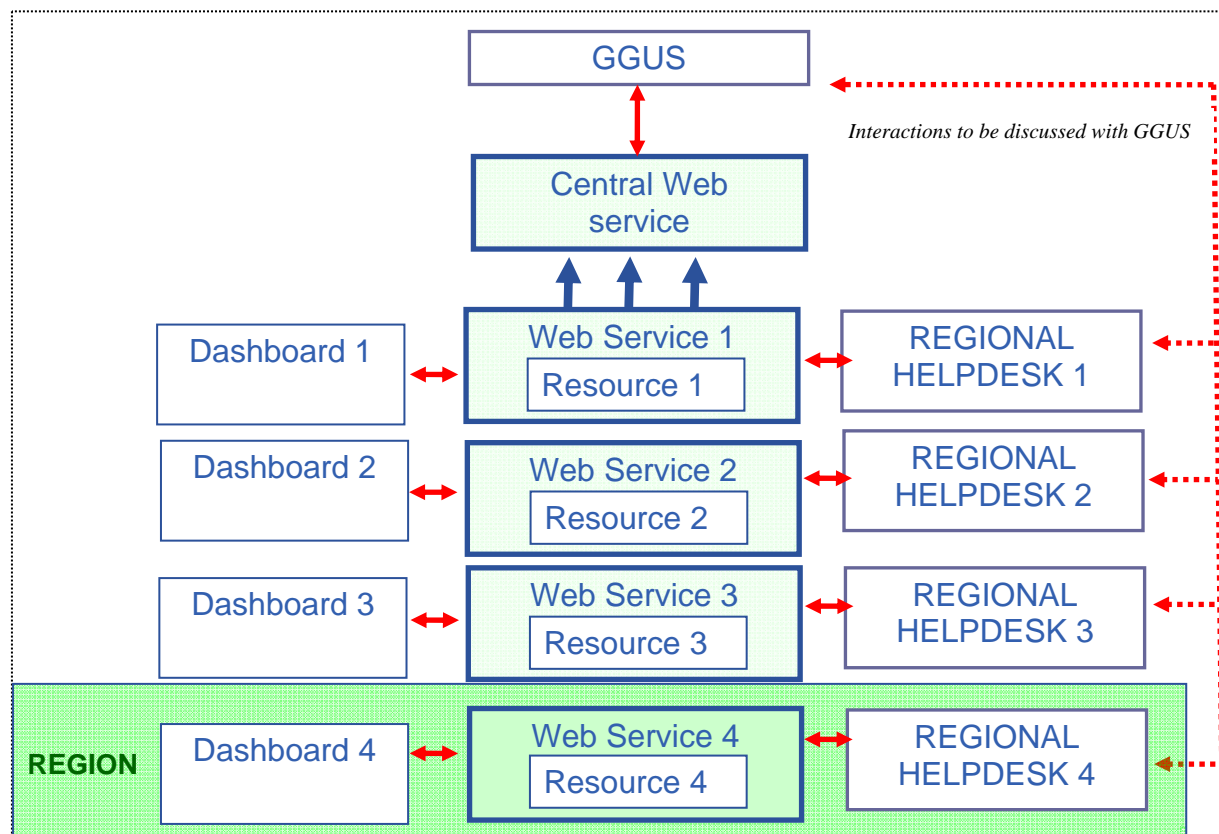
Step 1 represents the current situation.



Step 2 represents a simulated isolation of each regional resource and dashboard.



Step 3 represents the interaction of effective isolation of each regional resource and dashboard communicating with the central level through web service.



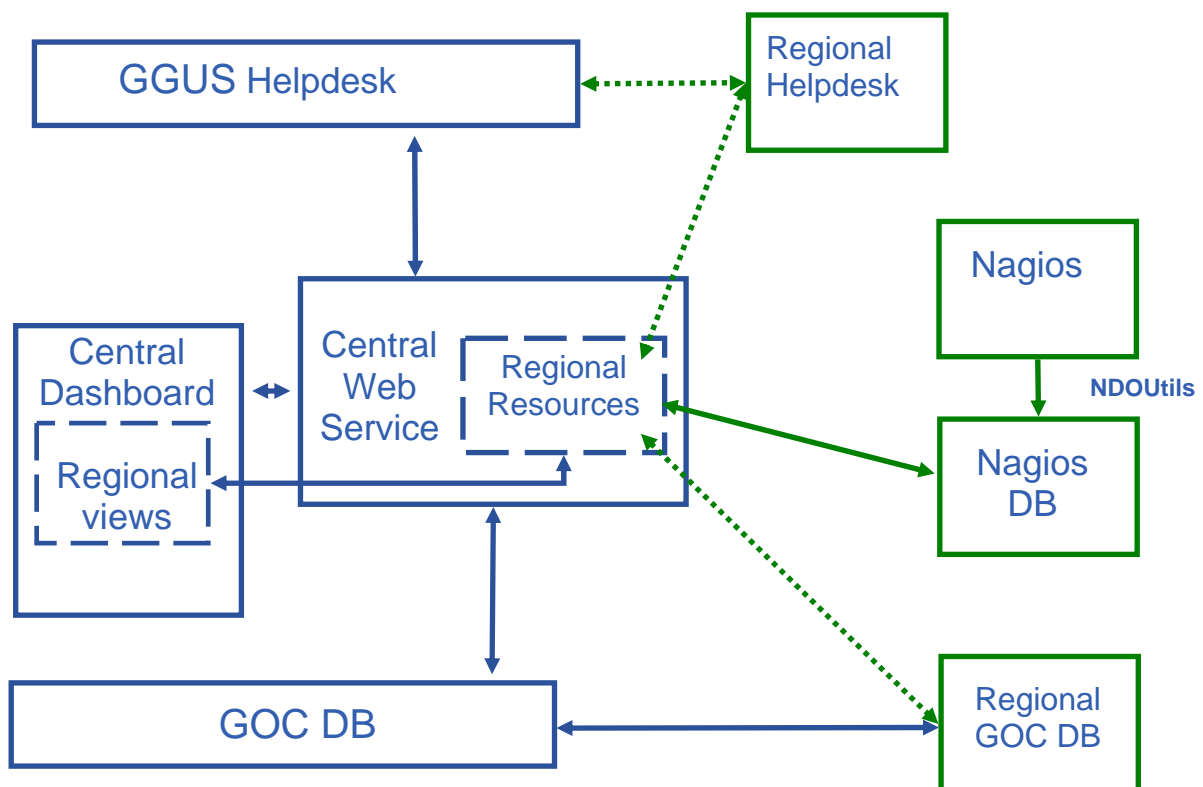
The final state is represented above where the right hand side of the sketch represents the regional instance of the tools.

### 2.2.2 Work 2 : Adaptation to Nagios.

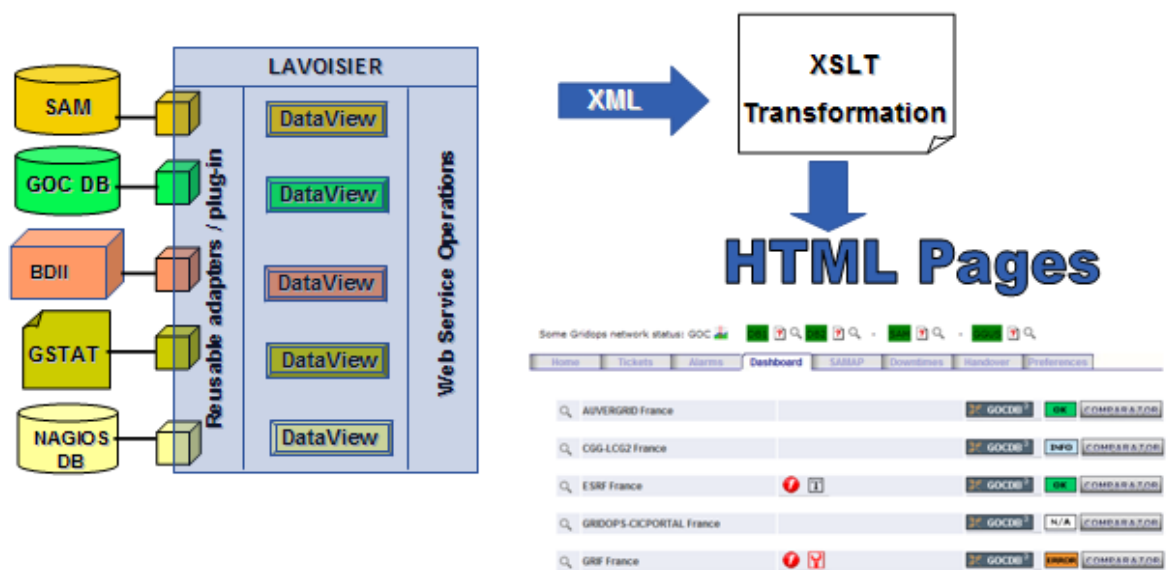
The move to Nagios implementation can start from initial phase as below.

Depending from the progress with the deployment work, we will start to integrate the nagios monitoring tool with a central view. Each federation will choose its own system of monitoring.

One of our concerns is to make sure the current operations are not altered. Indeed the Nagios evolution will be tackled as a separate undertaking to the regionalisation topic and the deployment of regional dashboard will start with the components present at a regional level or at central level by default .



Details of the Integration of Resources into the Web Service :



To work with Nagios we have to retrieve info from the Mysql Database and add it as usable resource for the dashboard.

In fact Nagios will be a new resource among other resources.

It will permit to integrate indifferently any Nagios in any Region.

This architecture will be the same at a regional and a central level.

The only difference will be the location of the resources but the usage and the transformation of these resources will be the same, as the interfaces are identical.

In fact after some discussions with the monitoring group in France we have decided to set up a regional Nagios in CC-IN2P3 :

We looked into the integration of features into the Web interface of Nagios it turns out that:

- The integration of new modules into the Web Interface is very painful, and it will not be easy to maintain.
- Moreover we have to be sure that dashboard is working independently from any specific monitoring tool.

Consequently the dashboard will be a stand alone instance integrating the information coming from the Mysql DB provided by NdoUtils in Nagios .

This work will be eased by the use of Lavoisier. We will develop a new adapter to integrate information from the Mysql DB coming with Nagios .

### **2.2.3 Work 3 : Packaging and Distribution in the regions**

This work will be essentially a reengineering work.

We will transform the dashboard into a package, easy deployable on a machine.

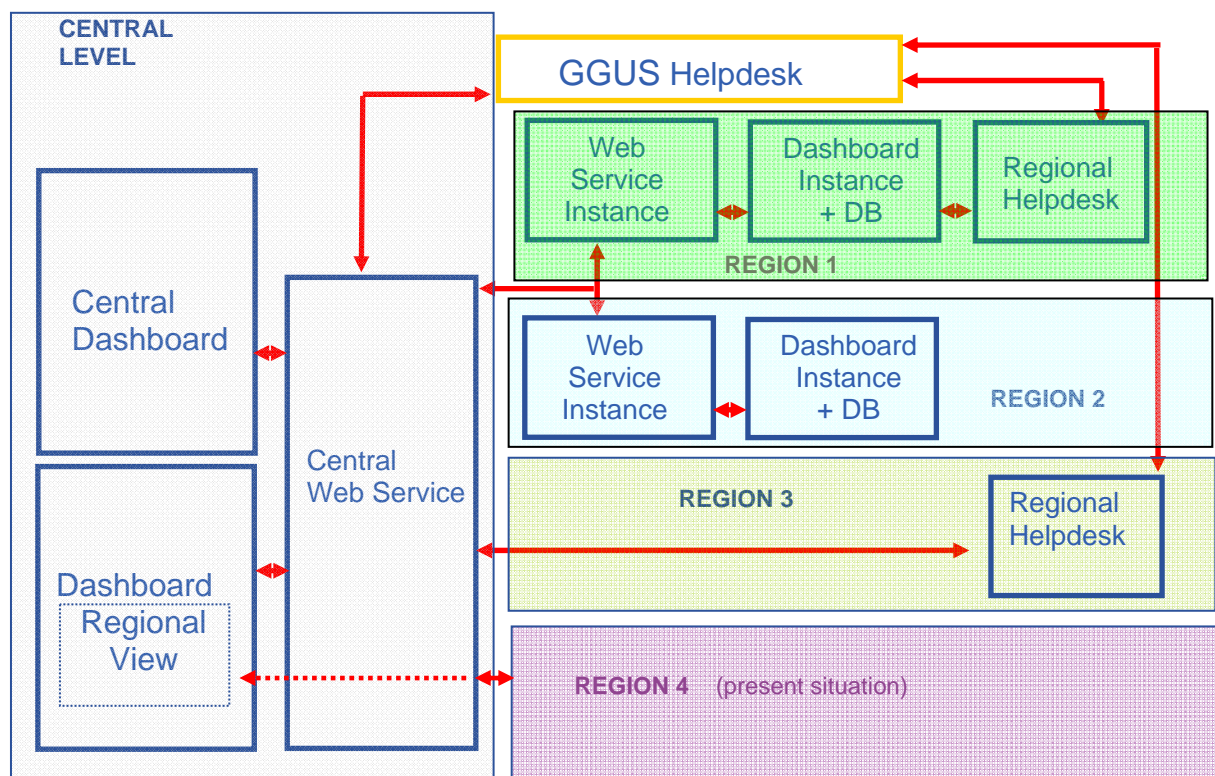
The work will be done on several parts:

- the php files : we need to be as independent as possible from external libraries
- the web service Lavoisier: we need to deliver a complete package of Lavoisier including the jar files (for the plug-in) , a global configuration file, the configuration files to exploit resources .
  - o <http://grid.in2p3.fr/lavoisier/>
- the database component : we have to provide an easily deployable DB and the tools to manage it . We will use the Database provided by NDOUTils in Nagios or the DB provided by SAM.

It means for the region to install a web service (Lavoisier), to deploy the php files, and of course a Regional Nagios.

### **2.2.4 Work 4 : Provide for every regional model**

In fact there is a set of different strategies for a region and the proposed solution enables us to classify the regions in 4 categories as shown below:



We must be sure that every case will work.

### 3 Timeline, requirements and limitations

#### 3.1 Timeline

**Short –Term – March 2009 - Phase 1** - Current work:

Future versions to include functional needs and lessons learnt from the test of regional operations of 1st quarter in 2009 i.e. Phase 1

Requirements already registered: [Savannah Support System](#) .

Namely, we have to take into account the outcome of the last COD meeting. i.e implement the metrics linked with alarms/tickets, the possibility to open tickets against OPS tools, add the possibility to open tickets with alarms not coming from SAM, especially for network troubles. Requirements are registered in [Savannah Support System](#) .

We have also to adapt to the new GOC DB programmatic interface. We have to transform in a first time the direct SQL queries of the php layer into queries to GOC interfaces. And in a second time we have to integrate this information into Lavoisier. (This task has to be finished for June).

### **Mid-term – June 2009 - Phase 2 (Work 1 and 2):**

Integration of Nagios information into a Central Dashboard with Regional View.

Development of an adapter for Lavoisier to retrieve information from Nagios .

Development of the dashboard regional evolution according to the steps towards regionalisation based on the current SAM probes.

These steps are needed to prepare for the interaction of the regional dashboard with the regional resources and the central instance of the web service.

At this point, each region will have the choice of handling its own strategy of regional or “catch-all” tools regarding GOCDB, Helpdesk and Dashboard.

### **Mid-Long Term – June to December 2009 (Work 3)**

Development of deployable toolkit for the dashboard .

Test of deployment with some ROCS ( Italy ?)

### **Long term - January 2010 - Phase 3 (Work 4 ):**

Deployment of regional packages into regions.

### ***3.2 Important considerations***

We will provide standard plug-ins to work with GGUS but the implementation of a new one for any regional helpdesk will be under the responsibility of region for the implementation.

This consideration is the same for other resources. We will provide reusable adapters for commonly used technologies: HTTP, LDAP, RDBMS, XSLT... and for classical resources used in EGEE ( BDII , GOC DB , VO id Card ) .

But development for specific resources has to be provided by the regions.

We will keep a central dashboard with regional views for regions which don't want to install a dashboard at a regional level.

For the regions which want to host a regional instance, they will be in charge of:

- maintaining a Lavoisier instance
- maintaining a Nagios instance with the Mysql Database

## **4 Lavoisier .**

### ***4.1 Global Presentation***

Lavoisier is a Open source Tool developed at CC-IN2P3 and will be maintained in the long terms. Indeed this tool is used in different interoperability projects the center is deeply committed to : JSAGA , IGTMD .

Lavoisier exposes its services as standard WS-Resource Properties operations. By exposing its services in a standardized way, client applications of Lavoisier can be based on standard tools.

Lavoisier has proven effective in increasing maintainability of the Operations portal. Indeed, the unified view enabled reducing considerably the amount and complexity of its code

Lavoisier is an extensible service designed to provide an unified view of data collected from multiple heterogeneous data sources (e.g. database, Web Service, LDAP, text files, XML files...). This unified view is represented as XML documents, which can be queried using standard languages, such as XSLT (and soon: XPath, XQuery).

Lavoisier also improves performance by caching the generated data views. The caching mechanism can be tuned according to the characteristics of the data views and of the data sources, and according to the constraints of the use-cases. This tuning has no impact, neither on plug-ins, nor on user's code.

The global architecture has four layers: the legacy data sources, the adapters used to generate the data views, the cache mechanism with flexible refresh triggering rules, and the service exposing both SOAP and REST (with XML or JSON) interfaces.

## **4.2 Features :**

- Access data views through SOAP, REST/JSON or REST/XML
  - get data data view
  - cross-query data views using standard languages: XSLT (and soon: XPath, XQuery)
- Separation of roles (changes related to a role have no impact on other roles)
  - the plug-in developer adds support for new technologies by developing adapters.
  - the service administrator configures the adapters and cache management according to the characteristics of the data views and data sources, and according to the constraints of the use-cases.
  - the user queries the data views.
- Data cache management
  - select cache type
    - in-memory (simple or hierarchical, parsed once)
    - on-disk (simple or hierarchical)
    - no-cache (directly invoke the data source)
  - combine cache refresh triggering rules: whenCreated, whenNotified, whenTime, whenMaxUnrefreshedTimeElapsed, whenMaxUnrefreshedTimeElapsedAndDataAccessed, whenUpdated, whenRefreshed, whenRefreshedAndPublished
- Extensibility
  - Reusable adapters are provided for commonly used technologies: HTTP, LDAP, RDBMS, XSLT...
  - Support for new technologies can be added by developing adapters
- Validation of generated XML data
  - check XML syntax only
  - check XML syntax and conformity to provided XML Schema
- Administration
  - get data views index and properties
  - get data views status
  - on-the-fly reconfiguration (detect views to be reconfigured)

## **4.3 References :**

- **Description and sources of Lavoisier**
  - <http://grid.in2p3.fr/lavoisier/>
  - <http://indico.cern.ch/materialDisplay.py?contribId=48&sessionId=13&materialId=paper&confId=048>