



SIMDAT

Data Grids for Process and Product Development using Numerical Simulation
and Knowledge Discovery

Project no.: 511438

Grid-based Systems for solving complex problems – IST Call 2
Integrated project



Deliverable

D.12.3.1 Service Architecture Design supporting Subscription as a first service

Start date of project: 1 September 2004

Duration: 48 months

Due date of deliverable: 01/09/2007

Actual submission date: 24/10/2007

Lead contractor for this deliverable: ECMWF

Revision: 1.1

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)		
Dissemination level		
PU	Public	
PP	Restricted to other programme participant (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	X

Revision history

Date	Version	Author	Modification
17.09.07	0.1	Guillaume Aubert	Initial version
25.09.07	0.2	Marta Gutierrez	Add Validation Report
26.09.07	0.3	Baudouin Raoult	Add Corrections
27.09.07	0.4	Guillaume Aubert	Add Data Repository Part
28.09.07	0.5	Guillaume Aubert	Corrections
01.10.07	0.6	Guillaume Aubert	Add Mike Turner's Comment
02.10.07	0.7	Marta Gutierrez	Add Chapter 6
03.10.07	0.8	Baudouin Raoult	Add Corrections
03.10.07	1.0	Guillaume Aubert	Add Correction & Release as Version 1.0
05.10.07	1.1	Guillaume Aubert	Add Lothar Wolf's Comment

Copyright

Copyright © Intel and other members of the SIMDAT consortium, www.simdat.org, 2007

Table of contents

1	Introduction	4
2	Subscription Service Requirements	4
2.1	Subscription Service goals	4
2.2	Functional Requirements.....	5
2.3	Infrastructure Requirements	7
3	Subscription Service Design	8
3.1	Data Repository the uniform interface to meteorological facilities	8
3.2	One Request Type to rule them all.....	8
3.3	Event Model	11
3.3.1	Catalogue Node and Data Repository out of sync	12
3.4	Subscription Information in the metadata definition.....	13
3.4.1	Destination, DestinationRepository	16
4	Subscription Service implementation within a Data Repository	18
4.1	Event Model Support in the DataRepository	18
4.2	Generic API and component framework.....	18
4.2.1	Subscription components	19
4.3	Internal Subscription Source	20
4.4	A file system based internal Subscription Source	20
5	Evaluation/Validation Report: Prototype PM36.	22
5.1	Grid Infrastructure.....	22
5.1.1	Requirements Validation.....	22
5.2	Distributed Data Access	24
5.3	Validation	24
5.4	Discovery, Ontology and Virtual Organisations	25
5.4.1	Validation	27
5.5	Analysis Services	32
6	Prototype at PM45.....	32
7	Resulting Requirements for PM42.....	32
8	Conclusion.....	38

1 Introduction

As the middle of phase III is reached, the framework developed by the Meteorology Activity partners has covered and fulfilled most of the requirements that have been defined at the start of the project. A complete infrastructure based on an innovative decentralized architecture and offering cataloguing, discovery and data retrieval facilities has been deployed on 11 sites, providing a worldwide coverage. During the first six months of SIMDAT phase III, a security solution relying on the decentralised architecture and offering a cross-domain single sign-on has been implemented to protect the most sensitive datasets. The software is also now available under an Open Source licence and a new version including the security modules is being redeployed on the different sites involved.

One of the remaining requirements, the Subscription Service, needs to be implemented in order to have an infrastructure ready for production. In the meteorology community, the Subscription Service is a delivery system offering users the possibility to subscribe to data that will be produced in the future. The Subscription Service will use a push mechanism to deliver the data to the users as soon as they become available.

This document details the requirements for the Subscription Service and provides a first design, which will imply major modifications to the current implementation, as it is an activity that touches all the different components of the VMC infrastructure.

The document also describes the Evaluation/Validation Report of the Meteo Prototype at PM36.

2 Subscription Service Requirements

A preliminary description of the requirements of the Subscription Service can be found in the initial first deliverable of the Meteorology Activity: *D18.1.1 Consolidated Meteorology Requirements*, paragraph 3.2.2.3 Subscription mode (Push Mode). Even though the requirements have not changed, it is time to revisit them in light of what has been learned during the last 3 years of the project. The requirements will be also reviewed to take into account the architecture built in response to the VGISC challenges. The following analysis will focus strongly on how a user perceives and interacts with the Subscription Service of the VGISC.

2.1 Subscription Service goals

The Subscription Service objective is to offer to the VGISC users a way to receive “future” data, that is data that will be produced at a later stage, usually in a recurring fashion (e.g. every day).

Most of the activity of the meteorology community worldwide is based on such the production and usage of such “real time” data:

- Observations (temperature, wind, pressure, ...) are collected on a regular basis (every 6 hours) throughout the world by observers or automatic stations and sent to sites that centralize these data

- The data is used as input to numerical weather prediction (NWP) models that produce daily products such as forecasts for the next 15 days (twice a day in the case of ECMWF).
- These model outputs will be sent to forecasters that will create products for end-users, such as decision makers (ship and aircraft routing, agriculture,...) or the general public (TV, radio, ...). The forecasters may also issue warnings (storms, cyclones, ...) to public authorities.

In the scenario described above, each step subscribes to the previous one, creating a flow of more and more elaborated information, from the observer to the end user.

In order to become a truly operational “Virtual Meteorological Centre”, The VGISC has to implement the Subscription Service.

Not only will the Subscription Service allow users to receive the data as soon as possible, but such a data delivery mechanism (also called “push” mechanism) minimizes the bandwidth and resources consumption, as there is no need for users to constantly query the meteorological databases to check for data availability.

2.2 Functional Requirements

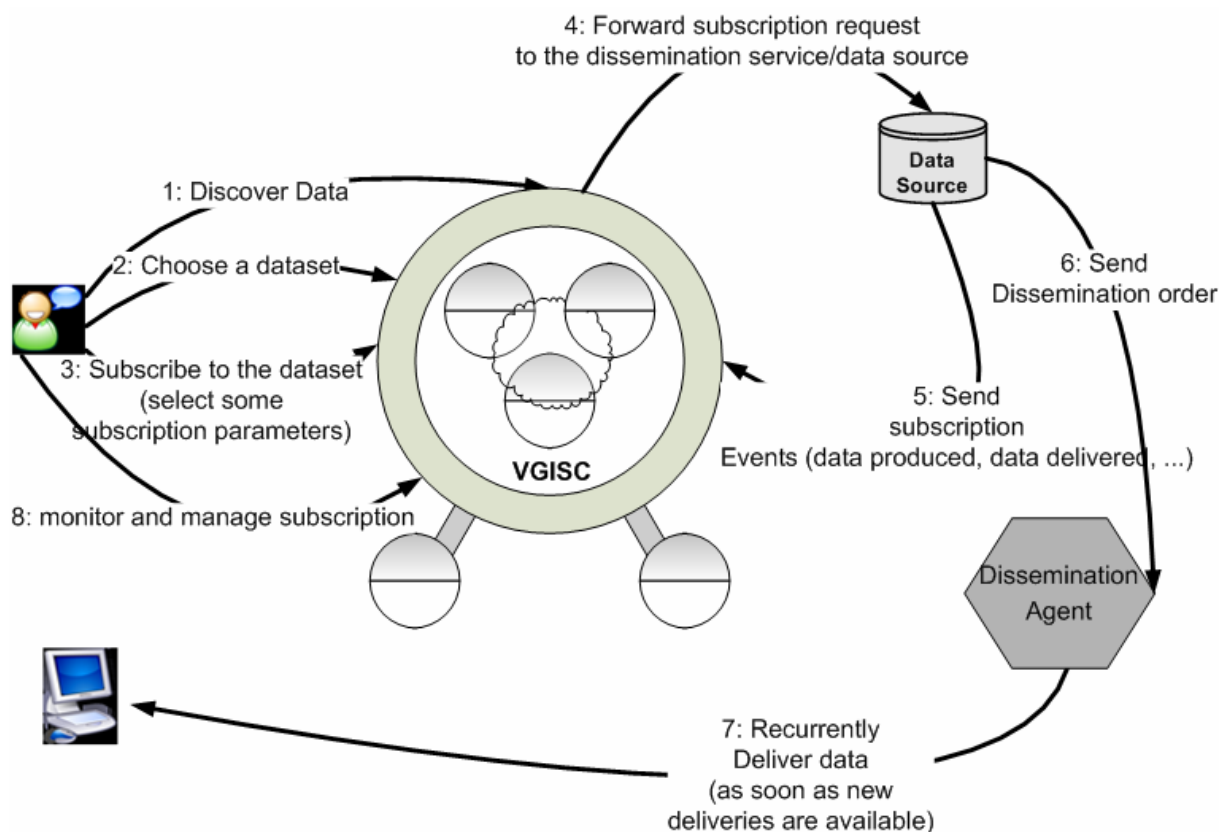


Figure 1 Subscription Scenario

The diagram above represents a typical subscription scenario, during which a user discovers one or more datasets of interest using the VGISC discovery facilities (1.) and, by selecting a given dataset from the list, is provided with a web page containing a detailed description of the chosen dataset (2.).

From this page the user has the possibility to subscribe to the dataset in order to receive any future data items that will be added to the dataset (3.).

The user can parameterize his/her subscription request by specifying which elements of the dataset he/she is interested in and through which dissemination mechanism he/she would like to receive the produce data. For example Jean-Paul Mouchard, a forecaster from Météo-France would like to receive, as soon as it is available, the temperature and cloud cover parameters restricted to the France area from each run of the ECMWF deterministic forecast. The service should send the data to the *ftp.meteo.fr* server and store it under a directory named *ECWMF-Forecast*. In order to achieve this, the user will need to define *ftp.meteo.fr* as a possible destination, and provide all the relevant information such as login name, password and number of retries.

The VGISC receives the subscription request (4.), stores the user-subscription relation in the request database and forward the request to the data source, which is coupled with an internal or external dissemination service:

- An internal data source/dissemination service is a service that is entirely managed by the VGISC software, using the infrastructure that has been deployed. For example, once a piece of data is ready, it is added to the subscriber's shopping cart in the VGISC portal so it can be downloaded.
- An external data source/dissemination service is a service that handles data delivery outside the VGISC infrastructure. Such services are usually dissemination services that already exist, such as WebWerdis (DWD), DIFFMET (Météo France) or ECPDS (ECMWF).

The data source/dissemination system constantly reports (5.) to the VGISC information on regarding the running subscriptions and sends events reporting the availability of new data for existing subscription, successful deliveries and errors.

As soon as some data is received, processed and linked to the subscription of the user, the data is passed to the dissemination agent (6.) which will send it to the user using the chosen supported data transport (7.). In addition, the user can monitor his/her subscriptions and see the last events received which are related to these subscriptions (8.). The user can modify his/her subscriptions parameters or cancel any subscription (8.).

Subscription Functional Requirements:
- A User can subscribe to a dataset supporting subscription in order to receive future data related to the dataset
- A User can select which parameters of the dataset are of interest in order to refine the subscription
- A User can define different destinations defining which transport protocol has to be used and where to send the data
- A User can modify his/her destinations and link one or several destinations to a subscription as long as the subscription supports the delivery mode used by the destination

- A User can monitor his/her subscriptions and check all the data deliveries related to each of the subscriptions
- A User can modify/cancel his/her subscriptions

2.3 Infrastructure Requirements

One of the corner-stone requirements of the VGISC framework was to build a non-intrusive infrastructure that adapts to the existing applications of the different meteorological services.

The Subscription Service has to follow the same philosophy and needs to provide an easy way to make use of existing dissemination systems such as DIFFMET or WebWerdis. Up to now, the VGISC framework has been built following a very generic approach, so that any data provider could continue to use its own database request mechanisms and parameters.

This generic approach must be preserved and extended to the Subscription Service, allowing “subscription providers” to make use of subscription mechanisms and parameterisation that are proper to their exiting dissemination systems.

Furthermore, the Subscription Service should be built on proven mechanisms and technologies, which must last many years (10 to 20).

Finally the VGISC architecture is designed following a fully decentralised model based on a peer-to-peer organisation (mesh network) in which any site can reach any other peers without being directly connected. The Subscription Service needs to work within this decentralized approach: for example, Jean-Paul Mouchard should be able to use the Météo France portal to subscribe to a dataset which is served by the Australian Bureau of Meteorology site, even though the Australian node is not directly connected to the French node.

Subscription Framework Requirements:
- The Subscription Service needs to be non intrusive and has to adapt to the existing dissemination and subscription applications,
- The Subscription Service needs to be built on proven concepts and technologies that will last for decades
- The Subscription Service should be generic to easily integrate new technologies and new data transports that will appear in the future.
- The Subscription Service should support the current decentralized VGISC architecture based on a mesh network topology.

3 Subscription Service Design

This chapter details the key design choices for the implementation of the Subscription Service.

3.1 Data Repository the uniform interface to meteorological facilities

The Data Repository has been designed as a bridge between the VGISC world and the meteorological facilities of each partner. In the current version, a Data Repository provides facilities to request on-demand data from a meteorological data source and to define ISO19115 metadata records describing the datasets accessible from this data source.

The Data Repository is the ideal candidate to relate user's subscription requests with new dataset arrivals as it is already connected to the real meteorological database and knows when new datasets are produced. The Data Repository is also the best candidate to orchestrate the work between the meteorological database and the dissemination system in order to deliver as soon as possible the newly produced data.

This is why in the design of Subscription Service, the Data Repository maintains a database of subscriptions and matches all new data arrivals with the users' subscriptions. The Data Repository also sends orders to the dissemination agent to transport the newly produced data to the interested users. The dissemination agent can be internal or external to the Data Repository.

The Catalogue Node will also host a subscription request repository in order to give to the user the control over its subscription requests. When a new subscription is created, deleted, or when an existing subscription is modified by the user, a new message will be sent to the Data Repository to update its database. The Catalogue Node manages the master copy of the subscription information, and is the only component that can modify, delete or create a subscription. However even though the Catalogue Node owns the subscriptions, the Data Repository is responsible for running them. In effect, the Catalogue Node will act only as a proxy between the user and the Data Repository.

3.2 One Request Type to rule them all

In the current VGISC architecture, when a user requests some data, an on-demand request is generated and stored into the Request Repository. The request mainly contains a request ID, a status, some logging information and some information to facilitate the download like the data size, data type, etc. The request content is updated during the request lifetime with the messages received from the data source (e.g.: the meteorological database interfaced to the VGISC world through the Data Repository).

Following the same idea, when a user subscribes to a dataset, a subscription request is created and updated using the messages received from the data source. This concept can be generalized to have one request object representing on-demand requests and subscription requests. *An on-demand request is a subscription request with one unique Data Delivery where the Destination is the shopping-cart of the user.*

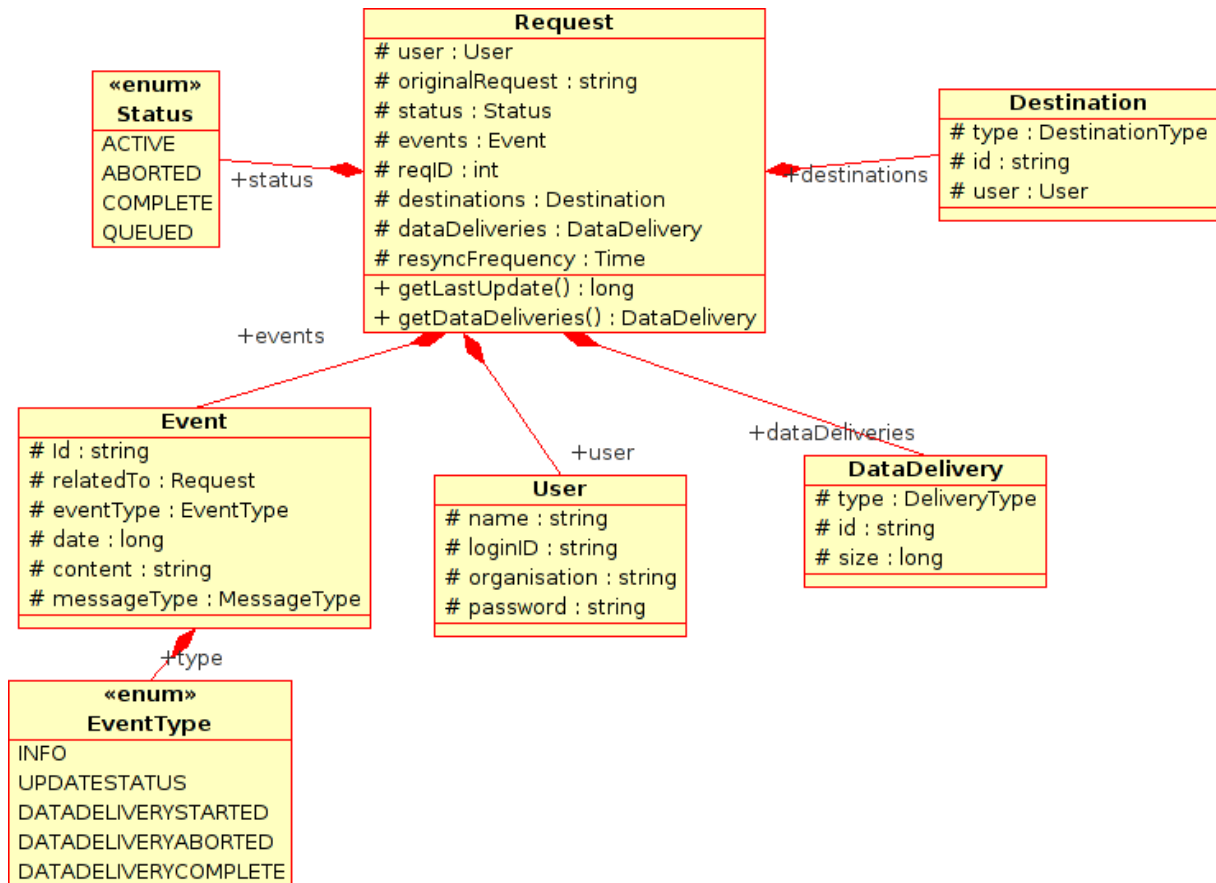


Figure 2 Request Class Diagram

The Request is articulated around 5 concepts or information:

- **Status** A request is always qualified by a state: QUEUED, ACTIVE, ABORTED or COMPLETE. The request follows a life cycle from QUEUED to one of the finished state ABORTED or COMPLETE. A subscription request producing recurrently some data (daily for example) will stay in the ACTIVE state.

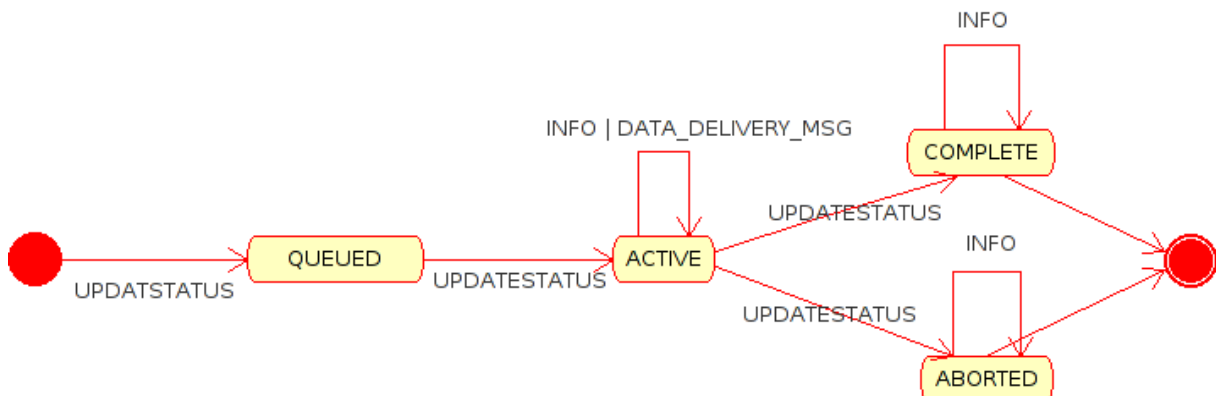


Figure 3 Request Life Cycle

- **Event** A request receives events and keeps track of them. There are 5 different types of events: INFO to report information such as logging messages, error messages; UPDATESTATUS to modify the request status (ACTIVE,...) and DATADELIVERYSTART,

DATADELIVERYABORTED, DATADELIVERYCOMPLETE to indicate when a Data Delivery related to the current request has started

- **User** Each request is attached to a user. The user can cancel, monitor or modify the request.
- **Data Delivery** A Data Delivery is created when some produced data has been delivered to a Destination. A Data Delivery is added to a request when a DATADELIVERYSTART event and a DATADELIVERYCOMPLETE event have been sent to the Catalogue Node. A request can be associated to several Data Deliveries as some recurrent data sources, such as the ones dealing with the observations, that constantly receives new data items. In this case, for each observation received, two new events will be received (note that to minimize the number of events created, a certain number of observations could be represented by only one Data Delivery). If a Data Delivery fails a DATADELIVERYABORTED event will be produced and the delivery might be re-queued as defined in the delivery policy of the dissemination system. A Data Delivery is associated to Destination such as the shopping cart of the user or a remote destination like an FTP server.
- **Destination** A destination defines where the data has to be delivered. It contains a data transport mode plus some associated configuration parameters. For example it could be the shopping cart of the user and in that case the user will go to his shopping cart to download the available and produced data. It could also be a FTP, a SFTP or an HTTP server or an external dissemination system such as DIFFMET the dissemination system of Meteo-France. A user can create as much destinations as he/she wants and test the destinations by receiving a test file.

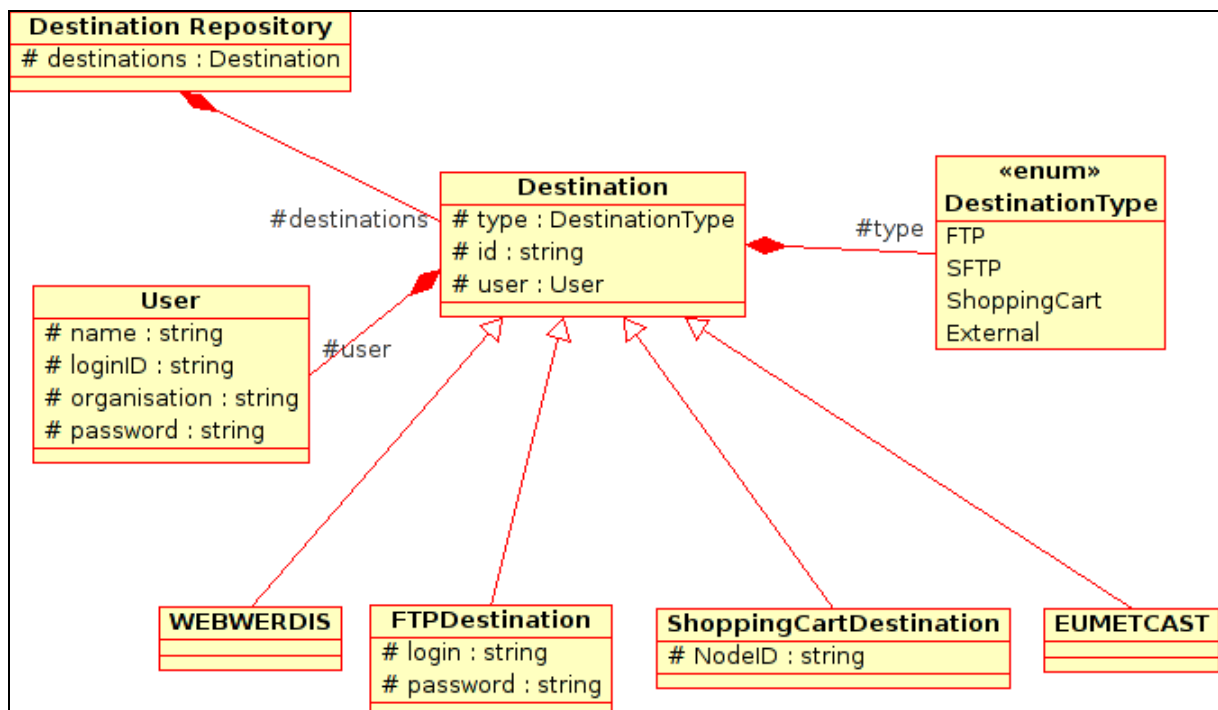


Figure 4 Destination Class Diagram

3.3 Event Model

At the moment the Catalogue Node contains a scheduler, which regularly spawns tasks to check the status of all running requests. These tasks send “GetSubmitStatus” XML messages to a local Data Repository or through the mesh network, to the Catalogue Node that owns the Data Repository hosting the data. This model was sufficient to demonstrate the features of the VGISC framework, but will not scale in a real-time context. An event model has to be adopted to manage efficiently in real-time events generated by on-demand requests or subscription requests.

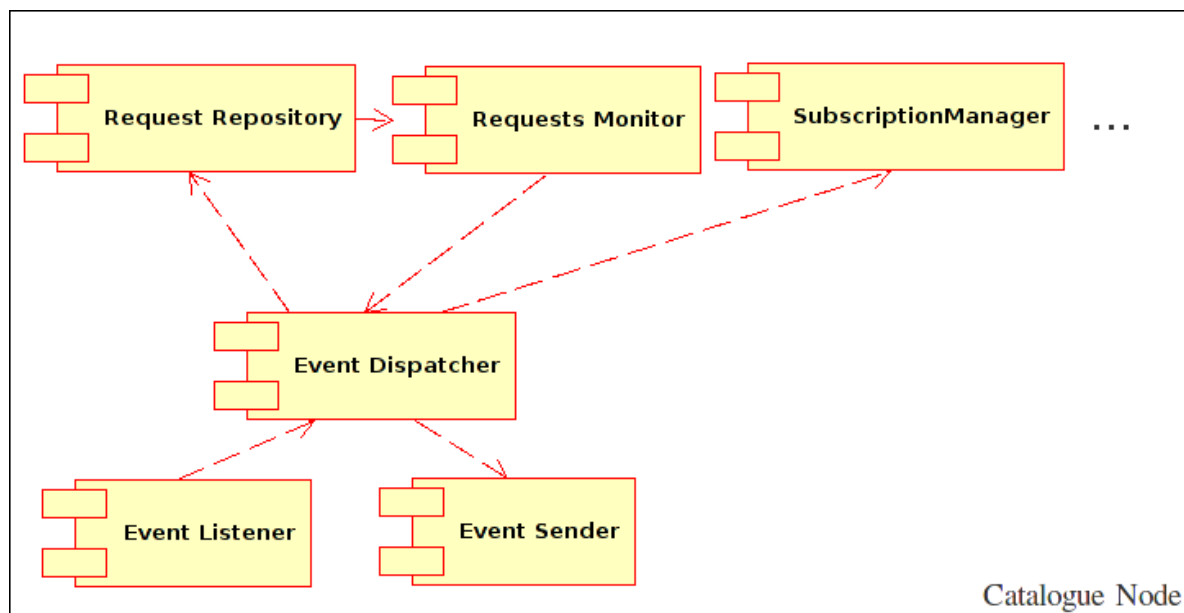


Figure 5 Catalogue Node Event Components

The new Catalogue Node contains an Event Listener that is listening for new events. The events will be XML Messages containing an Event Type (see diagram below), a Message Type, a content and a request ID to relate the event to a request. *Note that the design will most probably be refined and there might be a sub-class for each of the different events.*

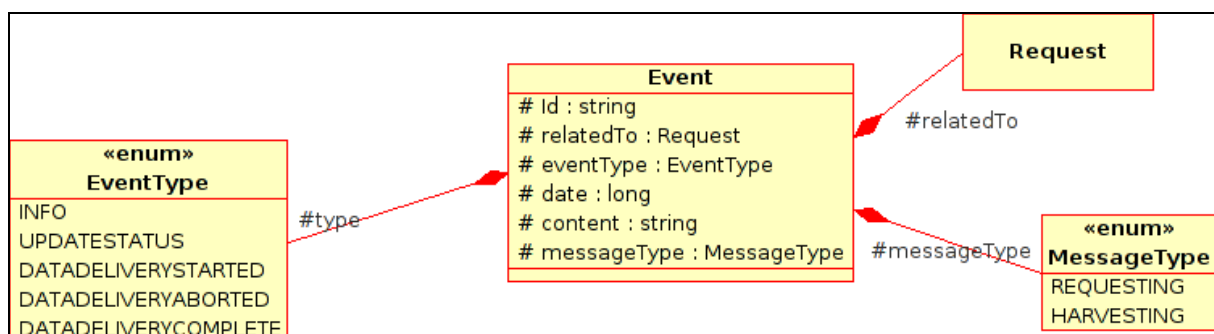


Figure 6 Event Class Diagram

The *EventListener* is a queue which could be transient (in memory) or persistent (on disk). Every new event once extracted and validated is sent to the *EventDispatcher* that dispatches it to the component responsible for processing the message. The *EventDispatcher* also receives

any outgoing messages that have to be sent by one of the main components (*SynchronizationManager*, *MetadataManager*, *RequestRepository*, *DomainAuthority*, etc).

Symmetrically, the Data Repository will embed the same components to send and receive events regarding the on-demand or synchronization requests.

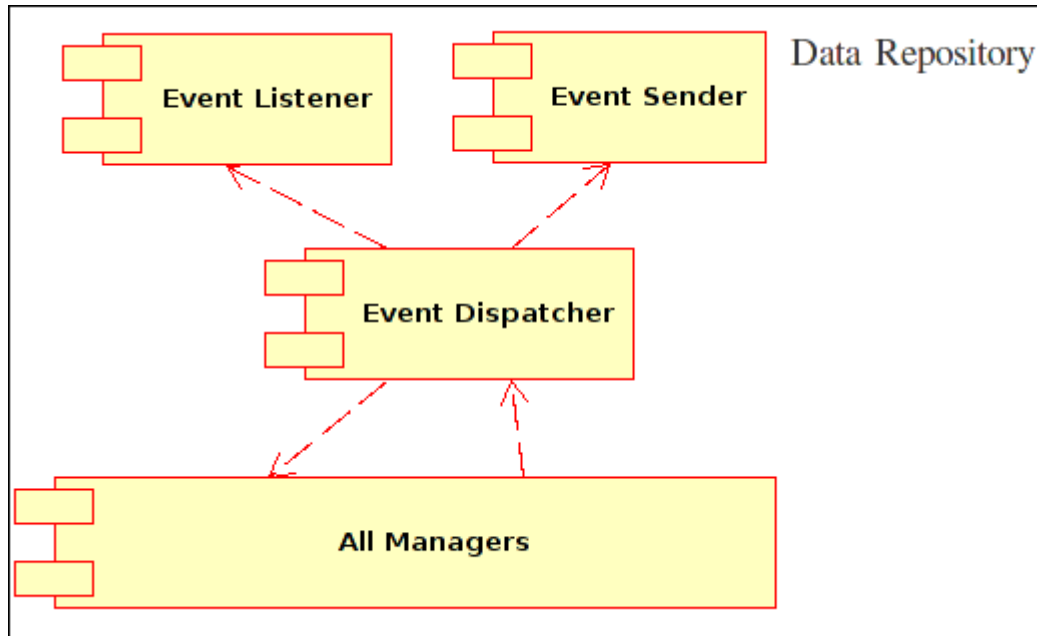


Figure 7 Data Repository Event Components

3.3.1 Catalogue Node and Data Repository out of sync

As explained above, the Catalogue Node maintains subscription requests and constantly updates them with the events received from the Data Repository (DR). In some cases like for example a crash of the Data Repository or power failure at a site, the Request Repository of the Catalogue Node could become out of sync with the state of the subscription requests in the Data Repository. Two different cases are possible:

- The Catalogue Node maintains a subscription that is not known by the Data Repository. In this case the subscription request must be resent to the Data Repository.
- The Data Repository has a subscription, which is not known by the Catalogue Node (CN), and the CN receives events linked to an unknown subscription request. In this case the Catalogue Node must send a request deletion message to the Data Repository.

A solution to these problems would be to build a system that guarantees a consistent state between the CN and the DR (using persistent queues, checkpoints, and acknowledgment messages) but such a system will be very difficult to put in place, and will have an effect on the overall performance of the system.

It is much easier and safer to implement a component that check the state of the subscription and resynchronize the CN and DR when necessary. This will be the task of the RequestMonitor component running within the CN. The RequestMonitor will have to rely on some defined criteria to decide whenever a request is out of sync. For this reason, every request will have a resynchronization frequency (define by the data provider when the

Subscription Source is set up) and if the Catalogue Node doesn't get any news regarding the requests within the time window defined by the resynchronization frequency, a "Give me an update on this subscription request" message will be sent to the Data Repository in charge of the subscription. If the Catalogue Node keeps receiving some events regarding an unknown subscription request, a request deletion order will be sent to the DR.

3.4 Subscription Information in the metadata definition

The newly created Subscription Service needs to be interfaced with the existing data sources and dissemination systems of each partner. Each data source provides its own way and own language to create data retrievals tailored to the user's needs. The VGISC infrastructure builds on these facilities to offer to its users the facility to sub-sample a dataset. In addition, data dissemination is part of the core activity of each meteorological centre, as they need to deliver in a very short time window the produced data to their clients. Each of these production dissemination tools provides a particular interface and the VGISC Subscription Service needs to let them keep their own procedures and interfaces. For example DIFFMET from Météo-France uses the concept of diffusion plan and diffusion channels as ECPDS from ECMWF uses the concept of destinations and products.

Using the same solution as the one previously adopted for the on-demand data requests, the VGISC Subscription Service leaves it to the data provider to define the different parameters that the user will be allowed to select for creating his subscription requests. The data provider can then include within the VGISC subscription request, parameters that are only relevant to his/her dissemination system.

The metadata describing the dataset needs to contain all necessary information regarding the Subscription Service, which can deliver this dataset. A user's interface (web interface) will be dynamically created from this information and the user can choose some of the exposed parameters to refine its subscription request. The subscription metadata information goes into the service metadata that is currently in the <vgisc> tag.

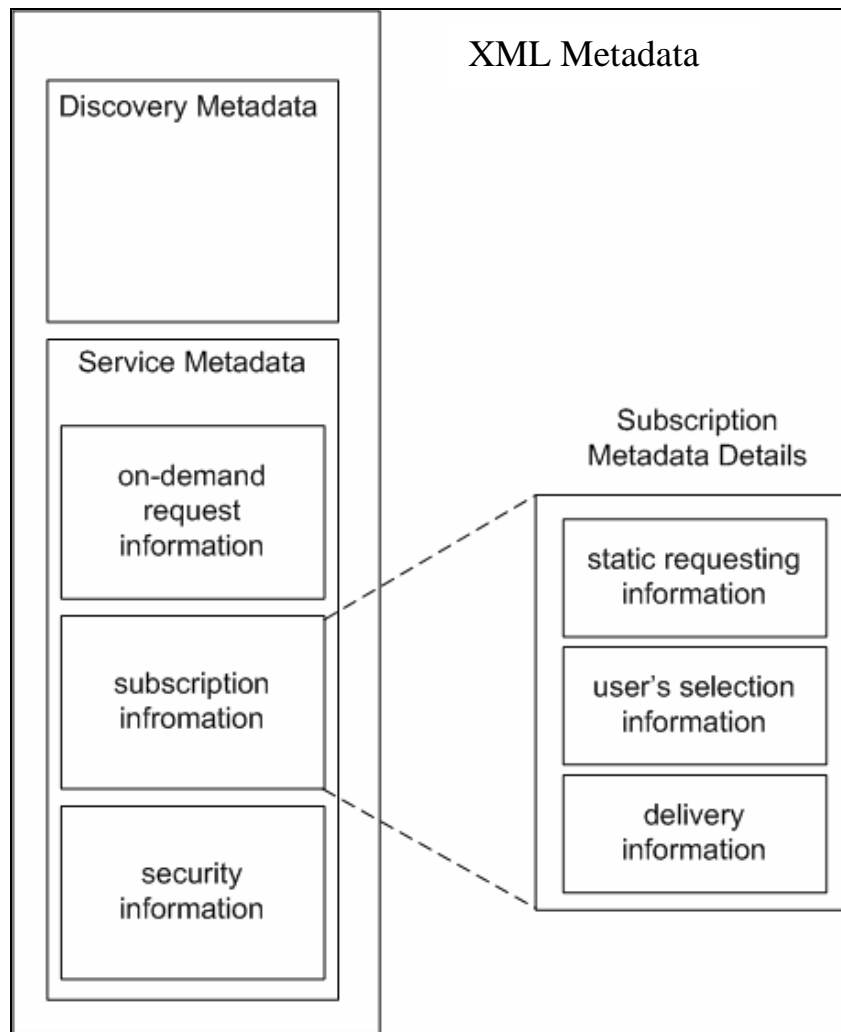


Figure 8 metadata stack

The subscription part of the service metadata contains the following information:

- `<static-info>` which includes mandatory parameters into the subscription request which will be sent to the Data Repository;
- `<selection>` that contains xml selection components (enum, freeText, etc) rendered as widgets in the user's interface and providing to the user some ways to tailor his/her selection;
- `<destinations>` which defines what kind of deliveries are supported by the Data Repository for the dataset.

```

<v.vgisc xmlns:v="http://www.vgisc.org">
...
<v.subscription>

<!-- static information which will be passed to the data repository for all subscription requests -->

<v.static-info>
<!-- free tags here -->
  <dataset xmlns:ec="http://int.ecmwf.vgisc">int.ecmwf.era40</dataset>
</v.static-info>

```

```

<!-- to represent the user's selection -->
<v.selection>

<!--All different sorts of widgets enum, freetext, temporal, geographical -->

<v.enum name="parameters" title="parameters" multiple-selection="yes">
<v.value title="la temperature" group="firstGroup">temperature</v.value>
<v.value title="le geopotentiel" group="secondGroup">geopotential</v.value>
<v.value title="la pression" group="secondGroup">pressure</v.value>
</v.enum>

<v.freeText name="title" title="my beautiful plot" nbMaxCharacters="128"/>

...

</v.selection>

<!--Destination configuration : supported destination (predefined type or free type for bespoke
dissemination systems -->
<v.destinations>

<!-- Internal delivery: ftp, email, etc -->
<v.destination description="Ftp Delivery" type="ftp"/>
<v.destination description="Email Delivery" type="email" >

</v.destination>

<!-- External delivery: webwerdis, diffmet, ecpcds, gts -->

<v.destinations description="WebWerdis Delivery" type="webwerdis">
<v.enum name="parameters" title="parameters" multiple-selection="no">
<v.value title="channel1">ch1</v.value>
<v.value title="channel2">ch2</v.value>
</v.enum>
</v.destinations>
</v.destinations >

</v.selection>
</v.subscription>
</v.vgisc>

```

Figure 9 Subscription Metadata

An XML request will be created and passed to the data source handling the subscription. This XML request will be composed of the user's selection serialized in XML, the <static-info> coming from <subscription> and the <destination> information as selected by the user. The created request will be saved as part of the Request object in the RequestRepository.

```

<v.subscription-request xmlns:v="http://www.vgisc.org/" id="int.ecmwf.cn/1234567" >

<!-- Static information from the subscription metadata -->
<v.static-info>
<dataset xmlns:ec="http://int.ecmwf.vgisc">int.ecmwf.era40</dataset>
</v.static-info>

```

```

<!--user's selection serialized in XML -->
<v.selection>
  <v.list name="parameters">
    <v.value>temperature</v.value>
    <v.value>pressure</v.value>
  </v.list>
</v.selection>

<!--destination as selected and defined by the user -->
<v.destinations>
  <v.destination id="jeanMouchard-ftp-dest1" type="ftp">
    <v.hostname>external.meteo.fr</v.hostname>
    <v.login>diss</v.login>
    <v.pass>xxxxxxx</v.pass>
    <v.dir>/dissemination/simdat</v.dir>
    <v.options>
      <v.passive on="yes"/>
      ...
    </v.options>
  </v.destination>
</v.destinations>
</v.subscription>

```

Figure 10 Subscription Request Example

3.4.1 Destination, DestinationRepository

Within the Catalogue Node, the DestinationRepository is the Destination database. It stores definitions, which are XML documents describing what kind of information need to be filled in by the user and what kind of selection interface has to be presented to the user.

If the metadata information doesn't contain any definition information and if the Destination has been registered, the user's interface will be built from the definition information stored in the DestinationRepository.

```
<v.destination description="Ftp Delivery" type="ftp"/>
```

Figure 11 destination information referring to the predefined information in the DestinationRepository

Subscription types can therefore be represented in the metadata records with a "reference" (ex: scope="internal" type="email" to use the default VGISC email delivery). This will avoid repeating the information in all the metadata for the datasets sent using the same delivery mechanism and it should therefore minimize the size of the metadata records and improve speed and efficiency.

An administrator is allowed to register new destination types in the DestinationRepository. In that case, he will have to provide an XML destination definition, which contains the needed information and a mechanism to call the component in charge of the delivery (a class name for example).

Using the DestinationRepository, the user can also create and store his destinations from the registered destination's types. The DestinationRepository therefore act also as a database for storing user's destination particular instances. The user will be allowed to select one of his

predefined destinations when creating its subscription requests. If the user has no predefined destinations, he will be allowed to create one of the fly that will be stored into the DestinationRepository.

4 Subscription Service implementation within a Data Repository

The meteorology activity intends to provide the Subscription Service within the Data Repository as a turnkey solution for the metrological centres that have no particular specificities or needs. Using this solution meteorological services should be able to integrate their disseminations applications and data sources by adding a lightweight scripting glue (Shell script, python, perl, ...) within the Data Repository.

For the meteorological centres using legacy applications with specific requirements, for example, old bespoke interfaces, a framework of object-oriented components with a comprehensive API will be developed. From there the developers should be able to build tailored-made components and deploy them within modified Data Repository.

The turnkey version will be based on the developed API and component framework.

4.1 Event Model Support in the DataRepository

The Data Repository needs to be extended to support the event model (push model). Using this mechanism, the components willing to send events to the CN will subscribe to the Event Sender.

4.2 Generic API and component framework

Note: that at this stage the components used so far in the Data Repository for solving the on-demand data requests requirements will be kept but it is very likely that these will also make use of the subscription components.

The transport layer of the new Data Repository, including the newly developed SubscriptionService, is not going to change and will be still based on XML communication over HTTP/HTTPS. A REST model has been used since the beginning of the project and up to 6 messages have been defined to implement the on-demand Request Service, the Metadata synchronization Service and the Downloading Service (see *D.20.1.3 Demonstrator-Phase-I-Technical-Design*).

Three new messages needs to be defined to exchange some information related to the subscription between the Catalogue Node and the Data Repository. These messages are detailed below following a functional representation:

- **Subscribe(in request)** : Message used by the CN to send the subscription request. The request object contains a requestID defined by the CN, the subscription selection request and the destination information. The subscription requestID is composed of the subscription source and an id defined by the CN. This message can also be used to modify an already existing subscription request;
- **DeleteSubscription(in requestID)**: Message used by the node to delete a subscription;

- **GetSubscriptionList(out requestsList):** Returns the list of existing subscriptions. This will be used to synchronize the subscription database in the Node with the database in the Data Repository;
- **getSubscriptionInfo(in requestID, out request):** Returns to the node, the request information stored in the Data Repository Database.

Following a designed implemented in the current version of the Data Repository, these XML messages will have to be mapped in Action objects (see *D.20.1.3 Demonstrator-Phase-I-Technical-Design*) that will make use of the Subscription components through the SimpleDataRepository abstract class.

4.2.1 Subscription components

A subscription source needs to implement the SubscriptionSource interface. The interface contains methods to subscribe to the source with a particular request expressed in XML, to manage the subscription (delete, modify), to get some information regarding the existing subscriptions (checkStatus, getSubscriptionInfo) and to receive events regarding the subscriptions managed by this source.

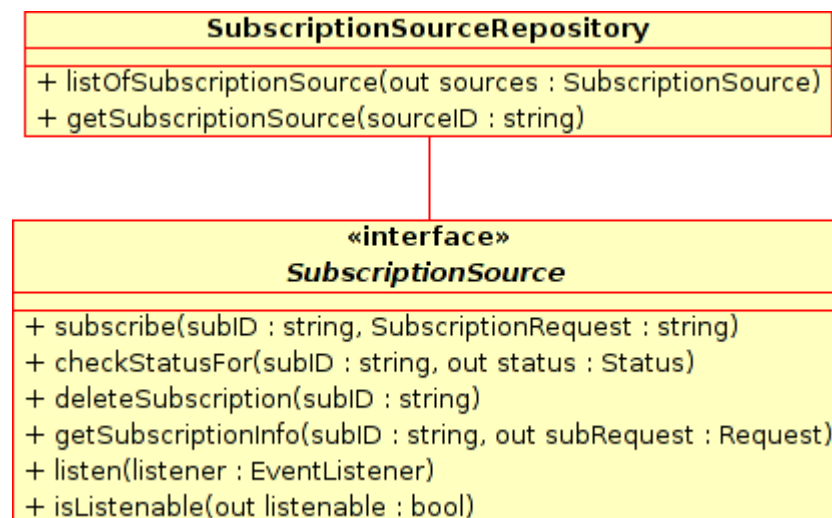


Figure 12 Subscription Request Example

If the Subscription Source is external (for example when the subscription is handled by ECPDS, DIFFMET, WEBWERDIS), a small module implementing the Subscription interface needs to be developed to link the VGISC infrastructure to this external source.

If the Subscription Source is internal, a set of components developed behind the SubscriptionSource interface will implement the subscription from the data arrival to the data delivery.

An implementation handling the real-time observations (GTS observations) will be developed as a proof of concept.

4.3 Internal Subscription Source

A Subscription Source receives some data (or is aware of the data arrival from some sources) and needs to link these data arrivals with the user's subscription in order to deliver the data to the user as soon as it is available. A Subscription Source has the following components:

- **Subscription Database:** This database maintains the user's subscription and stores all the events received to related the subscriptions (data arrival, data delivery start, ...). Each Subscription Source has its own subscription database. This database will most probably be a relational database;
- **Data Source:** The data source generates an event every time a data has arrived. This event contains a description of the data content. The data source has also some mechanisms to give access to the data;
- **Subscription Matcher:** This component accesses the Subscription Database to extract the user's subscription order (what the user's wants). It also receives events from the Data Source every time a piece of data has arrived. With these two pieces of information, the Subscription Matcher decides if the newly arrived data should be related to existing subscriptions and has to be sent to the users;
- **Data Delivery:** This component delivers the data to the user. In the case of the shopping cart, the data will stay in the Data Repository until the user downloads it and an event ordering the CN to insert the data arrival into the user's shopping cart is sent. This module could also only be a proxy linking the VGISC world with a specific delivery system;
- **Subscription Manager:** This component orchestrates the dialog between the four components described above.

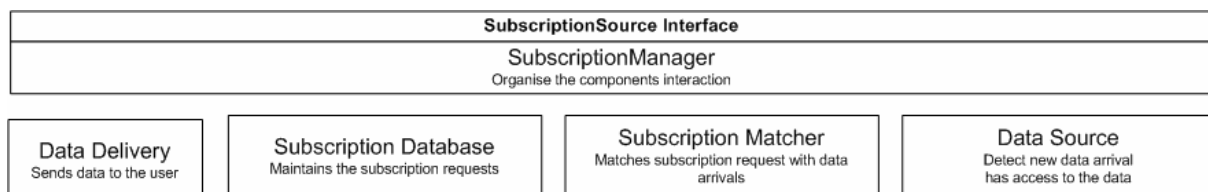


Figure 13 Subscription Request Example

4.4 A file system based internal Subscription Source

A Subscription Source with a Data Source based on a file system will be built as a reference implementation. This implementation will be used to provide subscription on the GTS real-time information.

The Data Source receives the observations as files in a directory. The file names follow a naming convention defined by WMO that details the data content.

Using this naming convention, all the files will have the following name format:

```
pflag_productidentifier_oflag_originator_yyyyMMddhhmmss[_freeformat].type[.compression]
```

This information is sufficient to link the user's subscription with the data arriving. Indeed in the case, the user's subscription request will contain the product identifier, the originator and the data type.

Every time a new data will arrive, an event containing the filename will be generated.

```
<event>
  <type>DATA_ARRIVAL</type>
  <name>A_HPWZ89LFPW131200RRA_C_LFPW_20020913160300.bin</name>
</event>
```

Figure 14 Possible Event for a NWP output arrival from France

The role of Subscription Matcher is to link the user's subscription with the data arrivals. In order to send the most rapidly the data to the user, the Subscription Matcher needs to preload all the request subscriptions and transform them into a pattern which can be quickly matched with the data arrivals.

In the first implementation, the Subscription Matcher will transform the XML subscription requests into regular expressions.

For example, the regular expression `A_HPWZ89LFPW131200RRA_C_LFPW_[0-9]+.bin` will match the filename defined in the above event. Every time one of the regular expressions is matched to the arriving data, a message is sent to the relevant delivery agent to send the data to the user. Some events are also generated and sent for logging purposes to the CN in charge of the subscription (INFO[data abc has arrived], DELIVERY_START[start delivery for data abc] ...).

If the solution based on regular expressions happens to be too slow, a system of bit mask and vector of bits could be used.

5 Evaluation/Validation Report: Prototype PM36.

5.1 Grid Infrastructure

The Grid infrastructure has been enhanced with several developments to finalise interoperability requirements and provide access through standardised interfaces. The main work produced in this area has been towards the implementation of a Web Services API's between the Clients (the Portal) and the Node. The communication between these two components is based on SOAP messages that implement the previous functionality of the portal. Web Services interfaces provide controlled access to resources deployed at the Node and described via WSDL documents. The transport of SOAP messages has also been used to include security headers through libraries such as wss4j, which provide an implementation of WS-Security.

	Browse/Search	Metadata Display	Data Selection	Data Request Order	Data Request Monitoring
Public Interface	✓	✓	X	X	X
Private Interface	✓	✓	✓	✓	✓

Figure 15 Services deployed at the Node available to the portal through SOAP/WSDL.

The messages interchanged across the rest of the components Node-Node and Node-Data Repository follow a REST style approach. These are messages created according to the VMC protocol (See *D.20.1.3-Demonstrator-Phase-I-Technical-Design*); those responding to data requests actions have been updated to include security headers in the same way as in WS-Security.

Versioning of these messages has been done in a controlled manner with dedicated API's so that the GRID infrastructure can deal simultaneously with several versions.

Finally the last developments towards interoperability have been focused in the integration of datasets served through OpenDAP data servers (www.opendap.org). This work has been carried out in collaboration with the Korean Meteorological agency.

5.1.1 Requirements Validation

The new GRID infrastructure as of PM36 has been redeployed in a test-bed including several nodes at ECMWF and a remote node in NEC. A full deployment involving the rest of the sites is planned for the forthcoming months.

The deployment has been tested connecting one of the nodes of the new infrastructure to the old one to ensure backwards compatibility with the rest of the nodes, in such a way that

redeployment by each of the partners can occur at any given time ensuring a smooth transition. **(Requirement Meteo P3.2)**

Access to OpeNDAP data servers is possible through the portal deployed in the Korean Meteorological Agency (KMA). **(Requirement Meteo P3.1)**

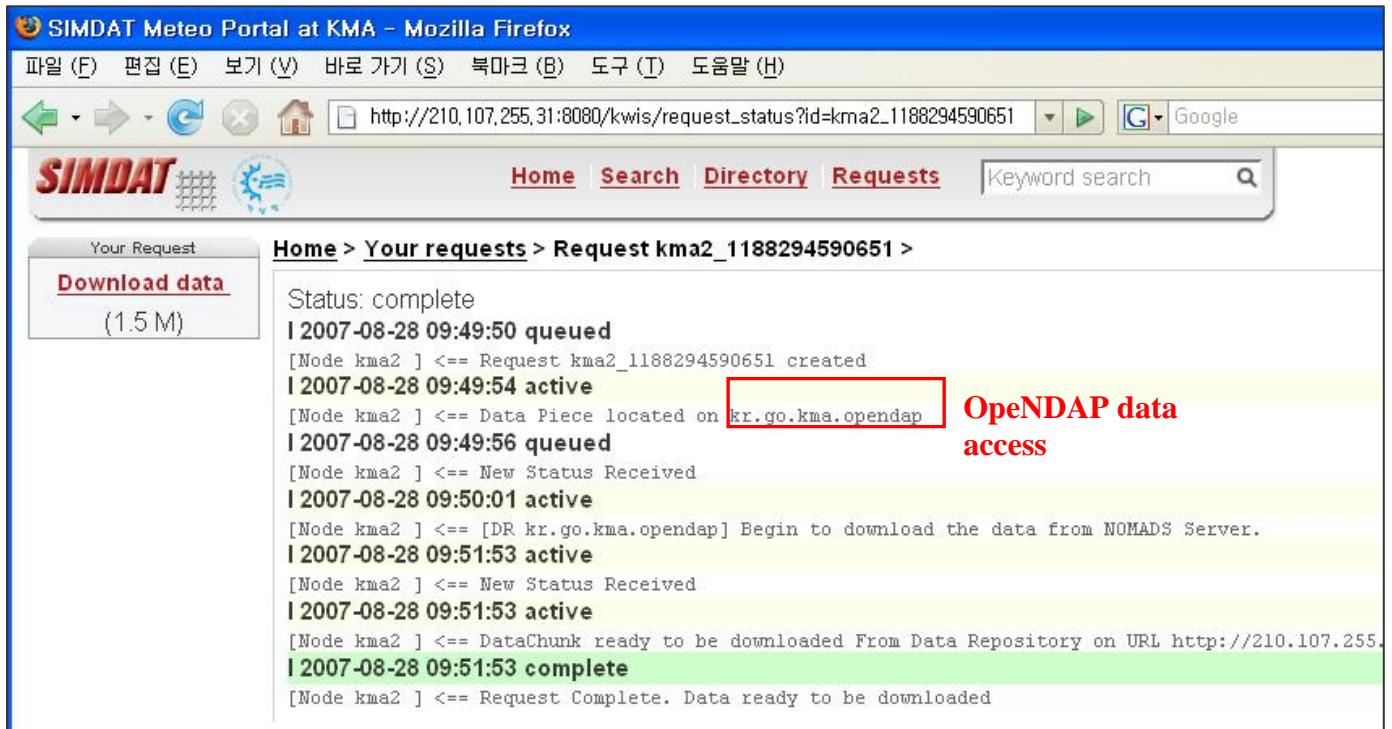


Figure 16: Access to OpenData Datasets

Grid Infrastructure Achievements	
The support for web services at the Node level allows portals to be integrated via standardised interfaces providing new potential partners with a better description of the services published at the node.	
The new infrastructure is compatible with the previous version, providing a backwards compatible release.	
The infrastructure provides access to datasets served by the OpeNDAP data server.	

Requirements checklists:

Grid Infrastructure Requirements				
Requirement	Description	Priority	Implemented	Remaining
Adopt Standard Interface	Integration of web-service interfaces to ease interoperability.	High	✓	
Interoperability Enhancement	Provide access through the	High	✓	

METEO-P3.1	infrastructure to datasets served through OpeNDAP			
Backwards compatible software releases METEO-P3.2	Provide compatibility with previous components	High	✓	
Push mechanism for critical data. METEO-P3.3	Implementation of push interfaces for weather warnings.	High	Ongoing	Implementation for PM42

5.2 Distributed Data Access

Performance enhancements have been done to the synchronisation engine to minimise the initialisation phase, time for a node to obtain the whole catalogue when it plugs into the infrastructure for the first time. As metadata records are synchronised, these are compressed during the transport minimising the times that records take to reach a given node. Once records reach their target, they have to be parsed to generate the indexes for the searches. The process has been improved by performing the indexing in an asynchronous way and by updating the xml parsing modules to make use of the lazy loading DOM libraries, loading xml records in memory only when needed.

In the area of large data transports protocols, a study carried out by INTEL has been performed to analyse the key protocols in this area. The report is part of the deliverable *D.3.3.1 Consolidated report on Data Access*.

5.3 Validation

Synchronisation engine performance improvements: (**Requirement METEO-P3.13**)

Function	Details	Time
Metadata Temporary Storage on Destination	Metadata Stored in a Persistence Container before to be ingested	6 min 07 s for 27834 elements
Ingestion	Metadata indexing and storage in the destination catalogue.	20 min to index 27834 elements by batch of 500 elements. (previous times for the same amount of datasets aprox. 40 min)

DDA Achievements
An study of the key large data transports protocols has been performed
Performance enhancements in the synchronisation engine results in better timings for the metadata transfers and the initialisation phase.

DDA Requirements				
Requirement	Description	Priority	Implemented	Remaining
Data Replication/ Caching for Real time data	Provide replication of real-time data to ensure high availability	High	Ongoing	Implementation for PM45.
Efficient transport of large amount of data	Provide an study with protocols supporting large amounts of data transfers	High	✓	
Synchronisation engine performance improvement	Improve the synchronisation engine to provide better timings in the initialisation phase	High	✓	
Acquisition of real time data	To acquire real-time data from the GTS Message Switch Systems	High	Ongoing	Implementation for PM45

5.4 Discovery, Ontology and Virtual Organisations

Within the ontology area, new developments have included the ontology navigation into the portal to complement the advanced search. The navigation browses through the Thesaurus developed during the second phase of the project, displaying the relationships between meteorological terms. This navigation will potentially help the end user to identify new terms that will guide him through the discovery of datasets.

A Thesaurus entry tool was also developed to introduce or maintain terms to existing Thesaurus such as the AMS or the GCMD or to the SWEET ontology. The tool also introduces terms in different languages to support multilingual labels.

The searching engine behind the discovery service has been improved in several aspects:

- The text searches provide support for Boolean operators and more sophisticated queries can be formulated by the end user.
- Scoring mechanisms have been improved to give more weight to textual descriptions.
- In the case of geographical searches a ratio is calculated based on the area searched compared to that of the dataset coverage, giving more weight to those datasets that expand over similar geographical areas (e.g. points size datasets have a higher score when the user is searching for a single location, while larger area have a lower score)
- Text field searches have been introduced, in the same style as Google fields. e.g. Search for keyword ozone in the Title only.

The Virtual Organisation has been the main block of development for the last six months. The development process has gone through two iterations delivering versions of the Virtual Organisation as described in D12.2.3.

Security components have been developed and integrated in the existing infrastructure in order to support the VO.

- Authentication web service: Service deployed in the Node and Client integrated at the Portal. The security information is added in the SOAP headers as specified in WS-Security, supporting schemes for user name and password authentication and certificate-based authentication.
- The database deployed in the Catalogue Node has been updated to include user details and Domain related information.
- Authorisation engine: The VGISC will have to handle very different schemes of authorisation. A generic engine has been built to be able to support plug-in modules supporting different requirements. As an example, NEC will contribute with an authorisation module based on the DAC framework which provides a standardised interface based on XACML and WS-Trust.
- Domain Authority component: This component has been deployed in the Node to generate tokens that are ported across organisations ensuring Trust Domain relationships are fulfilled and giving proof of user's roles in a remote location. This module will be reviewed and refactored by NEC as a STS (Secure Token Service).
- Update of the Metadata: The metadata has to carry information about the policies allowed to access the dataset described within the terms of the metadata.
- Update of the VGISC message protocol: The messages interchanged between the Catalogue Nodes and the Data Repository have been updated to include a security header when restricted actions take place.
- Management tools: Many of the complexities of the security integration come at the time when the VO is set up. The use of PKI infrastructure adds into this complexity, whereby issues like - keys distribution, support for multiple CA's and users not being security-literate with awareness and understanding of certificates itself - can jeopardise the correct use and deployment of the whole security infrastructure. For these reasons a special emphasis has been made to develop management tools that handle these issues where possible. A framework to register user, roles and domains and to publish domain keys has been developed supporting these functionalities from a web interface accessible through the Node and through command line clients.

The library implementing WS-Security (wss4j) provides a very static configuration with no support for dynamic load of certificates or certificate chains (certificates issued by a chain of Certification Authorities). In this respect, some certificates issued by toolkits like MyProxy, (e.g. Proxy Certificates include a delegation mechanism involving a chain of several certificates: root, user and a self signed certificate) are not straight away supported by the wss4j libraries. Extra development defining custom Handlers is required to support certificate chains.

5.4.1 Validation

The portal provides controlled access to datasets that are restricted to registered users with the right Domain Roles. Authorisation decisions are done provided the policies described in the metadata intersect with the ones assigned to the user. **(Requirement METEO-P3.7)**

The screenshot displays the Simdat-VGISC portal interface. At the top, the user is logged in as 'guest', with a 'Logout' button. The main content area shows metadata for a dataset, including a title, abstract, period, and bounding box. A world map is displayed below the bounding box. The 'Policies' section lists 'ecmwf_countries realtime' and 'wmo.essential'. A 'Please note' box on the left indicates that access to the dataset is restricted. The 'Retrieve Data' button is also visible.

User Authentication

You are logged in as user **guest**. **Logout** Home Search Directory Requests

Restricted Access

Please note:
The access to this dataset is restricted.
Your don't have the necessary credential to access it.

Metadata updated with policies.

Policies: [ecmwf_countries realtime](#)
[wmo.essential](#)

Figure 17: Security Components added to the portal interface

Registration of users, domains is possible through the Management interfaces available from Web portal or through command line clients. **(Requirement METEO-P3.8).**

SIMDAT Administration tools
You are logged in as user **simdat**. [Logout](#)

[Show Users](#)
[New User](#)

[Show Roles](#)
[New Role](#)

[Show Domains](#)
[New Domain](#)

List of registered users

Login name	Full name	Email	Information	Roles
guest	Guest user	guest@ecmwf.int	Generic guest user	
vmc_test	Test user	vmc@ecmwf.int	This is a test user	ecmwf_countries.realtime

```

vmc-node-v0.9.8/bin# ./vmc.sh tools AddDomainMember -help

vmc.sh tools AddDomainMember [options...] arguments...
  -domain MyDomain           : domain name
  -help                       : print usage message
  -keystore MyKeystore       : keystore containing the site certificate
  -keystoreAlias MyAlias     : alias of the certificate (or public key in the keystore)
  -keystorePassword MyPassword : keystore Password
  -keystoreType JKS          : keystore type (JKS for example)
  -site ECMWF                : site name. (self to add the current node certificate in the passed domain)
  -url http://...           : URL from where to download the certificate

-----
ex: Add the local site in the domain theDomain:
>./vmc.sh tools AddDomainMember -domain theDomain -site self
ex: Add an external site in a domain using an exported keystore:
>./vmc.sh tools AddDomainMember -domain theDomain -site A$ite -keystore /tmp/mykeystore.jks -keystoreAlias MyAlias -keystorePassword MyPassword -keystoreType JKS
ex: Add an external site in a domain and getting the certificate using a url:
>./vmc.sh tools AddDomainMember -domain theDomain -site A$ite -url http://simdat-cn1.ecmwf.int:9005/export/getDomainPublicCertInBase64

```

Figure 18: Administration Tools to Manage the VO

Portal has integrated the developments done in the ontology area and the discovery service. Providing better results hits on the searches and aided navigation through ontological concepts. **(Requirement METEO-P3.10a/b)**

The screenshot shows a search interface with the following components:

- Search >**: A header label for the search section.
- Keywords:**: A text input field for entering search terms.
- Ontology Keyword Selector**: A panel with a scrollable list of terms.
 - precipitation**: A category header.
 - insert**, **or**, **and**: Buttons for logical operators.
 - super**: A section containing the term **atmosphere**.
 - sub**: A section containing terms **dew**, **frozen precipitation**, and **rain**.
 - linked with**: A section containing terms **atmosphere** and **dew**.
- Language Selection**: A dropdown menu currently set to **English**, with a list of options: English, French, German, and Spanish.
- Location:**: A world map with navigation controls (directional arrows, zoom in/out, and a globe icon). Below the map are input fields for coordinates: **N** [], **E** [], **W** [], and **S** [].
- Period:**: A section with **From:** [] **To:** [] (yyyy-mm-dd) input fields.
- Actions:**: A section containing **Reset** and **Search** buttons.

Figure 19: Ontology navigation integrated at the portal.

Thesaurus maintenance service prototype has been developed to maintain and introduce new terms to the Meteorological Thesaurus. **(Requirement METEO-P.3.14)**

The screenshot shows the SIMDAT Thesaurus Maintenance Service interface with the following elements:

- SIMDAT**: The logo of the system.
- Navigation Menu**: **Info**, **Test**, **Save Model**, and **New Concept** buttons.
- Language and Search**: A dropdown menu set to **en** and a search input field containing **fore**.
- Found Labels**: A list of search results, including:
 - forecast
 - forecast amendment
 - forecast area
 - forecast chart
 - forecast district
 - forecast error
 - forecast evaluation
 - forecast lead time
 - forecast period
 - forecast updating
 - forecast verification
 - forecast-reversal test
 - forecaster
 - forecasting error
 - Forel scale
 - forerunner
 - forest climate
 - forest climate
 - forest meteorology
 - forestry
- Annotations**:
 - An arrow points from the text **language of labels** to the **en** dropdown menu.
 - An arrow points from the text **found labels** to the list of search results.

Figure 20: Prototype of the Thesaurus Maintenance Service.

A prototype of the Metadata Editor conforming to the WMO Core profile has been deployed to help scientist in the definition of Metadata to describe meteorological datasets. **(Requirement METEO-P.3.11)**

Home > Metadata VGISC Part Editor > ecmwf_Plots_vgisc.xml

Save Save Draft Preview All Cancel

Mandatory Fields Requests Variables

Hide Preview Hide Form Add Widget...

Date Delete

Name
Date Single Date

Title Start Date End Date
Temporal Coverage 1980-01-01 1990-12-31

Preview:
<v:vgisc xmlns:v="http://www.vgisc.org/">
 <v:variables>
 <v>Date type="date" title="Temporal Coverage">
 <v:startDate>1980-01-01</v:startDate>
 <v:endDate>1990-12-31</v:endDate>
 </v>Date>
 </v:variables>
</v:vgisc>

Temporal Coverage
1980-01-01 1990-12-31

Figure 21: Metadata Editor Prototype.

VO and Ontology Requirements				
Requirement	Description	Priority	Implemented	Remaining
VO Assessment and consolidation METEO-P3.6	Deploy the VO infrastructure at all partners	High	90%	Deployment at ECMWF and NEC, rest of sites will gradually deploy till PM42.
Metadata Integration of the data access policy. METEO-P3.7	Integrate data access policies in the metadata records	High	✓	
Development of tools for user and policy administration. METEO-P3.8	Develop administration tools to register users and policies within the Domains.	High	✓	
Address cross-organisational issues METEO-P3.9	Provide support for cross domain authentication and authorisation		80%	Full testing environment to be done with several domains at several organisations.
Monitoring Tools METEO-P3.9a	Develop tools to monitor the synchronization, data delivery and VO services	High	Ongoing	Implementation for PM45
Ontology based catalogue navigation METEO-P3.10a	Integration of the ontology in the portal	High	✓	
Discovery Service enhancement METEO-P3.10b	Text Search improved to provide better results.	High	✓	
Thesaurus Maintenance Service METEO-P.3.14	Implementation of a service to manage Ontologies	High	✓	
Metadata Editor METEO-P3.11	Implement web interface to generate metadata records compliant to the WMO.	High	✓	

Ontology/VO Achievements

Integration of Thesaurus and Ontologies in the portal provides better navigation through meteorological concepts, helping the user in the discovery of datasets.

The Virtual Organisation has been implemented and deployed at in a test bed. The components to provide a secured and controlled access to distributed datasets have been integrated within the infrastructure.

The full deployment of the VO across the present infrastructure involving all the partners and collaborators will be a challenge for the forthcoming months.

5.5 Analysis Services

During the reporting period the Meteo partners have been gathering requirements to deliver the design of the subscription service to deliver real time data as described in previous section of this document. Plans for implementation will be for PM37-PM42.

6 Prototype at PM45.

The prototype as off PM36 is getting close to its final implementation and fulfils most of the requirements that were defined at the beginning of the project. Each one of the phases has presented very challenging problems and this is also expected to be the case for the last months of the project. The main developments from now until PM45 will be oriented towards the implementation of the subscription service for real time data.

The Meteo partners also expressed the requirement for high availability of some the data, such as the observations. For these reasons, data replication was brought to the list of requirements for building a VGISC. A potential implementation to achieve replication of the observations will be proposed through the subscription service. In this way, a Node could subscribe to another Node to receive a replica of the data of interest.

These observations will be fed into the system through dedicated modules that interact with the operational meteorological network, making datasets and its metadata descriptions immediately available to the VGISC.

Finally, the system will benefit of some monitoring tools that will report on the status of some of the services such as synchronisation, subscription, etc...

7 Resulting Requirements for PM42.

Grid Infrastructure

- Push mechanisms for critical data

DDA

- Data Replication of critical data
- Acquisition of real-time data

VO

- Assessment and consolidation
- Address cross-organisational issues
- Monitoring Tools

Analysis Services

- Subscription Service

Name	Push mechanism for Critical Data		
Req. Id	METEO-P3.3		
Application Activity	Meteo		
Prototype(s)	Meteo Prototype for PM42		
Date Created	2006-09-10	Priority	High
Created By	ECMWF	Technology component	Grid Infrastructure
Status	Evaluation		
First Implementation Date	Phase III	SIMDAT module targeted	WP 12.3
Description	Among the meteorological data, certain types such as weather warnings or radar data need to be accessible from any Nodes of the network within 4 to 5 minutes. In such conditions a push mechanism will send the warnings with no delay in the system.		
Relation to prototype			
	Meteo PM45		
Requested functionality			
	Implement the push interface for critical data		
Validation			
	Synchronize in a matter of minutes the metadata describing the critical datasets. Make available the data from any Nodes in a matter of minutes.		
Assumptions			
	Build a solution authorizing to push metadata and data (full push mode) or push notifications advertising the arrival of new data and provide mechanisms to come and fetch the data (notify and fetch).		

Name	Replication/Caching for Real-time Data		
Req. Id	METEO-P3.4		
Application Activity	Meteo		

Prototype(s)	Meteo Prototype for PM45		
Date Created	2006-09-10	Priority	High
Created By	ECMWF	Technology component	DDA
Status	Evaluation		
First Implementation Date	Phase III	SIMDAT module targeted	WP 12.3
Description	To ensure high availability of some of the data, meteorological observations will be replicated/cached between the National Centres.		
Relation to prototype			
Meteo PM45			
Requested functionality			
Data replication for meteorological real-time data (such as observations)			
Validation			
Critical data available from at least two Nodes			
Assumptions			
The synchronization engine developed might be used for replicating the data			

Name	Acquisition of Real-time Data		
Req. Id	METEO-P3.15		
Application Activity	Meteo		
Prototype(s)	Meteo Prototype for PM36		
Date Created	2007-04-23	Priority	High
Created By	ECMWF	Technology component	DDA
Status	Evaluation		
First Implementation Date	Phase III	SIMDAT module targeted	WP 12.3
Description	To acquire real-time data from the GTS Message Switch Systems		
Relation to prototype			
Meteo PM42			
Requested			

functionality			
Acquire meteorological real-time data (such as observations) from the GTS			
Validation			
Implementation of the module providing an access to the real-time data coming from the GTS			
Assumptions			
The module will be integrated in the VMC Data Repository			

Name	VO Assessment and consolidation		
Req. Id	METEO-P3.6		
Application Activity	Meteo		
Prototype(s)	Meteo Prototype for PM42		
Date Created	2006-09-10	Priority	High
Created By	ECMWF	Technology component	Virtual Organisation
Status	Evaluation		
First Implementation Date	Phase III	SIMDAT module targeted	WP 12.3
Description			
During phase III the VO services implemented in phase II will be deployed at all the partners. The VO will ensure that the infrastructure provided is flexible enough to accommodate all the partners' requirements. A second iteration with all the partners involved will assess the suitability of the VO infrastructure to ensure interoperability and flexible integration.			
Relation to prototype			
Meteo PM42			
Requested functionality			
Interoperable VO infrastructure			
Validation			
Deployment at all the partners			
Assumptions			
Authentication and authorization services have been fully implemented.			

Name	Address cross-organisational issues.
Req. Id	METEO-P3.9
Application Activity	Meteo

Prototype(s)	Meteo Prototype for PM42		
Date Created	2006-09-10	Priority	High
Created By	ECMWF	Technology component	Virtual Organisation
Status	Evaluation		
First Implementation Date	Phase III	SIMDAT module targeted	WP 12.3
Description			
The VGISC might include a number of organizations wanting to maintain their own Certifications Authorities or Authentication mechanisms. Addressing issues such as key interchange to validate a number of CAs or support for different authentication mechanism will determine successful deployment and operation of the VO.			
Relation to prototype			
Meteo PM24			
Requested functionality			
Support for different CAs or authentication mechanism.			
Validation			
Authenticate a user at one domain with a given CA, and validate at a different domain.			
Assumptions			
Certificates Authorities must be fully established with well defined Certification Policies.			

Name	Monitoring Tools		
Req. Id	METEO-P3.9		
Application Activity	Meteo		
Prototype(s)	Meteo Prototype for PM42		
Date Created	2006-09-10	Priority	High
Created By	ECMWF	Technology component	VO
Status	Evaluation		
First Implementation Date	Phase III	SIMDAT module targeted	WP 12.3
Description			
Monitoring interfaces will be defined and implemented to control the activity of the system and ensure that system performs to the agreed QoS.			
Relation to prototype			
Meteo PM42			

Requested functionality			
VO activity monitoring.			
Validation			
System administrators will be able to monitor the synchronization, data delivery and VO services through the interfaces provided.			
Assumptions			
The infrastructure is fully deployed and configured.			

Name	Subscription service		
Req. Id	METEO-P3.12		
Application Activity	Meteo		
Prototype(s)	Meteo Prototype for PM42		
Date Created	2006-09-10	Priority	High
Created By	ECMWF	Technology component	Analysis Services
Status	Evaluation		
First Implementation Date	Phase III	SIMDAT module targeted	WP 12.3
Description	One of the missions of the VMC infrastructure is to deliver data to end-users, as soon as it becomes available. A service called subscription service allowing users to subscribe to incoming datasets will be implemented. The subscription service must be able to deliver to the end-users in quasi real-time meteorological observations but also large datasets such as forecast model outputs. The subscription service is able to make use of different delivery transports such as FTP, HTTP, etc		
Relation to prototype			
Meteo PM42			
Requested functionality			
Discover and subscribe to datasets in order to receive them as soon as they become available.			
Validation			
Delivery of datasets to subscribers within a minute			
Assumptions			
A robust delivery mechanism must be fully integrated and deployed.			

8 Conclusion

The meteorological application has undergone very active developments during the first six months of the project. The infrastructure has integrated its final requirements to enhance interoperability, providing interfaces in line with the web services specifications. Once more, the flexibility and “easy to integrate infrastructure” has been demonstrated through interactions with the widely adopted OpenNDAP data servers.

Ontology has been brought into the infrastructure to close the gap between the meteorological metadata standards (which follow a lengthy and formal evolution process) and a richer description of meteorological terms defined through ontologies and thesaurus that guide the non expert users through the navigation of more complex relationships.

The Virtual Organisation trust model has been implemented, providing a first version of the security, trust and management components. A first test-bed to demonstrate the VO has been set-up between several Nodes at ECMWF and NEC, where extensive testing has been carried out to ensure a stable release before the global deployment. The experience gained so far through the establishment of the Trust Relationships has made us realised that the infrastructure has gained complexity and needs the adequate tooling to be properly operated and managed (much of it due to the PKI). Extensive documentation and a careful deployment are required to ensure a smooth transition to the next version across the 11 sites that have already deployed the software. This deployment will be carried out in the following months of the project.

Still the final phase of SIMDAT will face challenging developments to fulfil the missing requirements of the VGISC such as the Subscription Service to data produced in the future. Once the Subscription Service ready, the infrastructure developed by the meteorology activity should be fully ready for production and start serving users worldwide.

Glossary

Term	Definition
AMS	American Meteorology Society
DR	Data Repository
GCMD	Global Change Master Directory
GISC	Global Information System Centre
GTS	Global Telecommunication System
MSS	Message Switch System
CN	Catalogue Node
OPENDAP	Open-source Project for a Network Data Access Protocol
SKOS	Simple Knowledge Organisation System
SWEET	Semantic Web for Earth and Environmental Terminology
UNIDATA	http://unidata.ucar.edu
VGISC	Virtual Global Information System Centre
VMC	Virtual Meteorological Centre
WMO	World Meteorological Organisation
STS	Security Token Service
PKI	Public key Infrastructure