Validation server Design Document

Deliverable title	Validation server Design Document
Deliverable number	D8.2
Revision	01
Status	final
Planned delivery date	30/06/2012
Date of issue	09/07/2012
Nature of deliverable	Report
Lead partner	S&T
Dissemination level	Public

This work has received research funding from the European Community's Seventh Framework Programme ([FP7/2007-2013]) under grant agreement n°284421.

DOCUMENT PROPERTIES

	FUNCTION	NAME	DATE	SIGNATURE
LEAD AUTHORS	Responsible persons at S&T for D8.2	L. Breebaart S. Niemeijer		
CONTRIBUTING AUTHORS				

Table of Contents

Applicable and reference documents

[MARS-UG]	ECMWF Technical Notes, *MARS User Guide*, January 2012
[NORS-DFD]	E. Mahieu, et al, *NORS Data format definitions*, NORS Deliverable D4.1, Revision 01, 2012-05-09.
[NORS-DOW]	DOW-284421 NORS - Part A, *NORS Description of Work*, 2011-08-04.
[NORS-URD]	B. Langerock, M. De Mazière, et al, *NORS Validation server User Requirements Document*, NORS Deliverable D8.1, Revision 02, 2012-05-14.
[NORS-MOM1]	N. Kalb and M. De Mazière, *NDACC Teleconference - Meeting Minutes,* 19 April 2012.
[NORS-MOM2]	B. Langerock and M. De Mazière, *NDACC Teleconference #2 - Meeting Minutes,* 28 June 2012.
[GEOMS]	C. Retscher, et al, *The Generic Earth Observation Metadata Standard (GEOMS)*, Version 1.0, 2011-03-21.

Acronyms and abbreviations

Acronym	**Explanation**
AMQP	Advanced Message Queuing Protocol
AVDC	Aura Validation Data Center
BIRA	Belgisch Instituut voor Ruimte-Aeronomie
COTS	Commercial Off-The-Shelve
DFD	Data Format Definitions
DMZ	DeMilitarized Zone
DOW	Description of Work
ECMWF	European Centre for Medium-Range Weather Forecasts
EOCDCIO	Earth Observation Correlative Data Centre Interoperability
FTP	File Transfer Protocol
GAS	GMES Atmosphere Service
GDF	GECA Data Format
GECA	Generic Environment for Calibration and Validation Activities
GEOMS	The Generic Earth Observation Metadata Standard
GMES	Global Monitoring for Environment and Security
GRIB(2)	GRIdded Binary (2)
HDF	Hierarchical Data Format
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
ISO	International Organization for Standardization
JSON	JavaScript Object Notation
LDAP	Lightweight Directory Access Protocol
MACC	Monitoring Atmospheric Composition & Climate
MARS	Meteorological Archival and Retrieval System
NDACC	Network for Detection of Atmospheric Composition Change
netCDF	Network Common Data Form
NFS	Network File System

NOAA	National Oceanic and Atmospheric Administration
NORS	Network Of Remote Sensing Ground-Based Observations
NRT	Near Real Time
NVS	NORS Validation Server
PDF	Portable Document Format
PNG	Portable Network Graphics
S&T	Science & Technology
SFTP	Secure FTP (FTP over SSH)
SMTP	Simple Mail Transfer Protocol
SQL	Structured Query Language
SSH	Secure Shell
UAD	NVS Admin Users
UALL	Unauthorized NVS users
UINT	Interactive NVS users
URD	User Requirements Document
URL	Uniform Resource Locator
UVIP	VIP NVS users
VM	Virtual Machine
WP	Work Package
WSGI	Web Server Gateway Interface
XML	eXtensible Markup Language

1. Introduction

1.1. Purpose and Scope

This document contains the design of the NORS Validation Server (NVS). It covers the deliverable D8.2 (*Validation server design document*) from work package WP8, as described in [NORS-DOW].

This document describes the software architecture of the NVS.

1.2. Overview

The primary objective of the NVS system is to generate in an operational and consistent way validation reports of GAS products from the MACC-II project, based on intercomparisons with independent NORS data products.

NVS operates in both an automated, data-driven mode where the incremental generation of default outputs for public consumption is triggered by the availability of new or updated products, as well as in an interactive, user-driven mode, where custom outputs are generated based on specific parameters, requests, or even input product data submitted by authorised users of the system.

The high-level user requirements for the NVS system are defined in [NORS-URD], the NORS Validation Server User Requirements document.

In this design document we define the architecture of the NVS. The document is structured as follows:

- Chapter 2, *System Architecture*, gives a high-level overview of the overall system architecture and interfaces, including the deployment architecture at BIRA.
- Chapter 3, *Functional Decomposition,* provides a detailed description of the functional components and interfaces that make up the NVS.
- Chapter 4, *Compliancy Matrix,* maps the User Requirements defined in [NORS-URD] to the component(s) in the design that take care of their implementation.

2. System Architecture

2.1. Context Diagram

Figure 1 depicts the context diagram for NVS, showing the system and its relationship to the external entities it will be interfacing with.

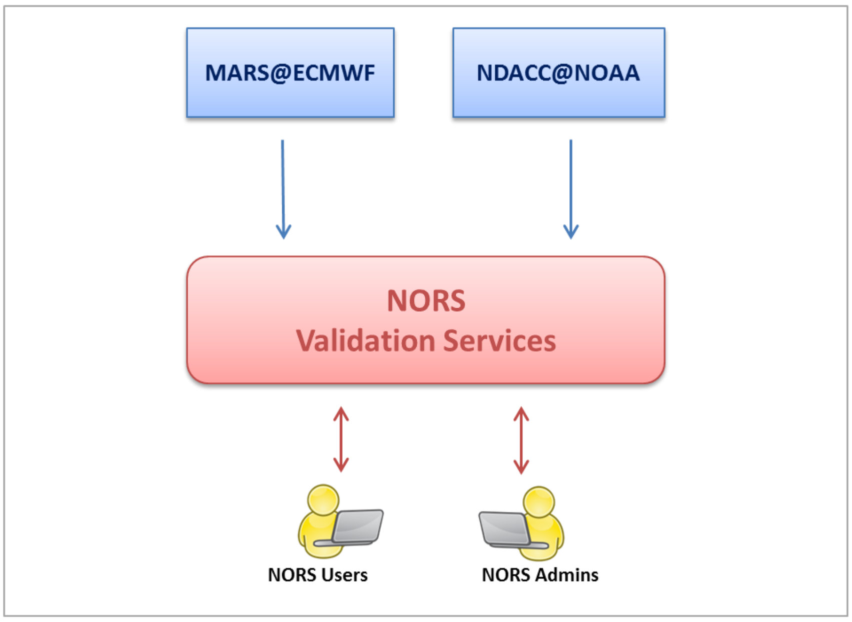

Figure 1. NVS Context diagram.

The external entities in question are:

- The Network for Detection of Atmospheric Composition Change (NDACC) archive, hosted at the National Oceanic and Atmospheric Administration (NOAA). This archive is the source of the NORS data products that are used by the NVS as its reference data inputs.

- The Meteorological Archival and Retrieval System (MARS), hosted at the European Centre for Medium-Range Weather Forecasts (ECMWF). This archive is the source of the MACC-II data products that are used by the NVS as its model data inputs.

- NORS Users. This category of users includes the three user roles identified in [NORS_URD]: UALL, UINT and UVIP. These are all users that provide inputs, and then obtain reports from NVS remotely over the Internet. Each role has different capabilities and permissions, but that is purely a function of the authorization level – conceptually they are all the same category of NORS/NVS end users.

- NORS Admins. This category of users (identified as UAD in [NORS-URD]) is permitted to make configuration changes to the NVS, and will require actual log-in access to the system NVS runs on in order to do so.

2.2. External Interfaces

The external interfaces of the NVS as given in this figure are global aggregates that may comprise a number of different protocols and data streams. They will be explained in greater detail further on in this document; a summary at the overall architecture level is as follows:

2.2.1. NDACC - NVS

This interface is used by NVS to retrieve both NORS remote sensing data products and metadata files describing those products from the NDACC archive. The data products will all be in GEOMS 1.0-compliant HDF format (as defined in [GEOMS]), where appropriate tailored by the NORS GEOMS templates (as defined in [NORS-DFD]). The metadata files will be XML files adhering to the EOCDCIO XSD data schema v0.8. The arrival of new metadata files (as upload by the NDACC admins) will serve as a 'push' mechanism to trigger the retrieval of the corresponding product files. All file transfers to NDACC take place using the (anonymous) FTP protocol.

As BIRA will be maintaining a local mirror archive of all the NDACC product files, the decision has been made for NVS (in the deployment scenario) to interface with this archive instead of directly with NDACC. The BIRA mirror is currently refreshed every week, but this frequency will be increased to daily, as it is expected that at least some NORS data will be updated that often (see [NORS-MOM1]), and it is desirable to have the validation server process new files without unnecessary delays. Similarly, the BIRA mirror archival scripts will be updated to deal with the proposed NDACC archive update, in which there will be separate top level directories for NRT and for consolidated uploads (see [NORS-MOM2]).

2.2.2. MARS - NVS

This interface is used by NVS to retrieve MACC-II GAS data products from the MARS archive. The data products will all be in GRIB2 format. MARS exposes two interfaces for retrieving its contents. The Web-MARS interface is intended for interactive browsing and querying, and is therefore not suitable for use by the NVS system. NVS will instead use the MARS Unix command-line interface (as defined in [MARS-UG]) for retrieving data products in batch mode. The ECMWF *ectrans* mechanism is subsequently used to transfer the files to the physical NVS system.

As BIRA already maintains a local archive of relevant MARS product files, the decision has been made for NVS (in the deployment scenario) to interface with this archive instead of directly with MARS.

2.2.3. NORS Users - NVS

This interface is used by NORS users to browse through default validation outputs, to view generated reports, and to request and download custom validations from NVS. The primary interaction is through a web browser running on the user's workstation that communicates with NVS over the Internet using the HTTP 1.1 protocol (i.e. web pages). Sufficiently authorized users (at UINT level or higher) will be able to request custom reports that may take a while to generate because they involve an invocation of the validation processing (rather than browsing through information already available on the system). In these cases, the processing will be done asynchronously and an SMTP e-mail interface will be used to eventually signal to the user that the processing is finished and that the resulting report can be viewed or downloaded via the browser. Users at UVIP level and higher are allowed to upload their own product files as inputs for the report generation process. For performance and robustness reasons this will not be done through HTTP, but through a separate interface channel using the SFTP protocol.

2.2.4. NORS Admins - NVS

This interface is used by NORS admins to configure and interact with the NVS. This mainly takes the form of changing various 'hardcoded' parameters associated with the default report-generation process (e.g. time intervals, output colours, selection of MACC-II models to use, etc.). Other possibilities are user account management, viewing log files, observing system performance, or adding support for new NORS products or MACC-II models (which may involve manually adding code to the validation tool chain, e.g. for handling effective airmass calculations).

All Admin configuration functionality will be exposed to UAD users through the NVS web interface. It is expected that all the explicit requirements mentioned [NORS-URD] will be able to be handled this way.

A possible exception to this is the aforementioned case where administrative actions require access to the physical NVS system, for example to actually change the configuration of the web application, or to add/update scripts. Since in the deployment scenario access to the backend servers will be only available to BIRA system administrators, external UAD users will not be able to log on to such systems from the wider Internet.

The current design for NVS will ensure that there is no need for physical access to the system beyond what is available in the BIRA deployment setup. Code updates can be handled either by an update of the NVS tool chain (or other components) by the NVS developers, to be installed in cooperation the BIRA sysadmins, or by a system through which UAD users can upload new scripts or snippets as files to the NVS systems, from where they can be (semi)-automatically integrated.

2.3. Data storage

The NVS system will store several different types of data. In the following, when we mention the word 'database', we talk about the conceptual: a task database, a user database, etc. In the actual architecture, each of these conceptual databases is likely to be stored as a set of tables in one physical 'NVS' database. However, with a view towards scalability and flexibility it should be understood that a distributed implementation (i.e. using different physical databases) is also possible.

The data stored by the NVS system includes:

2.3.1. Product files

- For the default validation chain, new product files (both NORS and MACC-II) are retrieved and cached on the system as they become available. These files are governed by a UAD-configurable expiration policy depending e.g. on system disk space and other factors. As MACC-II data is released faster than the NORS data, the default retention time for model product files will be large enough (> 1 month) to ensure that the files will actually have been used before expiration.

In the BIRA deployment situation, with the NVS core application running in the Internal zone with direct access to the BIRA file archive, a shortcut to this design may result in no caching being used, but files instead being directly retrieved from the file system.

- For the custom validation chain, a user may request files that extend beyond the caching period. In this case, the Product Handling component will ensure that the necessary files can be transparently (and asynchronously) requested from the source archive.

  Product files used for custom validations can be deleted as soon as the validation has taken place.

- All retrieved product file metadata (possibly augmented with derived metadata properties not directly represented at the source) is stored permanently in a metadata catalogue.

2.3.2. Validation Outputs

- The NORS default validation chain generates a set of outputs files in various formats, as described in Chapter 8 of [NORS-URD]: plots, text files, GECA Data Format (GDF) data files. These outputs are stored permanently on the system as a validation is run. Some of these outputs will be exposed for browsing and viewing by the NORS user (e.g. the plots); others (e.g. the data files), will be cached to be re-used as inputs to subsequent invocations of the default validation toolchain.

- Metadata describing the validation outputs is stored in an outputs metadata catalogue.

- Outputs generated by user-requested custom validations are only temporarily stored (subject to an expiration policy) and are not integrated into the output browsing mechanism.

2.3.3. Users

- Registered NVS user accounts are stored in a user database.

2.3.4. Tasks

- Information about currently running and scheduled NVS tasks (e.g. custom validation runs requested by users, or periodic runs of the default validation chain) is stored in a task database.

2.4. Process architecture

Figure 2 depicts the process architecture of the NVS system:

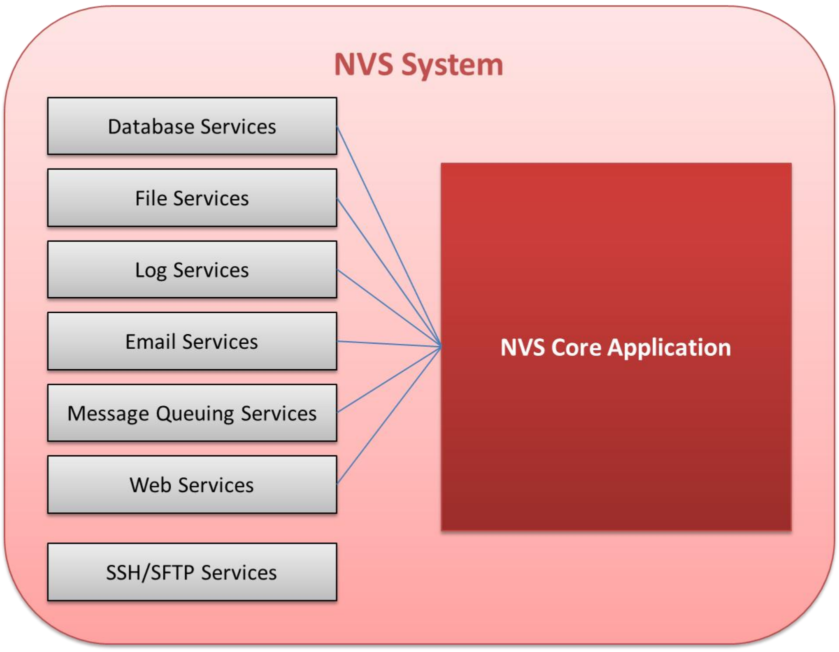

Figure 2. *The NVS Deployment architecture.*

Each grey box in the figure represents a high-level heavy-weight process component of the NVS system. These components will be loosely coupled so that in the final deployment it is possible to run individual components on different hardware, and change this configuration over time. This can be done for performance reasons, or to make it possible for e.g. the web server component to be decoupled from the core report-generating processing backend. This also enables deployment strategies where e.g. the web server resides in a demilitarised zone (DMZ), but the product file server (with access to the product files) on a local Intranet. Another example would be, for example, having the SMTP server for sending notification emails reside on a different system in the network. In general, the processes in this architecture are connected via IP network connections, using various different protocols. It is e.g. possible for a file system to be mounted transparently over NFS or Samba, and be accessed by the core app as a local file system, rather than as an Internet networked service.

The components in the architecture are:

2.4.1. NVS Core Application

This is the central web application that controls the user-interaction, retrieval, validation and output-generating functionalities of the NVS system. The core application is not a monolithic entity, but itself a collection of interacting and interfacing components (see below). The Core application is the only component in the architecture that will be actually be developed by the NORS project. All other components are Open Source or COTS components that will be integrated with the Core application into an overall NVS system.

The NVS Core Application will be developed in ISO Standard C (for the validation toolchain components) and Python 2.x (for everything else)[1], using the Django web-application framework to provide the user interface.

[1] At the moment, Python 3, a major, backwards-incompatible update to the Python language, is not yet wide-spread enough to be used for NORS. Too many libraries (including Django) still require Python 2.x, which is still actively being

2.4.2. Database Services

This is an interface to one or more SQL databases used for storing output and product metadata, the conceptual user and task databases, and other internal storage needs for the NVS.

The Django framework can interface with all major SQL servers available (such as MySQL or PostGres). There is therefore no specific database server or protocol mandated by this NVS design. For the reference system at S&T, a local installation of MySQL will be used. For the deployment at BIRA, an interface to the DBServer in the Protected Zone (using MySQL) will be used. It is envisioned that there will two physical databases used for the NVS data storage: one residing in the internal zone for the core (meta)-data required by NVS, one on the website for the data dealing more directly with the user interface. This decoupling will make it easier to separate updates to the web interface of NVS from updates to the core internal processing.

Both databases would primarily be managed by a Django application.

2.4.3. File Services

This is an interface to a file-based directory system used for storing actual reports and product files. As mentioned above, the file server will be transparent to NVS: the file system will either be local to the physical NVS machine, or else mounted from a remote server as a local file system so that it still appears local to the core application. For the deployment at BIRA, the calculating parts of NVS will run on the Internal Zone and have the aforementioned local access. The Web server part of NVS running in the Protected Zone will be able to access file services (and make these files available to the user) through the BIRA Requestor proxy mechanisms.

2.4.4. Log Services

This server stores tracing and debugging messages generated by the various components in the system. The log server writes its messages to a log file (or log file database) that can be reviewed by UAD users.

NVS will not connect to a dedicated log server, but will directly store to textual log files on the (local) file system.

Log files will be stored for at least 60 days.

2.4.5. Email Services

This is an SMTP-compliant server that is used to send notification messages to authorised NORS users who have requested a time-consuming task that will be executed asynchronously. It can reside anywhere in the net.

NVS will generate RFC-5322 compliant email messages.

developed and supported by the Python foundation, in parallel to the Python 3 effort. There are no deficiencies in Python 2.x that would make it unsuitable for developing NVS.

2.4.6. Message Queuing Services

This is an interface to an AMQP-compliant message queuing system used to facilitate asynchronous task execution by the core NVS application back end. This server is used, for example, when time-consuming custom report generations are requested by the users, but will also be the mechanism by which e.g. the MARS/NDACC polling and retrieval tasks are handled.

NVS will interface to the AMQP-compliant *RabbitMQ* system.

2.4.7. Web Services

An HTTP 1.1-compliant WSGI-enabled server used to host the web GUI of the Core application.

NVS will use the Apache web server.

2.4.8. SSH/SFTP Services

The system service that allows UVIP users (via SFTP) to upload input files from the wider Internet to the DMZ.

In the actual BIRA deployment scenario, this is an interface that has to be particularly carefully integrated in the wider architecture (because of security concerns), which is a design issue that is still being discussed with all the relevant stakeholders.

2.5. Deployment Architecture

The operational NVS will be deployed on a Linux-based platform running at BIRA. A reference version of the server will be hosted at S&T during the project's development time.
The network topology and file system architecture at BIRA are depicted in Figure 3 and Figure 4.

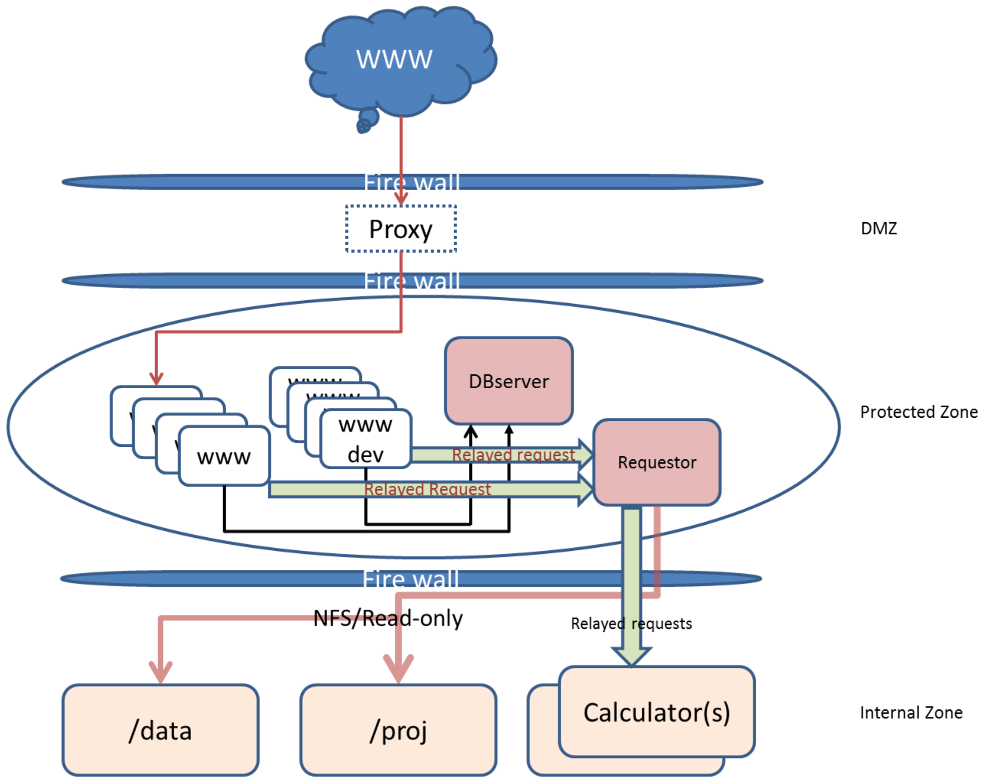

Figure 3. *Network Topology at BIRA*.

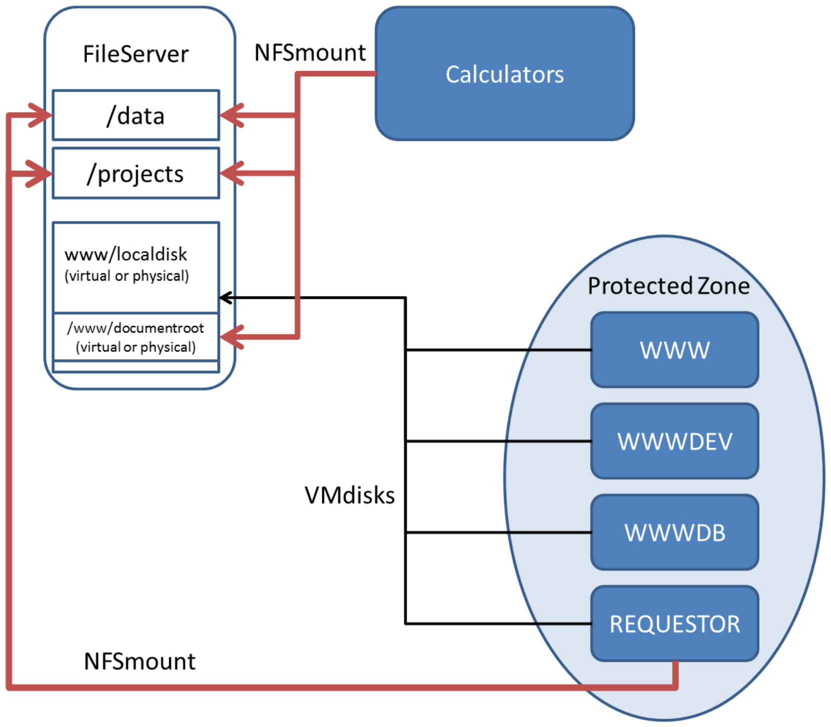

Figure 4. *File system architecture at BIRA.*

For the deployment architecture, we will map the NVS components to the BIRA topology, such that only the web application UI frontend and the Database server reside in the protected zone. All other services run in the Internal Zone, where they can be reached via the REQUESTOR interface (e.g. to deliver file artefacts for viewing purposes).

The DMZ zone also contains the FTP service component that will be used by UVIP users for data upload.

3. Functional Decomposition

The decomposition of the NVS Core application into functional components is as follows:

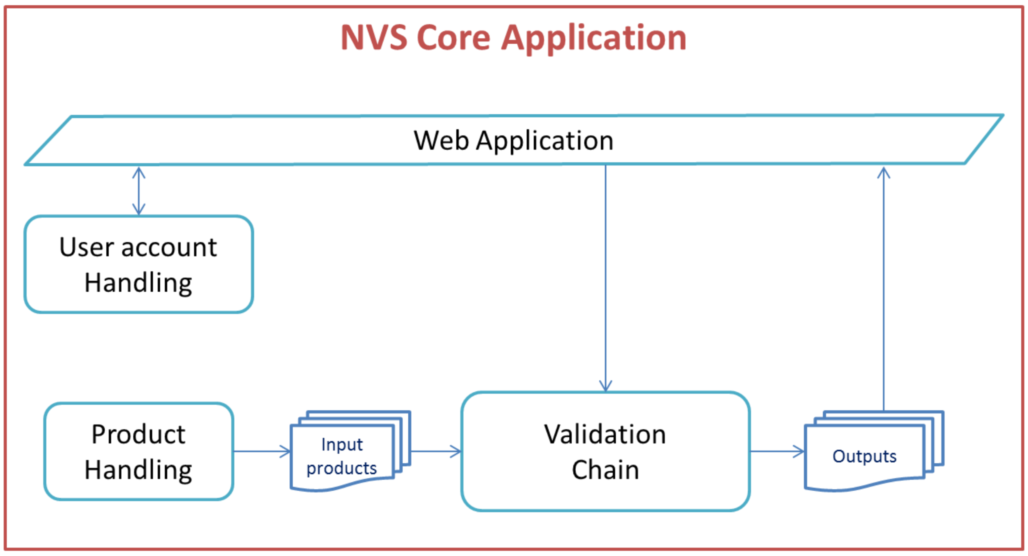

Figure 5. *NVS Core Application Design.*

The NVS core application will be implemented as a Web application written in the Python language using the Django framework. Within the NVS core application we distinguish between three major top level functional components. For clarity's sake in Figure 5 only the most important data-flow, that of the actual validation and report generation, is shown. The detailed design for each of the components is given below.

3.1. User account handling

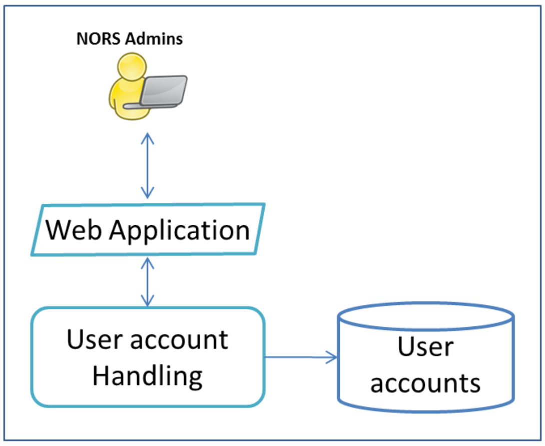

Figure 6. *User account handling.*

This component, using the standard Django 'admin' component, manages the NVS user database. It defines a User model that contains the following fields:

- Username
- Password
- E-mail address
- Full Name
- Affiliation
- Maximum-allowed Role/Group (One of UINT, UVIP, UADM)

As per the User Requirements in NORS-URD, it is envisioned that the Role/Group values form an hierarchical set, where each role includes the privileges of the roles 'beneath' it. That is, a UINT user will have access to all the NVS functionality that an anonymous UALL user has, plus some extra functionality. Similarly, a UVIP user can do anything a UINT user can, and a UAD user has access to *all* functionality in the system.

The User Account handling component provides the following functions:

- A web interface for logging into NVS. The user can select a specific kind of role at log-in (or during their session), up to and including the maximum allowed level for that user. For example, a user who has UVIP as their maximum-allowed role, can log in as or switch to a UINT user (and e.g. be presented with a less complex user interface as a result.)

- Authorisation based on URL-matching, i.e. different web page views throughout the NVS Core Application will query this component to determine if they are available to the current user attempting to view them.

- A web interface for managing NVS users (adding, deleting, modifying). No change tracking – only the most recent user record persists in the database. Obviously, this interface is itself only available to user logged in as UAD. This interface also supports sending new users an email with their account information.

- A user account page where users can request an account, and manage their own account information once they have received one from the NORS admins. If a new user account is requested, an email request is generated to a designated NORS user registration email address. Once a NORS admin has created the account, the user is notified in email (see previous point) and can start using the system.

Note that the authentication of an NVS user is completely handled by the NVS Core app itself via this component. It is therefore not necessary to implement authentication through an external component such as e.g. the Web Server or LDAP.

3.2. Product Handling

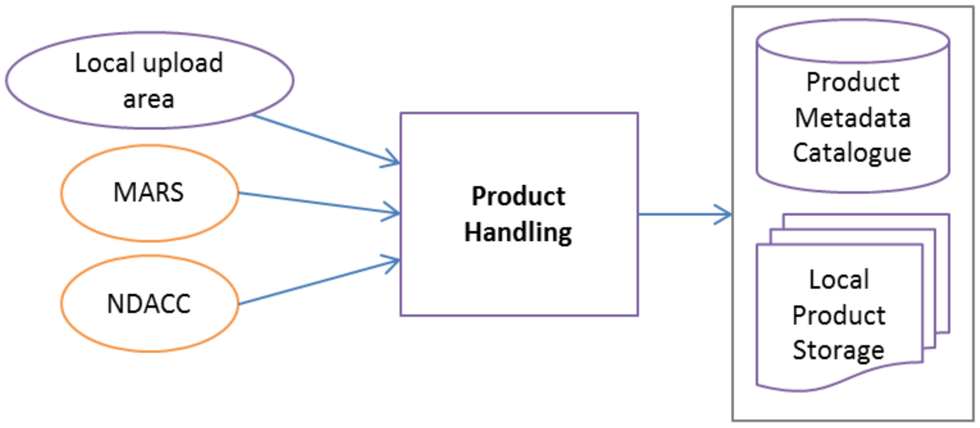

Figure 7. *Product Handling.*

This component shields the other components in NVS from having to know the implementation details of where and how the model and reference data products are retrieved. Multiple different product handling mechanisms can be configured in this component, so that different ways of retrieving data for NVS can be supported.

For the default NORS validation chain, Product handling is a data-driven component that is triggered by the availability of new product

The Product Handling component:

- Maintains and governs the *Local Product Storage*: the local file system-based caching of product files (both reference and model data). An expiration mechanism is built-in and configurable for UAD users via a configuration file.

- Maintains and governs the *Product Metadata Catalogue*, a local database containing the product metadata. The Metadata model is a superset/aggregation of existing product metadata (e.g. the EOCDCIO metadata XML made available for NORS reference products), augmented with additional metadata fields derived by direction inspection from the product files themselves.

- Provides a programmatic interface for querying the metadata catalogue in order to obtain references to the resulting product data.

- Is able to interface with the following three sources of product files:

  - The MARS remote archive at EMCWF (for model data product files, via *mars-command & ectrans*)

  - The NDACC remote archive at NOAA (for reference data product files, via anonymous FTP)

  - The local UVIP user upload area (for both reference and model data product files, via monitoring the file system)

Note that, as per the deployment architecture mentioned earlier, the Product Handler may be interfacing with a local BIRA proxy service for the ECMWF and/or NDACC archives.

- Implements watchdog functionality that will autonomously, continuously monitor the three product sources for relevant updates, such as the addition of new files, or updates to existing files. The exact means by which these events are determined is modular and can differ for the different sources.

- Notifies the Validation Chain of changes to relevant products. This way, the arrival of a new product file on the system can trigger the start of a Validation Chain default processing sequence.

3.3. Validation

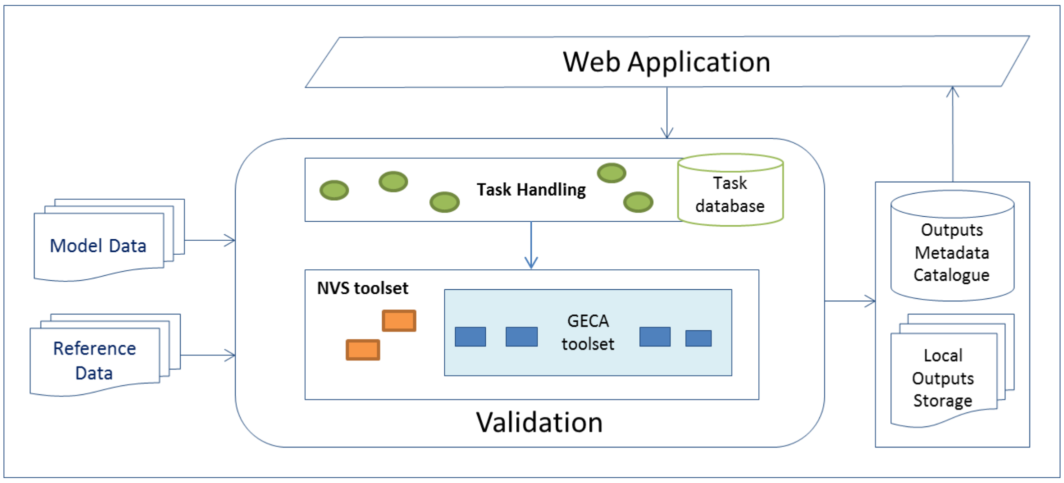

Figure 8. *Validation toolchain*.

The Validation component implements the central NVS data processing. There are two main processing flows in NVS:

- The *default validation* process is a data-driven processing flow that corresponds to the default use case, as described in [NORS-URD].

  As the result of default validation, an outputs database is incrementally built up over time on the local filesystem. 'Outputs' are here defined as all artefacts resulting from the validation process. These do no only include the outputs that are intended for viewing by users (e.g. the plots and statistics files), but also e.g. the processed files in GDF format that can be used to generate other outputs.

  As soon as new product files become available, a task is scheduled for handling the further processing.

- A *custom validation* process is a request-driven processing flow that corresponds to the interactive and VIP use cases, as described in [NORS-URD].

As the result of a custom validation, a workspace with output results is created from scratch, and made available to the user for download.

As soon as a custom validation is requested, a task is scheduled for handling the further processing.

3.3.1. The GECA toolset

The GECA Toolset is a toolkit that has been developed in the context of the Generic Cal/Val Analysis Environment (GECA) project for ingesting, processing and inter-comparing satellite data against correlative data, which can be either in-situ data or other satellite data.

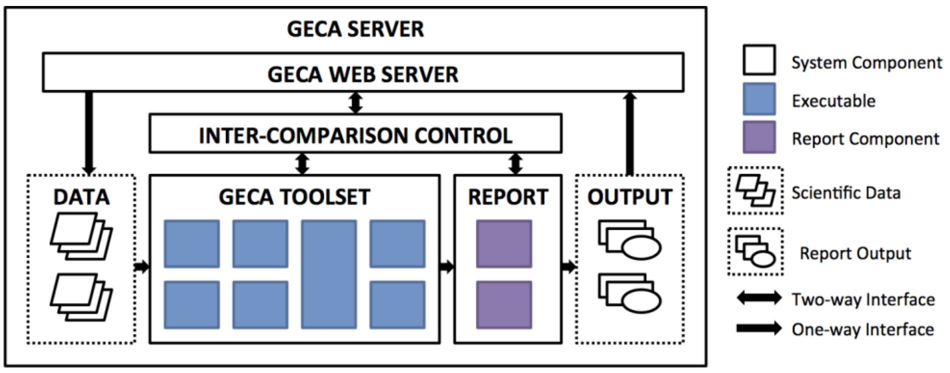

Figure 9. *The GECA Toolset architecture at the GECA server.*

The GECA toolkit is composed of a set of command line tools and a library of analysis functions that together provide the following features:

- Bring data into same format, same resolution (temporal, spatial, smoothing, etc.), and define ‘pairs’
- Tradeoff in algorithms between first-order and best-practice
- Uses an internal file format based on GEOMS (netCDF)
- Actual comparison of quantities is performed by a reporting component

The main goal of the GECA Toolset is to assist in the inter-comparison of data sets. By appropriately chaining calls to the GECA Toolset command line tools one can pre-process satellite and correlative data such that the two datasets that need to be compared end up having the same temporal/spatial grid, same data format/structure, and same physical unit. At the end of the toolchain you will have a set of data files that can be directly compared by user programs and tools, or that can be further processed using the GECA Toolset Reporting component.

A typical part of intercomparison toolchain flow for GECA is illustrated in Figure 10. After two datasets A and B have been ingested and collocated at the measurement level, they are ready for further conversion, filtering, resampling, and processing.

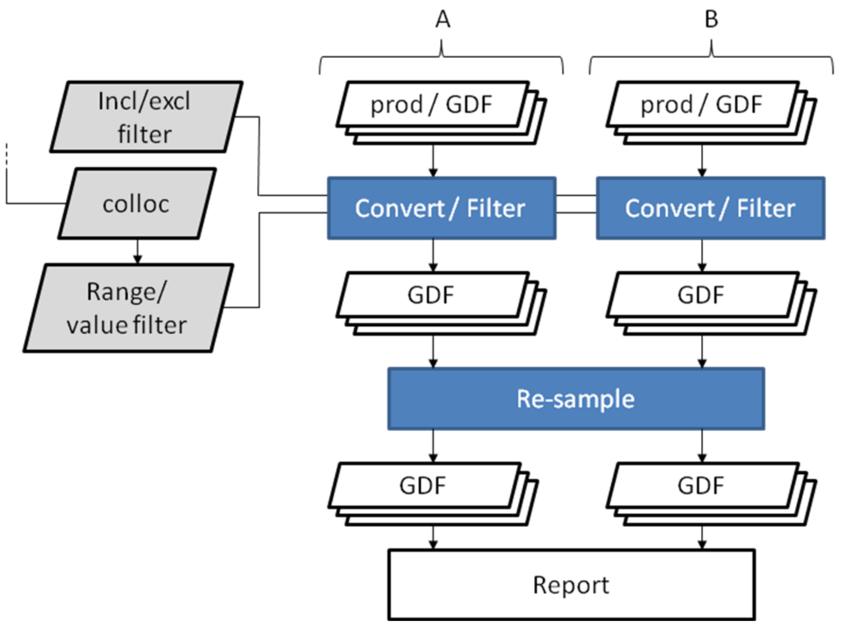

Figure 10. *GECA Intercomparison.*

The intermediate files the GECA toolset generates are in the GDF (GECA Data Format) format, which is GEOMS-based.

Although the GECA use cases are not the same as the NORS use cases, there are clear overlaps and similarities between the two tool chains, hence the decision to base the NVS toolchain components on the GECA tool chain, where necessary extending the GECA tools, or adding new functionalities and tools.

The available tools in the GECA toolkit that will be relevant to the NVS system are as follows:

3.3.1.1. gtfilter

The *gtfilter* program is a multi-purpose tool for filtering data and converting quantities. It can be used to:

- Convert the unit of a variable
- Add a derived variable from variable(s) in the product
- Filter out certain values of a variable
- Filter out invalid values for a variable
- Apply a point distance filter
- Apply an area filter
- Apply the collocation filter
- Exclude certain variables

Gtfilter both operates on and produces input files in the GDF format.

3.3.1.2. gtconvert

The *gtconvert* program is a tool for converting external format product files into the intermediate GDF format.
For performance reasons, *gtconvert* also includes the same functionality as *gtfilter,* so that file ingestion/conversion and filtering can be combined.

The input product formats that *gtconvert* currently supports are ASAR, ATSR, RA, RA-2, GOME, GOMOS, MERIS, MIPAS, MODIS, and SCIAMACHY.
Gtconvert produces GDF files.

3.3.1.3. gtcollocate

The *gtcollocate* program is a tool for deriving a list of measurements that match in time, latitude and longitude for two sets of GDF-compliant files.
Collocation of two sets of files involves the following steps:

1. *gtcollocate matchup* is used to derive the collocation result file.
2. *gtcollocate resample* is optionally used to further process the collocation result file.
3. When needed, *gtcollocate update* can be used to refresh the collocation result file.

Gtfilter operates on input files in the GDF format. It produces an ASCII collocation results (which can also be an input to a subsequent *gtcollocate* call, e.g. to perform an update).

3.3.1.4. gtprofile

The *gtprofile* program is a tool that is used to manipulate two sets of vertical profile data.
It can be used to:

- Get pairs of vertical profiles in two datasets on the same vertical grid.
- Set vertical boundaries and convert units.
- Perform smoothing with Averaging Kernel Matrices.
- Convert Vertical Profiles to (Partial) Columns

Gtprofile both operates on and produces input files in the GDF format.

3.3.1.5. gtreport

After the GECA toolchain has run and produced two sets of processed / collocated / smoothed / etc. GDF files (plus optional collocation text files), the *gtreport* tool is used to produce a user-consumable report that describes a single intercomparison chain.

Gtreport operates on datasets of input files in the GDF format, ASCII collocation files, and a report configuration file in the JSON format. As output, it generates a directory (or zip file) containing a single HTML report that aggregates a variety of plots and statistics (each of which is also available as a separate file). A PDF version of the report is also generated from the HTML version. The contents of the report are defined by the report configuration file (which can be handwritten or generated by *gtcontrol* for specific types of intercomparison).

Gtreport offers the following generic output components for use in its reports:

- Images (in PNG format)
- HTML Text
- Generic Tables
- Statistics Tables (featuring the columns *count, min, median, max,* and *std*).
- Dataset Information
- Generic Plots
- XY Series Plots
- Location Plots (map-based)
- Histograms
- Confusion matrices
- Collocation statistics

These component are also available to other programs through a Python library

3.3.1.6. gtcontrol

In the GECA context, *gtcontrol* is the tool that manages the entire intercomparison chain. Based on a single intercomparison definition file, it creates a dedicated area for all the intermediate and output files, and it invokes all the GECA tool components in sequence on the specified inputs.

3.3.1.7. Other tools

In addition to the tools mentioned above, the GECA toolkit contains a number of other tools, such as support utilities (e.g. for checking the validity or dumping the contents of a GDF file), or tools for manipulations irrelevant to the NORS case (such as 2D-regridding).

It should also be noted that all the common functionality in the GECA tools is concentrated in the *libgeca* and *libcoda* libraries, suitable for re-use by new tools.

3.3.2. The NVS Toolset

The approach to be followed in NORS is to re-use the components of the GECA toolset as much as possible. However, the GECA functionality itself is not always going to be sufficient to cover all the required functionality for NORS. Therefore, the GECA tools will be expanded and extended where necessary (in a generic fashion, so that the result is actually an iteration of the GECA toolset, not a NORS-specific fork). It is also possible that some entirely new tools will need to be created, but that is a design decision we postpone until there is full clarity within the project on all the algorithms required for the NORS intercomparisons. As of writing, the current design of the NVS toolset is expected to be:

3.3.2.1. gtfilter extension

Can be re-used as-is. No extensions are likely to be needed.

3.3.2.2. gtconvert extension

The *gtconvert* tool will be extended with the capability to:

- convert MACC-II GRIB2 data files to GDF.
- convert NORS GEOMS data files to GDF format.

3.3.2.3. gtcollocate

The *gtcollocate* tool will be extended to incorporate the collocation algorithms required for NORS validation as defined in Section 7.2 of [NORS-URD].

3.3.2.4. gtprofile

The *gtprofile* tool will be extended to incorporate the vertical regridding and smoothing algorithms as defined in Section 7.3 of [NORS-URD].

3.3.2.5. gtreport

The *gtreport* tool will be extended to incorporate the generation of the outputs as defined in Chapter 8 of [NORS-URD].
In particular, the underlying 'geca.report' library will be extended with support for a number of new types of visualisation components, specifically the following:

- Additional statistical quantities and scoring values, as defined in (S1)-(S7) of [NORS-URD]
- Taylor diagrams
- Mosaic plots
- PDF output for image formats.

In addition, *gtreport* will be extended to include an invocation mode

3.3.2.6. gtcontrol

In NORS, the *gtcontrol* utility will not be reused, because the data-driven, incremental approach to output generation for the default validation process is fundamentally different from the strictly sequential-from-scratch approach followed in GECA (and implemented by *gtcontrol*).

For the custom validation process, as well as for large-scale re-generation processes that may occur when e.g. an entire MACC model is updated, it is still necessary to perform a validation chain from start to finish, but this will be implemented for NVS in terms of the finer-grained task management functionality incorporated into the system.

It should be noted that there is still an actively on-going discussion in the NORS infrared working group as to the exact nature design of the algorithms to be used for profile construction and effective airmass location, etc. Once these discussions lead to a consensus on the desired algorithms and validation sub-steps, the NVS tool chain will be extended to incorporate those requirements. It is not envisioned, however, that this would require a radical redesign of the current tool-based approach.

3.3.3. Task Management

Task Management is the component within NVS responsible for the maintenance and scheduling of the processing tasks being executed by the system.
There is a need for both periodic tasks, and tasks that are triggered by certain events.

Although it offers more functionality than is needed for NVS, we have chosen to use a full message brokering system rather than just a lightweight messaging library such as *0mq*. The task management subarchitecture looks as follows:

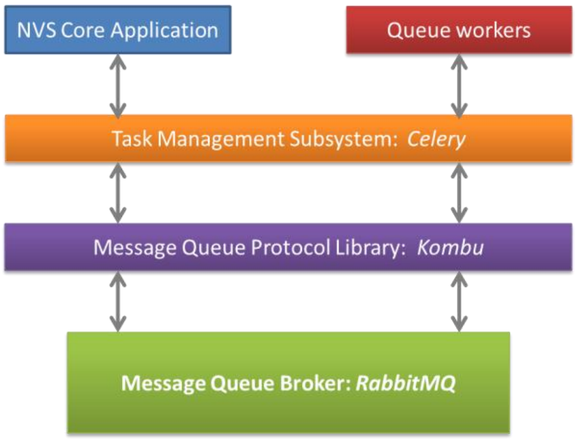
NVS Core Application
Queue workers
Task Management Subsystem: *Celery*
Message Queue Protocol Library: *Kombu*
Message Queue Broker: *RabbitMQ*

Figure 11. *Task management architecture.*

RabbitMQ will be installed on the NVS system as an external service component. NVS itself will interface with *RabbitMQ* through the *Celery/Kombu* Python bindings.

3.3.3.1. Default validation

Default validation processing runs use task management based on triggers from the product handling component: as soon as a new product arrives (either MACC or NORS), a task is started which will:

- Ascertain if any new measurements are available for extraction
- Ascertain if all the relevant data from the corresponding stream is already available
- Start the processing chain with the correct inputs.
- Store the outputs (file artefacts on the local file system, metadata to the Database server).

3.3.3.2. Custom validation

Custom Validation tool chain runs will be scheduled to run asynchronously (via the message queuing external interface) so that the NORS user's session with the NVS can continue uninterrupted. In order to facilitate this, a Task database will be maintained that keeps track of the different tasks being scheduled or running. UAD users will be able to monitor the current tasks, and be given a basic interface to control them. This interface will allow the following functionality:

- Cancelling a task
- Suspending / restarting a task
- Rescheduling a task
- Changing the frequency of a periodic task

If there is an UINT or UVIP user who instigated a custom report, they will also be use a similar web interface to control their own tasks (i.e. the ones they created). Once a task is finished, Task Management will notify the user (through the email external interface) that the report is available and direct them to a web page with relevant information to view or download the report from.

If an admin user has made a change that affects a user's scheduled task, this will also be automatically communicated to the user via email.

In addition, an admin user will have the capability to view and control system-defined tasks (such as e.g. periodic expiration or retrieval jobs) through the task manager web interface.

3.3.3.3. Load balancing

The task management system will ensure that the user-scheduled validation tasks do not overwhelm the system or affect performance negatively. This is implemented through configuration of the message queuing system in combination with higher level congestion control mechanisms: once a task (data-driven or user-requested) *can* theoretically be started, it will be placed in the queue first, and not actually executed until all conditions (such as e.g. acceptable the server load, or the required minimum number of parallel tasks) are fulfilled. The message broker will also provide a means to guard against starvation, and make sure that it is not possible for tasks to be continually excluded by newer tasks appearing in the system.

3.3.3.4. Error handling

When a task execution fails, the error is always logged to the system log, and in the case of a custom validation tasks a detailed notification will be sent to the originating user. In the case of an internal NVS failure (or anything else not directly attributable to a user error), the admin users will be notified as well.

3.4. Web Application

The Web Application makes the NVS validation results available to the NORS users.
It guarantees the consistency and availability of the default outputs as identified in [NORS-URD], and supports the handling of the interactive use cases also described in [NORS-URD].

This component provides to NORS users the main NVS web interface for browsing and querying the outputs database. This interface is used to collect from the user the so-called validation parameters that will configure the validation tasks. For UALL users (in the default use case), the validation parameters are simply selections from fixed enumerations of values, that allow a view on existing outputs database.

For UINT and higher, the report parameters can be new values that may require execution of a custom validation task, to be relayed to the NVS backend.

4. Compliancy Matrix

The following table describes the correspondence between the user requirements defined in [NORS-URD] and the component responsible for implementing that requirement, as described in this document:

ID	Description	Covered by
UR1	Authentication of the users is done by username/password. The users list is maintained by admin users UAD. Users that must provide a valid email (for the delivery of validation data). Users may change their password (the username is the users email address).	3.1 User Account Handling
UR2	The global attribute DATA_QUALITY in the NORS data file should mention that the data can be used for validation purposes. If this is not the case, the NORS server should ignore the NORS file.	3.2 Product Handling
UR3	The validation server should be designed in such a way new NORS products can be added to the list. The freedom for the admin user UAD to add NORS products is limited to the affiliation and location field: e.g. FTIR.O3_*_*.	2.2.4 NORS Admins - NVS
UR4	The admin user UAD should have a flexible interface to the MARS archive for the MACC-II models, in particular the experimental models.	3.2 Product Handling
UR5	All MACC-II products in the corresponding models in the above table should be validated	3.2 Product Handling
UR6	For each NORS product, except the NO_2 stratospheric profiles and total columns, a characteristic *n*-hours time interval is defined which is smaller than or equal to the *m*-hours time interval between MACC-II outputs (*m* is typically equal to 6).	3.3.2 The NVS Toolset
UR7	UAD should be able to change *n* for each NORS product, and depending on the frequency of the MACC-II model output (*m*).	2.2.4 NORS Admins - NVS
UR8	A NORS measurement should only be compared to MACC-II model data within this time interval.	3.3.2 The NVS Toolset
UR9	No further filtering for quality of the NORS data is required.	N/A – This is not a requirement.
UR10	For the spatial co-location, the co-location target associated with each NORS measurement (data point) should be the location of the effective airmass associated with that measurement.	3.3.2 The NVS Toolset
UR11	The design of the server S/W should be such that it is easy for a S/W engineer to implement or update for a given NORS product a code for the calculation of an effective airmass that differs from the location of the instrument, once this code is provided by the NORS consortium. The validation server should then use this calculation for co-locating the MACC-II output.	3.3.2 The NVS Toolset
UR12	The server allows an easy interface for UAD to change the default units for the different products and height coordinate in the default output.	2.2.4 NORS Admins - NVS
UR13	If derived products (like total column of total optical depth) are not available in the NORS data or MACC-II data, then the server is responsible for calculating these derived quantities if profile information is available from the NORS data or MACC-II data, respectively.	3.3.2 The NVS Toolset

ID	Description	Covered by
UR14	The NORS validation server should be designed such that it is an easy task for a S/W engineer to update the system when new instruments/products are added to the NORS database. For example, when FTIR stratospheric NO_2 data become available, a procedure similar to the one described above for the DOAS stratospheric NO_2 data will be applied.	2.2.4 NORS Admins - NVS
UR15	UAD must be able to set/change the partial column boundaries to be used in the default use case, for each NORS product (Pj); the settings will apply to all instances of (Pj).	2.2.4 NORS Admins - NVS
UR16	UAD must be able to set/change the vertical grid for the comparison of aerosol extinction profiles	2.2.4 NORS Admins - NVS
UR17	UAD must be able to set/change the common vertical grid for enabling more extensive statistical evaluations	2.2.4 NORS Admins - NVS
UR18	The MACC-II VAL contains all statistical quantities that should be implemented.	3.3. Validation 3.3.2.5. gtreport extensions
UR19	In addition to the statistical quantities, the system creates plots. All plots allow colours. Apart from plotting w.r.t measurement time, the server also allows plots for which a finest time unit is given (e.g., seasons DJF, MAM, JJA, SON within a validation plot over a total time interval of one year). The data plotted is then the average over the finest unit (i.e. the average of all biases for all measurements in a season).	3.3. Validation 3.3.2.5. gtreport extensions
UR20	All plots should contain - a clear indication of the physical quantity that is plotted with its unit, - the identification of the NORS products (species, instrument, location, data provider) and the MACC-II models, including the data file version for both, - a corresponding legend in the case of multiple plots in one figure.	3.3.2.5. gtreport extensions
UR21	The server produces for every validation process (D1) All the datastreams B^{P1} and associated model data streams $A^{(M1,P1)}$ used in the validation process (after smoothing/interpolations/regridding) (D2) When applied in the validation process, the averaging kernels and apriori profiles	3.3. Validation 2.3.2 Validation outputs
UR22	The server produces for every validation process (T1) The filenames of the NORS data used (T2) The MACC-II models experiment id's and that part of the changelog files of the MACC-II models that covers the time interval under consideration (T3) A detailed overview of the different steps executed in the validation process (i.e. including the toolset command lines used for the data manipulations)	3.3. Validation 2.3.2 Validation outputs

ID	Description	Covered by
UR23	The output format of the data produced by the server is described in the following list. (O1) (Dj) is in HDF format. The metadata should indicate - Theunits - the physical/mathematical quantity, - the data provider (both for NORS and MACC data), - the data creator (e.g., NORS validation server v1) (O2) (Sj<7) in text format (O3) (Sj>7) in pdf and png format. (O4) (Tj) in text format	3.3.2.5. gtreport extensions
UR24	For each scalar NORS product and MACC-II model in which the NORS product appears: (S10) Bias vs measurement time (finest time unit=days) (S11) Relative bias vs measurement time (finest time unit=days) (S13) Modified normalized mean bias vs measurement time (finest time unit=days) E.g. a plot of product (P5) FTIR O3 at LA.REUNION compared against (M1)	2.4.2 NVS Core Application 3.3.3.1 Default validation 3.3.2.5. gtreport extensions
UR25	For each profile NORS product and MACC-II model: (S16) Mean model profile plot and (S18) Mosaic plots	2.4.2 NVS Core Application 3.3.3.1 Default validation 3.3.2.5. gtreport extensions
UR26	For each MACC-II model: (S15) Taylor diagram	2.4.2 NVS Core Application 3.3.3.1 Default validation 3.3.2.5. gtreport extensions
UR27	Overlay plots of type (S12) for (S10) and (S11) (finest time unit=days) Overlay plots of type (S14) for(S13) (finest time unit=days) Overlay plots of type (S14) for (S16) A fixed colour coding is used to distinguish between the models.	2.4.2 NVS Core Application 3.3.3.1 Default validation 3.3.2.5. gtreport extensions
UR28	For each scalar NORS product and MACC-II model in which the NORS product appears: (S10) Bias vs measurement time (finest time unit=days) (S11) Relative bias vs measurement time (finest time unit=days) (S13) Modified normalized mean bias vs measurement time (finest time unit=days)	2.4.2 NVS Core Application 3.3.3.1 Default validation 3.3.2.5. gtreport extensions
UR29	For each profile NORS product and MACC-II model: (S16) Mean model profile plot and (S18) Mosaic plots	2.4.2 NVS Core Application 3.3.3.1 Default validation 3.3.2.5. gtreport extensions
UR30	For each MACC-II model: (S15) Taylor diagram	2.4.2 NVS Core Application 3.3.3.1 Default validation 3.3.2.5. gtreport extensions
UR31	Overlay plots of type (S12) for (S10) and (S11) (finest time unit=days) Overlay plots of type (S14) for (S13) (finest time unit=days) Overlay plots of type (S17) for S(16) A fixed colour coding is used to distinguish between the models.	2.4.2 NVS Core Application 3.3.3.1 Default validation 3.3.2.5. gtreport extensions

ID	Description	Covered by
UR32	The user may choose among the different time windows, the different NORS products (determined by instrument/molecule/locations/affiliations), the MACC-II models. The web interface is interactive: if a user chooses an instrument/molecule/locations/affiliations and a MACC-II model, the web content adapts and shows all plots/results that fit the user's choice. The user can select the final validation product and automatically downloads a zip file containing the statistics/plots along with the underlying data (Dj) and the traceability report (Tj).	3.4 Web Application 3.3.3.2 Custom validation
UR33	All seasonal and monthly data should be stored permanently. The moving time window is updated daily and previous results are not stored.	2.3.2 Validation outputs
UR34	In case NORS data are updated, all seasonal and monthly plots and traceability reports based on these data should be regenerated. In the case MACC-II data is updated (e.g., if an output time is missing and added at a later stage) the server updates all monthly and seasonal plots and traceability reports based on this data. Only the most up-to-date version is available online.	3.3.3.2 Custom validation
UR35	The user may request all types of output data described in Section 8. The time unit in the plots can be chosen among presets: days, weeks, months, seasons.	3.4 Web Application
UR36	After authentication the UINT member is redirected to a webpage on which he/she can chose among the different settings for the specific validation process that will be set up. In particular the user defines the NORS product, the MACC-II model, the desired output. In the definition of the time window, the user may choose the start and end date, or may choose among preset windows per month/seasons. After the validation process finishes, the user will receive a link to a zip file containing the requested data/plots.	3.4 Web Application 3.3.3.2 Custom validation 3.3.3 Task management
UR37	A scheduling option should be possible where the user requests that this configuration for a validation is performed weekly or monthly.	3.4 Web Application
UR38	The user interface slightly differs from the interactive case. A user is allowed to upload his own GRIB file(s) or provide the link(s) to an NDACC GEOMS compliant data file. There should be a compatibility check. The species of the NDACC file and the chosen time window should be compatible with the modelled species in the GRIB files. The menu structure should be adapted: the NORS product fields and model fields are enlarged to contain the extra NDACC and GRIB files.	3.4 Web Application 3.3.3.2 Custom validation 3.3.3 Task management
UR39	He/she is able to adapt existing scripts to add/remove NORS products and MACC-II models.	2.2.4 NORS Admins - NVS
UR40	He/she can adapt the web menu structure to add/remove NORS products and MACC-II models.	2.2.4 NORS Admins - NVS
UR41	He/she can adapt the colour coding for the MACC-II models and modify the default output template.	2.2.4 NORS Admins - NVS