Customer	**:**	**ESRIN**	**Document Ref**	**:**	**SST_CCI-SPD-BC-001**
WP No	**:**		**Issue Date**	**:**	**28 June 2013**
			Issue	**:**	**1.1**

Project : **CCI Phase 1 (SST)**

Title : **CCI-SST System Prototype Description**

Abstract : This document contains the SPD for the ESA SST_cci project.

Author :

R. Quast, N. Fomferra,
M. Boettcher
Brockmann Consult GmbH

Checked :

C. Merchant, Science Leader
University of Reading

P. Spinks, Project Manager
Space ConneXions Ltd.

Accepted by ESA:

Digitally signed by Craig Donlon
DN: cn=Craig Donlon, o=EOP-SME, ou=ESA/ESTEC, email=craig.donlon@esa.int, c=NL
Date: 2013.07.02 12:41:38 +02'00'

Craig Donlon
ESA Technical Officer

Acceptance date:

Distribution : SST_cci team members

ESA (Craig Donlon)

EUROPEAN SPACE AGENCY
CONTRACT REPORT

The work described in this report was done under ESA contract. Responsibility for the contents resides in the author or organisation that prepared it.

sst
cci

AMENDMENT RECORD

This document shall be amended by releasing a new edition of the document in its entirety. The Amendment Record Sheet below records the history and issue status of this document.

AMENDMENT RECORD SHEET

ISSUE	DATE	REASON FOR CHANGE
1.0	29 Jan 2013	Initial version
1.1	28 Jun 2013	Updates according to ESA RIDs CD-1 and CD-2, new introductory section

RECORD OF CHANGES IN THIS ISSUE

Issue	Page/Sec.	Reason	Change

EXECUTIVE SUMMARY

This Sea Surface Temperature (SST) System Prototype Description describes the prototype of the SST system for the European Space Agency (ESA) Climate Change Initiative (CCI). SST is one of 13 Essential Climate Variables (ECV) currently studied by CCI. The SST system will be used to generate and continuously update the SST part of the CCI climate data record (CDR).

The prototype system consists of three sub-systems that are running at different sites: Météo France (CMS) runs a sub-system for producing L3 SST from MetOp and SEVIRI Earth observation data, University of Edinburgh (ECDF) runs the production of L2 and L3 SST from the ATSR and AVHRR series of instruments, and UK MetOffice produces analysis data from L2 and L3 SST.

This document describes each sub-system including the multi-sensor match-up system prototype for supporting continuous algorithm improvement, which is part of the Edinburgh component.

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Purpose and scope

This Sea Surface Temperature (SST) System Prototype Description (SPD) identifies and describes the distributed prototype of the SST system for the European Space Agency (ESA) Climate Change Initiative (CCI). SST is one of 13 Essential Climate Variables (ECV) currently studied by CCI. The SST system will be used to generate and continuously update the SST part of the CCI climate data record (CDR).

Each sub-system of the distributed prototype is described in terms of its infrastructure, its external and internal interfaces, data management, production control, and operational scenarios. The data processors in each sub-system are described in the Sea Surface Temperature Detailed Processing Model (DPM) Document [AD 8].

1.2 References

The following documents are applicable to this document:

ID	Title	Issue	Date
[AD 1]	ESA Climate Change Initiative Phase I - Scientific User Consultation and Detailed Specification Statement of Work (SoW), including Annex G: Sea Surface Temperature ECV	1.4	09.11.2009
[AD 2]	Sea Surface Temperature ECV Proposal		16.07.2010
[AD 3]	Sea Surface Temperature CCI User Requirements Document, SST_CCI-URD-UKMO-001 (URD)	2.0	30.11.2010
[AD 4]	Sea Surface Temperature Data Access Requirements Document, SST_CCI-DARD-UOL-001 (DARD)	1.0	27.01.2012
[AD 5]	Sea Surface Temperature Product Specification Document, SST_CCI-PSD-UKMO-002 (PSD)	2.0	11.11.2011
[AD 6]	Sea Surface Temperature MMD Content Specification, SST_CCI-REP-UOL-001	C	22.07.2011
[AD 7]	Sea Surface Temperature Input Output Data Definition Document, SST_CCI-IODD-BC-001 (IODD)	1.0	28.09.2012
[AD 8]	Sea Surface Temperature Detailed Processing Model, SST_CCI-DPM-BC-001 (DPM)	1.0	04.10.2012
[AD 9]	Sea Surface Temperature System Requirements Document, SST_CCI-SRD-BC-001 (SRD)	1.2	12.04.2012
[AD 10]	Sea Surface Temperature Product Validation Plan, SST_CCI-PVP-UoL-001 (PVP)	1.0	30.01.2012
[AD 11]	Sea Surface Temperature Algorithm Selection Report, SST_CCI-ASR-UOE-001 (ASR)	1.0	30.06.2012
[AD 12]	Sea Surface Temperature Algorithm Theoretical Basis Document, SST_CCI-ATBDv0-UOE-001 (ATBD)	1.0	08.11.2011

The following documents are referenced in this document (see Reference Documents List, SST_CCI-REP-UOE-001):

ID	Title	Issue	Date
[RD 1]	Reference Documents List, SST_CCI-REP-UOE-001	1	27.09.2011
[RD 2]	Acronyms List, SST_CCI-REP-UOE-002	1	27.09.2011
[RD 181]	GBCS Users guide and ATBD		
[RD 296]	C.J. Merchant, O. Embury, N.A. Rayner, *et al.* (2012) A twenty-year independent record of sea surface temperature for climate from Along Track Scanning Radiometers, J. Geophys. Res., 117, C12013		

1.3 Acronyms

The following SST-specific acronyms are used in this report (also see Acronyms List, SST_CCI-REP-UOE-002):

Acronym	Definition
ARC	ATSR Reprocessing for Climate
(A)ATSR	(Advanced) Along-Track Scanning Radiometer
AVHRR	Advanced Very High Resolution Radiometer
BADC	British Atmospheric Data Centre
BEAM	Earth observation toolbox and development platform
CCI	Climate Change Initiative
CF	Climate Forecast
CMIP5	Coupled Model Intercomparison Project Phase 5
DARD	Data Access Requirements Document
DPM	Detailed Processing Model
ECDF	Edinburgh Compute and Data Facility
ECMWF	European Centre for Medium-Range Weather Forecasts
ECSS	European Cooperation for Space Standardisation
ECV	Essential Climate Variable
ESA	European Space Agency
GBCS	Generalised Bayesian Cloud Screening
GDS	GHRSST Data Processing Specification
GHRSST	Group for High-Resolution SST
GMPE	GHRSST Multi Product Ensemble
IR	Infrared
MetOp	Meteorological Operational (EUMETSAT)
MD	Match-up Dataset (single-sensor)
MMD	Multi-sensor Match-up Dataset

Acronym	Definition
MMS	Multi-sensor Match-up System
NOAA	National Oceanic and Atmospheric Administration
NEODC	NERC Earth Observation Data Centre
NERC	Natural Environment Research Council
NWP	Numerical weather prediction
OSI-SAF	Ocean & Sea Ice Satellite Application Facility (EUMETSAT)
OSTIA	Operational Sea Surface Temperature and Sea Ice Analysis
PMW	Passive Microwave
SDI	Saharan Dust Index
SEVIRI	Spinning Enhanced Visible and Infrared Imager
SGE	Sun Grid Engine
SST	Sea Surface Temperature
UoE	University of Edinburgh

Additional acronyms are listed in [RD 2].

1.4 Document structure

After this formal introduction

Section 2	describes the activities in the SST CCI project and their relation to components in the SST CCI prototype and future systems
Section 3	provides an overview of the SST CCI prototype with its interfaces for users and for data exchange and the decomposition of the prototype into distributed subsystems.
Section 4	describes the design of the multi-sensor matchup system which is hosted in the ECDF subsystem
Section 5	describes the design of the CMS subsystem of the prototype, which is part of the OSI SAF, with infrastructure, interfaces, data management and production control, and its operational scenarios
Section 6	describes the design of the ECDF subsystem of the prototype
Section 7	describes the MetOffice subsystem of the prototype, which is part of the OSTIA system. Again, infrastructure, interfaces, data management, production control, and operational scenarios are listed.

2. THE SST CCI PROJECT AND ITS ACTIVITIES

In the following we distinguish the "prototype system" from the "prototype processor". The prototype system is the more general term covering the end-to-end capabilities developed within SST CCI Phase 1. The prototype processor specifically refers to the chain by which products specified in the Product Specification Document (PSD) are created [AD 5].

2.1 Cardinal requirements and system overview

The Climate Change Initiative (CCI) Sea Surface Temperature (SST) project of the European Space Agency (ESA) aims at providing SST satellite data records to meet the requirements of the climate research community [AD 3].

Three of ESA's five cardinal requirements (CR) name the main outputs that are expected from the CCI project: "climate-quality" algorithms (CR-1), "world-class" time series of ECV products (CR-2), and complete specifications for an operational production system (CR-4).

In Phase 1 of the SST CCI project a prototype production system has been built, which generates world-class SST products [AD 5] (CR-2) and has been used for a range of activities (see below) for climate quality algorithm development and testing (CR-1).

For addressing CR-4, the SST CCI team has used ESA's cardinal requirements, SST technical requirements, the user community's user requirements, and the inherent knowledge of the SST CCI team to specify the system requirements for the SST CCI Phase 2 system (hereafter referred to as "the future system") in the SRD [AD 9]. The technical specification of this future system will be described in the System Specification Document (SSD).

The SST CCI prototype processor has been implemented, hosted and maintained at University of Edinburgh (a cluster known as ECDF), Centre de Meteorologie Spatiale (CMS, using some developments within the Ocean and Sea Ice-Satellite Applications Facility, OSI-SAF), and the MetOffice (OSTIA, the operational surface temperature and ice analysis system). The ECDF provides a high-performance cluster of servers connected to a large amount of high-performance fibre channel disk storage so that each node in the cluster sees the same data in the same way.

The prototype processing system is capable of processing about 50 TB of input data into about 5 TB of output data in a highly parallel fashion. The ECDF offers a Sun Grid Engine (SGE) processing grid, which is able to distribute workload across the ECDF resource pool by an efficient scheduling of jobs.

The SST CCI user products comprise L2P data (Level 2), uncollated and daily-collated Level 3 products, and daily-analysed SST from OSTIA (Level 4). At the end of Phase 1 of the project, SST products will have been produced and be available for the long-term (1991–2010) and demonstration (June & August 2007 and January to March 2012) periods. The products have been fully specified in the PSD [AD 5].

Though several elements of the prototype production system can be used in the future system, not all elements of the future system have been fully prototyped. In particular, the elements for supporting a continuous user-driven improvement of retrieval algorithms have not been implemented in a sustainable manner. The elements and activities needed for implementing the continuous algorithm improvement become clear by illustrating the mode of operation for the future system as specified by the team (Figure 2-1, trapezoidal shapes indicate manual activities).

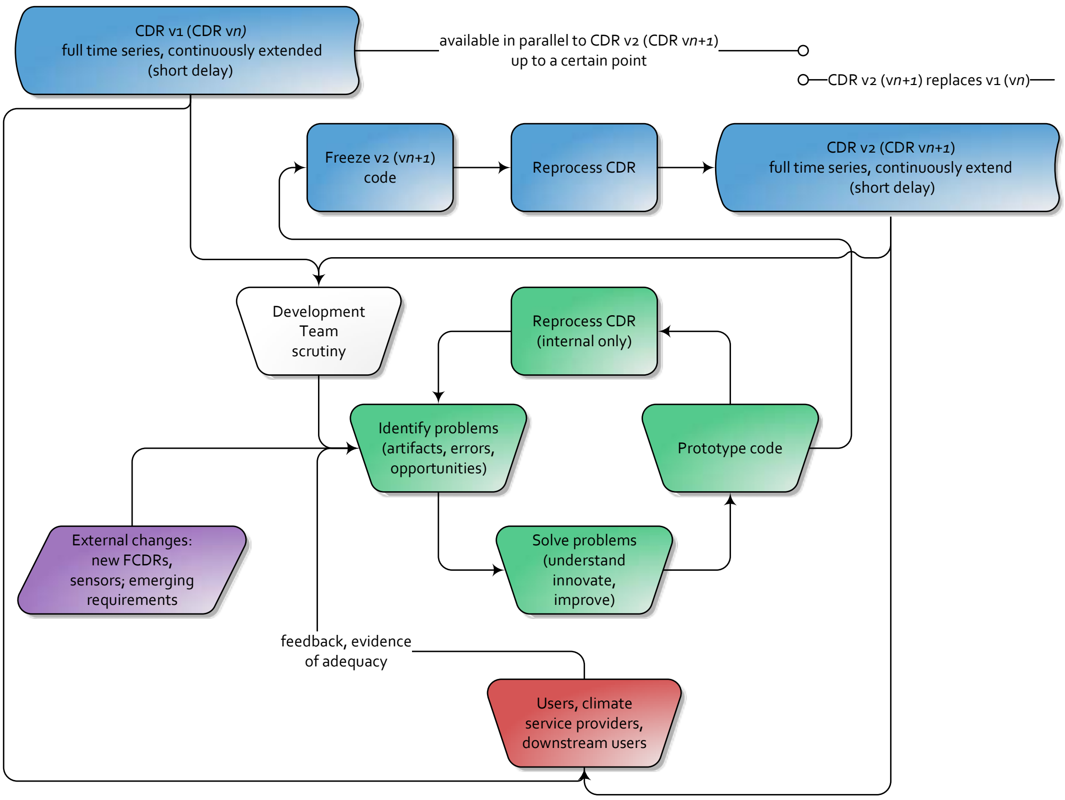

Figure 2-1: Algorithm improvement triggered by user feedback

Improvement of retrieval algorithms used in the future system will be essentially user-driven. The model of improvement involves the following elements and steps:

1. There is a stable climate data record (CDR) of a certain version. This stable CDR is continuously extended by newly acquired data in short-delay processing mode (blue cylindrical shape at the top of Figure 2-1, the initial CDR v1 is output of the prototype system).
2. Climate users and SST community users as well as the SST CCI Development Team itself assess the data and provide feedback. This feedback can trigger a need for an internal algorithm improvement cycle (green boxes and connecting arrows). External changes like new versions of input data, new sensors, and new emerging requirements can trigger the need for an improvement cycle as well.
3. The expectation is that the improvement cycle will be conducted repeatedly and rapidly in reaction to different identified problems. The cycle involves data ranging from multi-sensor match-up datasets to the full FCDR. The individual activities in the internal improvement cycle are:
    a. Using evidence from users and the development team's scrutiny to analyse and identify problems
    b. Problem solving by problem understanding and suggesting algorithm improvements and innovations
    c. Prototyping retrieval algorithm changes
    d. Extending multi-sensor match-up datasets with recomputed retrieval results and, if applicable, internal reprocessing of the CDR
4. Consolidation of retrieval algorithms after internal improvements leads to a 'freezing' of the source code. Reprocessing of the full FCDR with the 'frozen' algorithms gives the next version of the CDR (users have made clear that this shall not occur more often than once in a year). Both the older and the newer version of the CDR (v2, second blue cylindrical shape at the top of Figure 2-1) are available in parallel for a certain period of time.
5. The newer version of the CDR replaces the older version; it is extended and improved by the activities described in the previous Steps 1 to 4.

Key elements of the algorithm improvement are the implementation of the Development Team and the management of user feedback and requirements. From the technical point of view, the activities of the Development Team during the internal algorithm improvement cycles (green boxes and connecting arrows in Figure 2-1) need to be supported by a largely automated system component capable of computing match-ups of multiple *in-situ* and sensor and ancillary data sources. The rules and methods for this multi-sensor match-up system (MMS) have been implemented in Phase 1 of SST CCI; implementing the automated operation within the future system is foreseen for Phase 2. Similarly, the management of user feedback has only been marginally implemented in Phase 1 through an electronic mail interface. The purpose and scope of the MMS and the related activities are further described in Section 2.3

Figure 2-2 illustrates a high-level decomposition of the future system into its essential functional elements and activities. Rectangular shapes indicate technical elements (Level 2/3 and 4 processors, MMS, distribution system for documents, data, and tools). Cylindrical shapes indicate data storage elements (satellite and ancillary input data, validation and other ancillary data, SST CCI products and MMD files) while trapezoidal shapes indicate manual activities (user feedback management, algorithm development, data ingestion for verification and validation). Note that the Development Team does not appear in the diagram explicitly, though its members conduct the manual activities and operate the system as a whole. The colouring of shapes is used to discern elements receiving external input (orange), elements distributing output to external users (purple) and core elements (blue). Boldface letters denote elements, activities or functions that have been prototyped in Phase 1; plain letters denote elements that have been prototyped for the demonstration but not for the long-term production in Phase 1; italic letters indicate functions that have not been prototyped in Phase 1, but will be addressed for implementing the future system in Phase 2.

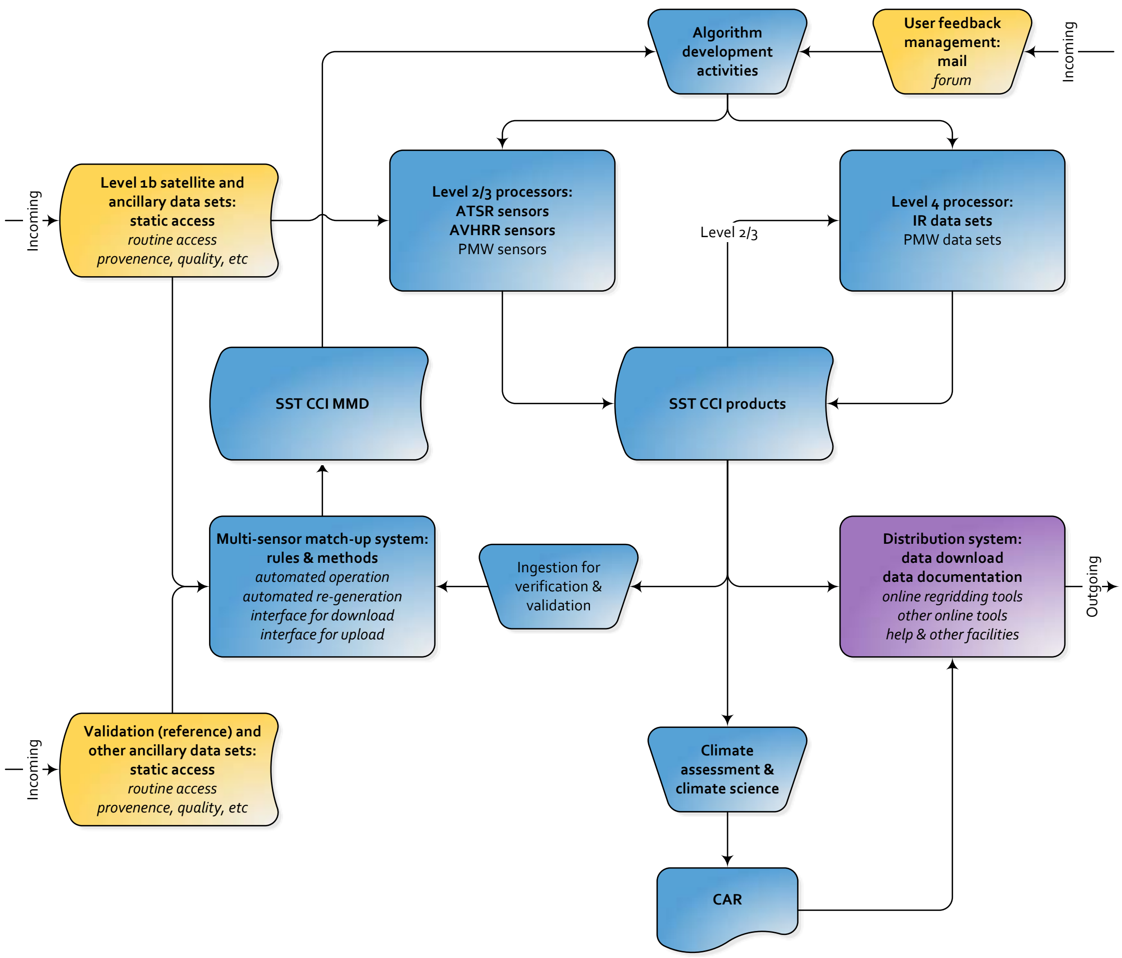

Figure 2-2: High-level decomposition of the future system

The remainder of this overview briefly describes and discusses the system elements and functions in turn. Where these elements have supported major activities within Phase 1, these activities and the role they fulfil are also described.

2.2 Input data access

The access requirements and procedures for all data that have been needed as input to perform the SST CCI project are listed and described in the Data Access Requirements Document [AD 4]. The data include:

- Satellite products from ESA and third party missions (e.g. Level 1b from ATSR-1, ATSR-2, and AATSR; NOAA AVHRR GAC Level 1, MetOp internal Level 1b, SEVIRI Level 3C, AMSR-E GHRSST L2P, TMI GHRSST L2P)
- Ancillary data (e.g. ECMWF)
- *In-situ* observation data sources as well as higher-level products needed for product inter-comparison
- Historical archives and currently operational sources

All data have been available for the SST CCI project via FTP, SFTP or HTTP to obtain from source and have been downloaded in manual activity. The data are accessible statically at ECDF.

2.3 Multi sensor matchup system (MMS)

The MMS is a component of the prototype that is a novel development. It is capable of computing matchups between in-situ and satellite data from different sensors, and of generating multi-sensor matchup datasets (MMD) of matchups that can include satellite sub-scenes, in-situ records, NWP ancillary data, and processed SST.

One challenge for the MMS has been the heterogeneous input with respect to data content, format, and temporal and spatial coverage. In the prototype MMS the matchups have been based on pre-matched single sensor matchups. The data types and sensors that have been included in Phase 1 are:

- Single-sensor matchups from ATSR1, ATSR2, AATSR, MetOp AVHRR, SEVIRI, and NOAA AVHRR
- Level 1 satellite images from ATSR1, ATSR2, AATSR
- Level 1 satellite images from MetOp AVHRR
- Level 1 satellite images from AVHRR from NOAA-10 to NOAA-19
- Level 2 satellite images from AMSR-E
- Level 2 satellite images from TMI
- Level 3 Aerosol and sea ice concentration
- *In-situ* observation trajectories
- ECMWF era-interim ancillary data
- Results of SST retrieval from ARC CCI processing in MMD format

In an early period of Phase 1 of the SST CCI project the MMS was used to generate a complete set of MMDs for the years 1991–2010 containing more than 6,000,000 matchups. These MMDs have been the basis for doing further work. The initial complete run of the MMS has revealed several types of errors in input data and processing, which have required a semi-automated approach to analyse the cause of errors and to handle errors. The MMS has been verified by comparing its results to the outputs of an independent implementation of the corresponding matchup and extraction algorithms. In addition, all types of outputs have been manually inspected and compared with corresponding input data values on a sample basis.

The MMS database contains all information necessary for doing queries and extracts on spatial and temporal criteria, and criteria of satellite combinations contained in matchups. The MMS already provides the infrastructure to compute matchups on the basis of *in-situ* data and satellite data without pre-computed single sensor matchups, and contains functions to extend the matchups into the future with newly ingested inputs. These functions, however, have not been exploited in Phase 1.

In Phase 1 the MMS has supported the following activities:

- Algorithm development by the SST_cci team for SST retrieval
- Round robin exercise for SST retrieval algorithm decision
- Algorithm development for classification in high latitudes
- Algorithm development for ocean thermal skin model
- Algorithm development for near-surface ocean turbulence model
- Algorithm development for uncertainty estimation
- Product verification
- Product validation (including stability assessment and uncertainty validation)
- Climate assessment

The MMS is therefore a key component in the problem identification and problem solving components of Figure 2-1, as well as in verification and validation of products. The prototype MMS is described in further detail in Section 4.

2.4 Level 2/3 processor

The SST CCI system prototype has been based on software from an earlier project, ARC [RD 296, Merchant et al., 2012]. Functionally, it performs pre-processing (including co-location adjustments), Bayesian cloud detection (which involves associating NWP fields with satellite data and fast RT modelling), coefficient-based and optimal estimation retrieval (first version, denoted OE1) of SST, and creation of 0.05° resolution daily outputs. Another project under UK national funding created additional processing modules for OE1 SST retrieval using AVHRR-GAC inputs. This work was completed prior to the start of the CCI project.

The SST CCI processor development has involved several modifications of the ARC basis including modification of auxiliary files (e.g., updated coefficient files) and development of new software modules. In particular, new modules have been developed for using prior SST gradients for cloud detection, for optimal estimation retrieval (second version, denoted OE2) for ATSRs and AVHRR-GAC, and for conversion of outputs to L2P. In addition, the cost parameter from the optimal estimation retrievals has been used to improve cloud detection. The ARC-based prototype Level 2/3 processor is described in further detail in Section 6.

The creation of MetOp AVHRR SSTs for SST CCI has built on the MetOp operational production at OSI-SAF. Internal to the OSI-SAF processor, work files at 0.05° spatial resolutions are created. These work files are an intermediate product representing considerable additional value compared to the MetOp satellite observations, since cloud detection and confidence level processing have been conducted, and, in addition, useful auxiliary information has been associated with the satellite data. The work files have been made available to the CCI project on an as-is basis. They are property of the EUMETSAT OSI-SAF, and their use within the CCI project is formally acknowledged.

Within the SST CCI project, these work files have been augmented with outputs of fast radiative transfer simulation (using RTTOV9), in order to support OE1 and OE2 retrieval of SST. These new work files contain all fields necessary to support OE1 and OE2 retrieval of SST. The system demonstrated under the CCI therefore consists of a new processor to generate OE2 SSTs for MetOp, which has been demonstrated on the ECDF cluster.

SSTs from SEVIRI are provided by the OSI-SAF operational and experimental systems. SEVIRI OSI-SAF products are copyright of EUMETSAT and their use within the CCI project is formally acknowledged. The OSI-SAF prototype system for SST CCI is described in further detail in Section 5.

A summary of the structure of the infrared (IR) processor, showing the origin of different elements, is illustrated in Figure 2-3. Note that the radiative transfer modelling (RTM) call is not relevant to the MetOp and SEVIRI choice of cloud detection (Bayesian or OSI-SAF). For AVHRR-GAC, CLAVR-X cloud detection has been embedded into the ARC system under the NCEO project.

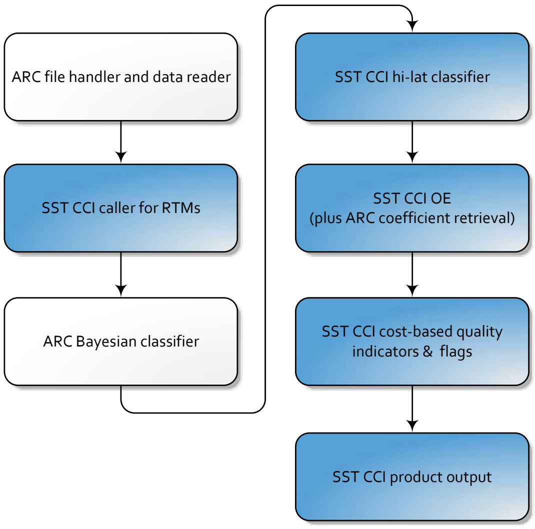

Figure 2-3. Schematic of IR processor, showing components carried over from ARC and those (blue boxes labelled SST CCI) developed in this project

Passive microwave SSTs are as obtained from Remote Sensing Systems (RSS). Useful and practicable improvements in PMW SST uncertainty characterisation are expected, and so an MW post-processor module has been developed to update PMW SSTs with improved uncertainty estimates and output in SST CCI L2P format.

2.5 Level 4 processor

The L2P SSTs produced by the Level 2/3 processing chains have been used for Level 4 SST analysis in a specific version of the Met Office's OSTIA system. The Level 4 analysis is based on the operational OSTIA configuration, but improved via exploitation of both the improved SSTs and uncertainty estimates contained new products. The long-term SST CCI Level 4 product is a daily mean SST depth analysis, whereas the operational OSTIA is foundation SST analysis.

Since the Level 2/3 SST retrievals for ATSR and AVHRR have aimed to be independent of in situ observations the possibility of OSTIA providing a truly independent satellite-only analysis has been raised. The feasibility and benefit of this possibility has been assessed and realised. The prototype Level 4 analysis system is described in further detail in Section 7.

2.6 Distribution system and user feedback management

There has been no dissemination system within the prototype SST CCI system. However, an SST CCI project website has been created that is available at http://www.esa-sst-cci.org (see Figure 2-4).

Figure 2-4: Entry page to SST CCI website

The website provides a project overview, information about the project team, the project plan, information about the round-robin procedure, access to public project documents, contact points, and a collection of frequently asked questions. An elaborate SST retrieval system is scoped for Phase 2 of the project, as outlined in the systems requirement document [AD 9]

For the lifetime of the prototype, public access to the SST CCI products is provided via HTTP and FTP to the ECDF NAS. Afterwards the data will be sent to the GHRSST Long Term Stewardship and Reanalysis Facility (LTSRF) for archiving.

The Climate Data research package (CDRP) is linked from the SST CCI website, but physically served from a remote FTP service hosted by CEDA. A simple registration on the SST CCI website will provide the access information to the user by email, and the user can download the data packages.

Tools for re-gridding and regional averaging have been developed in the project to generate subset products and composites with uncertainty propagation. These tools will be available for download when the development and verification is completed. Integration of these tools into a web service is required for the future system to be developed in Phase 2 of the project [AD 9].

Access to MMD datasets for the round-robin exercise has been provided by FTP. A registration by email provides the access information to download the MMD files and to upload MMD result files for participation in the round-robin comparison.

There have been no automated or tracked user feedback mechanisms within the prototype. However, all SST CCI product files contain metadata referring users to the project website at http://www.esa-sst-cci.org and a contact email science.leader@esa-sst-cci.org.

A system for managing user feedback is scoped for Phase 2 of the project, as outlined in the systems requirement document [AD 9].

2.7 Activities in Phase 1 using the system prototype

2.7.1 Algorithm development for SST retrieval

Algorithm development activities have focused on improving heritage development that were brought to (ARC software) or made available (OSI-SAF processor work files, Met Office's OSTIA system) for the project. The main algorithmic improvement has been the further development of optimal estimation (OE) SST retrieval in order to achieve independence of *in-situ* observations.

Earlier OE retrieval was by design tied to the calibration of *in-situ* observations, mostly from drifting buoy observations because of their availability throughout the 1991 to 2010 period. Being an optimal estimation method, it requires use of a fast radiative transfer model (RTM), and only gives low-bias (GCOS compliant) SSTs, if the fast RTM has low bias when simulating brightness temperatures (BTs) relative to the observed sensor BTs. The bias correction is implemented on BTs (radiances), not on SST. This is preferable, since then the optimal estimator minimises the risk of introducing patterns of bias in SST, which may happen if bias correction is attempted directly on SSTs. An overview of the design for the new OE retrieval is illustrated in Figure 2-5 (blue boxes indicate new SST CCI developments).

Independence has been achieved by fully exploiting the ATSR series (the only series whose accuracy is adequate for the purpose) as a reference sensor. The bias correction is undertaken on BTs (radiances) for all IR sensors, using ATSR SSTs based on fully updated radiative transfer modelling. This approach has been intended to ensure that the retrieved SSTs ultimately exhibit low bias relative to ATSR SSTs, and independence.

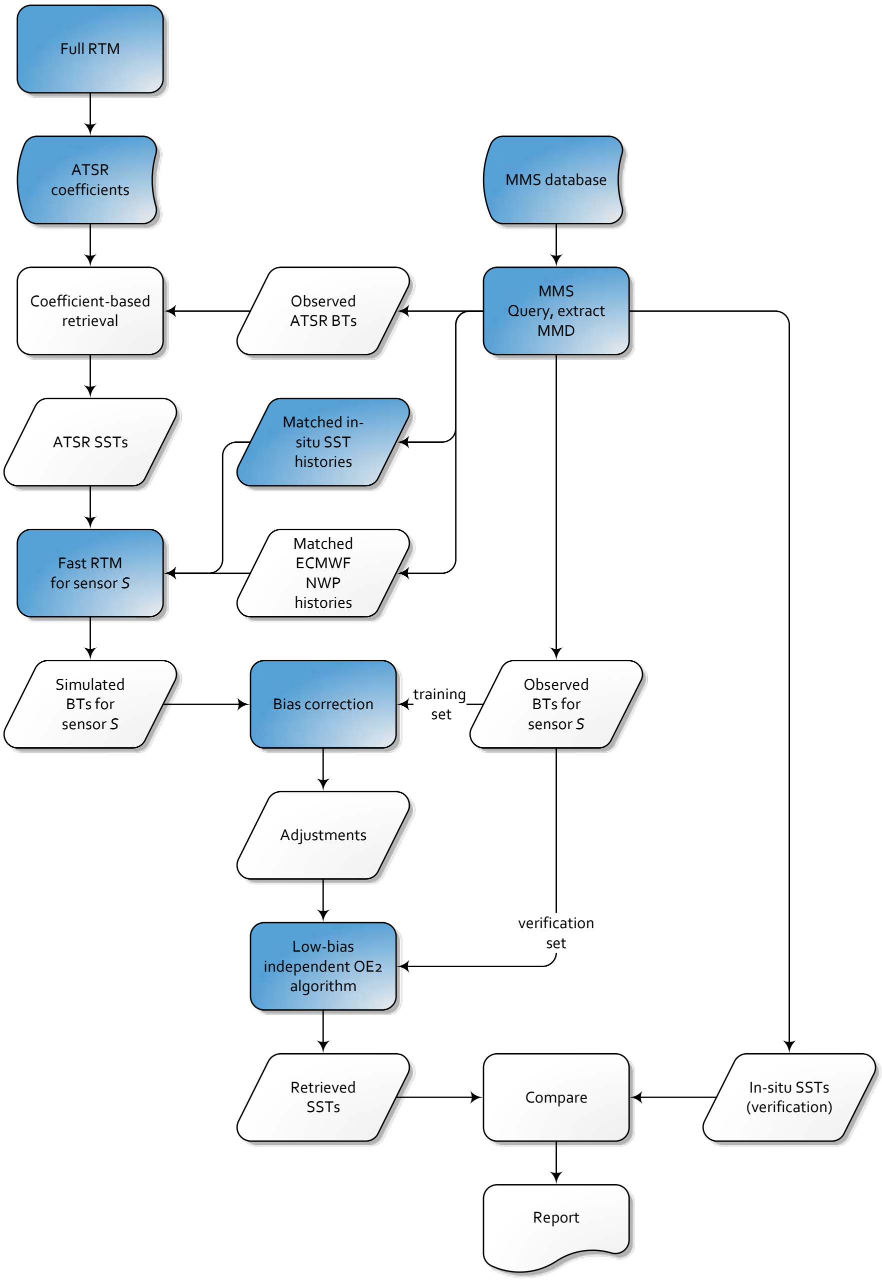

Figure 2-5. Development of in-situ independent OE2 retrieval for AVHRR and SEVIRI with by radiance bias correction to ATSR

2.7.2 Round-robin algorithm comparison and selection

In order to identify the best performing retrieval algorithm or combination of algorithms, the SST CCI project ran an open algorithm selection exercise. This consisted of algorithm intercomparison (described in ESA documents as the "Round Robin") followed by selection of algorithms following criteria defined in the Product Validation Plan [AD 10]. It is expected that future algorithm selection exercises will have to be carried out for each subsequent reprocessing to ensure the best performing algorithm is always implemented.

The exercise was open over a four-month period ending 31st January 2012 and involved both the project team and external participants. Ten external teams expressed interest in participation, of which two were able in practice to submit algorithm selection results in time for consideration. The submitted external algorithms were cutting edge algorithms of significant interest. Relevant to the ATSR series was the Oxford-RAL Aerosol and Cloud retrieval (ORAC, submitted by RAL), an advanced optimal estimator recently extended to include SST, although only applicable to daytime scenes. Relevant to the AVHRR series was Incremental Regression (IR, submitted by NOAA), which is a powerful fusion of model-based and empirical regression approaches. The internal algorithms included existing coefficient based retrievals for the ATSRs, and a day-and-night (infrared only) optimal estimator tuned (for both ATSRs and AVHRRs) to ATSR SSTs. ORAC as currently formulated is not sufficiently general because it doesn't apply to night-time scenes and gave out-of-target biases. Optimal estimation was selected as the best available, most consistent and independent algorithm for use by SST CCI for ATSR and AVHRR sensors. The whole selection procedure is described in the Algorithm Selection Report [AD 11].

To participate in the SST CCI round-robin algorithm selection exercise, participants obtained information about accessing the multi-sensor (ATSR-2, AATSR, MetOp AVHRR, NOAA-17 AVHRR, NOAA-18 AVHRR and NOAA-19 AVHRR) matchup data set provided as the common data set for the exercise, the document explaining the data contents, and the round-robin protocol, which sets out the procedures for involvement.

An important aspect of the exercise has been the approach taken to ensure an objective comparison of algorithms. Most importantly, subsets of the provided information have been earmarked for particular uses: as training data (for algorithm development, including and empirical tuning), testing data (for internal testing of results by participants) and selection data (reserved for calculation of selection metrics, not used in algorithm development). This concept is illustrated schematically in Figure 2-6.

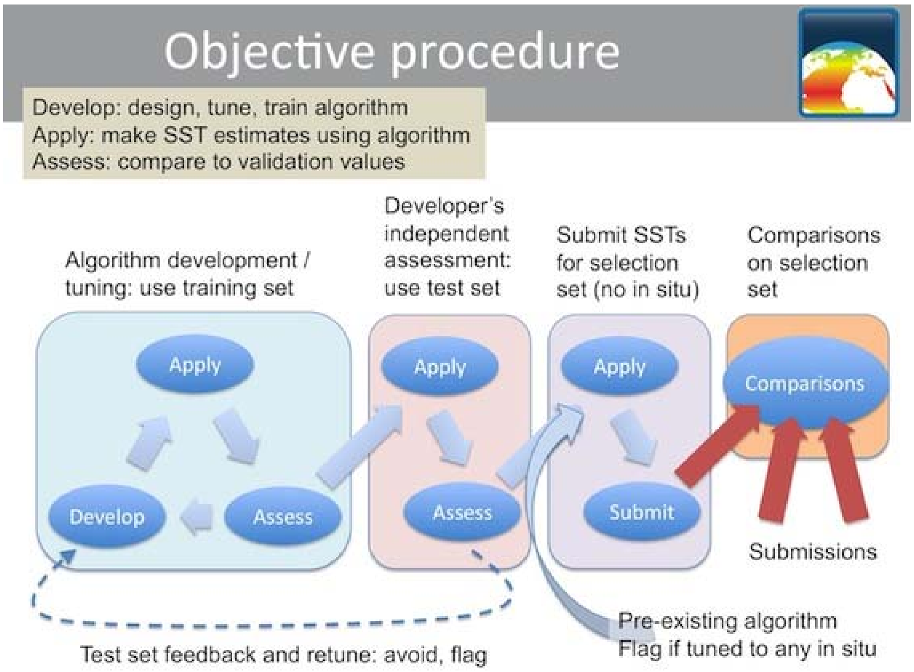

Figure 2-6: Objective procedure for comparing algorithms in round-robin exercise

The selection criteria for the SST CCI round-robin algorithm selection exercise were pre-defined before the start of the activity. All assessments were carried out with reference to drifting buoys. Further details on each criterion can be found in Section 4 of the Product Validation Plan [AD 10].

To support the development and validation activities, the SST CCI project has created a multi-sensor match-up dataset (MMD) of temporal and spatial coincidences between multiple satellite datasets of both brightness temperatures and SST retrievals and time series of SST from in situ sensors (such as a drifting buoy). The round-robin data package (RRDP) essentially has been an MMD, which includes multi-sensor matchup data records for training, test, and algorithm selection. The MMD Content Specification [AD 6] provides a detailed description of contents and format.

The round-robin exercise has been a major driver for the requirements on the system for producing MMD files, the MMS, which are described in the System Requirements Document System Requirements Document [AD 9].

2.7.3 Other algorithm developments

Besides the algorithm developments for SST retrieval summarised in Section 2.7.1, work has been carried for developing methods for SST uncertainty estimation, SST depth adjustments, SST diurnal adjustments, and further developing methods for detecting clouds and classifying sea ice at high latitudes. The development results are described in the Algorithm Theoretical Basis Document [AD 12].

Again, similar to the procedures illustrated in Figure 2-3 and Figure 2-5, using MMD files created by the MMS has facilitated the algorithm development and validation tasks.

2.7.4 System verification

The verification activities in SST CCI Phase 1 will be fully described in the System Verification Report (SVR). The activities address four areas of functionality: prototype processors (create SST CCI prototype products), multi-sensor match-up system (MMS, supports CCI science behind products), tools applicable to SST CCI products (for aggregating SST data), and data provision (user access to the Climate Data Research Package).

Verifying the MMS by spot-checking and manually inspecting MMS database records and MMD files created for the round-robin exercise (see Section 2.7.2) has been an early activity within the project.

Verifying the prototype processors with focus on verification of the products they produce are the key activities within the still ongoing verification tasks. Besides checking product files for completeness of content and consistency with the product specification, all files are checked for content ensuring that all variables have values within the specified limits or a fill value. In particular, retrieved SST values and uncertainties are further spot-checked by verifying SSTs extracted from the outputs to be equal to (within a certain tolerance within required accuracy limits) to SSTs calculated independently from multi-sensor matchup dataset (MMD) files.

2.7.5 Product validation and climate assessment

The validation and climate assessment of SST CCI products is a work still to come at the writing of this document. In brief, the SSTCCI Level 2, Level 3 and Level 4 products will be independently validated using high-quality SST measurements made *in-situ* from a number of sources. In addition, the SST CCI Level 4 products will be compared to other Level 4 products as part of the Group for High Resolution SST (GHRSST) Multi Product Ensemble (GMPE) and other intercomparisons carried out as part of the Climate Assessment Report (CAR). The CAR will also include other kinds of assessment, as detailed in the Product Validation Plan [AD 10].

2.7.6 Summary of project activities

A summary of the activities of the SST CCI project carried out in Phase 1 and how these activities are related to the prototype and the future systems is given in Table 2-1.

Table 2-1: Summary of project activities

Activity in Phase 1	Prototype system	Future system
Defining user requirements	None	User requirements have been a source for the future system's System Requirements Document. New user requirements can trigger algorithm improvement activities as illustrated in Figure 2-1

Data access	Static datasets (historical data) have been obtained	In addition need routine (short-delay) access to ongoing missions. Extend datasets used to IASI and SLSTR. Update with new Level 1b where relevant.
Input data provenance, quality control, etc.	Has been carried out ad hoc as required, e.g., by noting and reporting corrupted data when found in processing	Automated checking of data; provenance control, e.g., track version of data used in different outputs
Development and creation of multi-sensor match-up rules and datasets	Has been applied in a single MMS run, using in part pre-calculated match-up datasets for individual sensors	Same rules and methods, expanded to new datasets. Automated operation and generation of matches (in-situ, high-latitude, and clear-sky satellite-satellite)
Algorithm development activities	MMS has supported use of controlled subsets of data for algorithm development activities. Retrieval development, classification, skin-depth adjustment, and uncertainty algorithms	Interface needs to be defined and implemented to configure subsets of MMS output for automated extraction and delivery, tailored to particular investigations
Upgrading and generalising prototype processor	Improved and newly developed algorithms have been implemented. ECDF prototype has been extended in order to be able to process MMD input	Need for capability of processing MetOp FRAC and SLSTR. Assess migration of the ECDF part (including MMS) to CEMS
Round-robin exercise	MMS supported controlled, blind Round Robin protocol, with receipt of external contributions	Interfaces for external RR participants to download and upload MMD and automated ingestion of contributions need to be defined and implemented
Product generation	Has been carried out once, for the long-term (1991—2010) and demonstration (June and August 2007, and January to March 2012) periods	Algorithm improvement will be triggered by user feedback and external changes, followed be reprocessing of the full CDR as decided by the Development Team
Product verification	Verification procedures have not been integrated into the prototype	Verification will be integrated into the future system: verification tools, automated ingestion from products into MMS
Product validation	To be carried out using MMS outputs in line with the PVP	Requires more standard tools for routine validation on generation of new products
Climate assessment	To be carried out using products and MMS outputs	Requires automatic generation of standard assessment metrics on generation of new products

3. USER INTERACTION AND EXTERNAL INTERFACES OF THE SST CCI PROTOTYPE

This section describes the SST CCI project website, the access to data by users, and the email-based user feedback handling. The section also provides an overview of the other external interfaces of the system.

3.1 SST CCI project website

The SST CCI project website is available at http://www.esa-sst-cci.org (see Figure 2-4 on Page 15) and provides:

- A project overview
- Information about the team
- The project plan
- Information about the Round Robin exercise
- Access to project documents
- Contact points
- Frequently asked questions

The website also provides information how to access data. Data access is described in the following subsections.

3.2 Output data access by users

During the lifetime of the prototype, public access to the output products will be via http / ftp access to the output products stored on the ECDF NAS. Afterwards the data will be sent to the GHRSST Long Term Stewardship and Reanalysis Facility (LTSRF) for archiving.

The CDRP is planned to be available via the SST CCI Web page, but physically served from a remote FTP service hosted by CEMS. A simple registration on the Web page provides the access information to the user by email, and the user can download the data packages.

3.3 MMS data access by users

Access to MMD datasets for Round Robin has been provided by FTP to the ECDF NAS. A registration by email provides the access information to download the MMD data files and to upload MMD result files for participation in the Round Robin comparison.

3.4 User feedback handling in the prototype

There are no automated or tracked user feedback mechanisms within the prototype. However, all output files contain metadata referring users to the project website http://www.esa-sst-cci.org and a contact email science.leader@esa-sst-cci.org

A system for user feedback is scoped for Phase 2 of the project as outlined in the systems requirement document [AD 9].

3.5 SST prototype system context

The prototype SST CCI system has interfaces to its data providers and its users. Figure 3-1 shows the SST CCI system in its context.

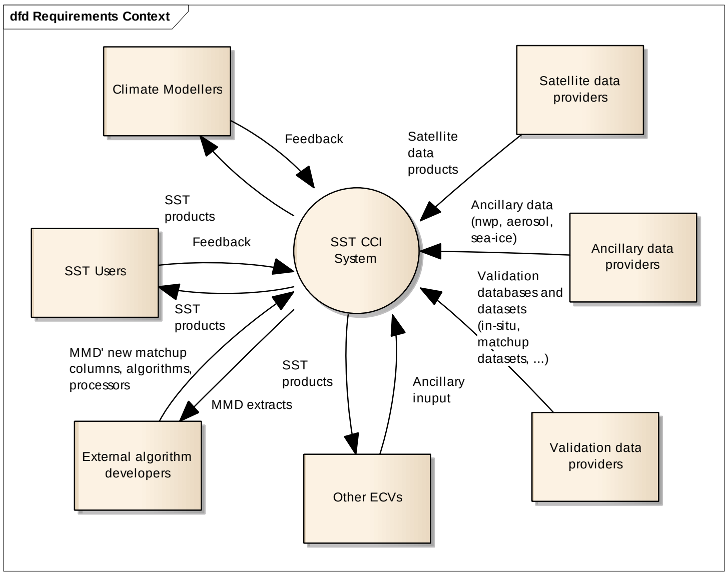

Figure 3-1: Context of the SST CCI system with data providers and users

Climate modellers and users from the SST community receive SST products and documentation. The SST CCI development team actively requests feedback from climate modellers and the SST community. The interface to satellite data providers is the Level 1 product. The interfaces to ancillary data providers are the ancillary data products. Sea ice and aerosol as well as other ECVs provide comparison data to SST; other ECVs can receive SST products for consistency checks. The interface to external algorithm developers is the multi-sensor match-up dataset (MMD) [AD 6]; Validation data providers provide their data as match-up datasets in the MMD format [AD 6].

3.6 External and internal interfaces

Table 3-1 lists the external interfaces of the SST CCI prototype. There are six main interfaces with the satellite input data interfaces on one end and the SST Climate Research Data Package on the other end. Some of the interfaces have different endpoints. Therefore, they are split into corresponding sub-interfaces. In order to avoid too much duplication with the DARD [AD 4] the list of sources for the validation data are not repeated here. DARD section 6 and 7 provide a comprehensive list including access information.

Table 3-1: External interfaces of the SST CCI prototype

Ifc ID	Interface Name	Source	Location/Protocol	Interface item description
Ifc-1	**Satellite input data interface**	Various	FTP pull	DARD, section 4
Ifc-1.1	(A)ATSR input data interface	ESA	FTP from NEODC	DARD, section 4.1
Ifc-1.2	AVHRR GAC L1 input data interface	NOAA	Tape, NOAA CLASS/University of Maryland	DARD, section 4.3
Ifc-1.3	MetOp AVHRR L1B input data interface SEVIRI L3C input data interface	Eumetsat	CMS	DARD, section 4.4, 4.7
Ifc-1.4	AMSR-E L2P input data interface	NASA	FTP from GHRSST LTSRF	DARD, section 4.9, 4.10

Ifc ID	Interface Name	Source	Location/Protocol	Interface item description
	TMI L2P input data interface			
Ifc-2	**Ancillary input data interface**	Various	FTP pull	DARD section 5
Ifc-2.1	ERA-Interim	ECMWF	FTP from BADC	DARD, section 5.1
Ifc-I2.2	CLAVR-X	NOAA	FTP from University of Wisconsin	DARD, section 5.2
Ifc-2.3	NCEP/NCAR Reanalysis	NOAA NCEP	FTP from NOAA	DARD, section 5.3
Ifc-2.4	OSI-SAF Maximum Gradient Atlas interface	CMS	CMS	DARD, section 5.4
Ifc-2.5	AOML Ocean Current Climatology Atlas interface	AOML	AOML	DARD, section 5.5
Ifc-2.6	SSM/I Sea ice concentration maps OSI-409 Sea ice concentration reprocessing	OSI SAF	FTP from OSI SAF High Latitude processing centre	DARD, section 5.6, 5.7
Ifc-2.7	Absorbing aerosol index	NASA GSFC, TEMIS	NASA GSFC TEMIS	DARD, section 5.8
Ifc-2.8	SAGE II Aerosol	NASA	NASA	DARD, section 5.9
Ifc-3	**Validation input data interface**	Various	FTP pull	DARD section 4
Ifc-3.1	ATSR match-up dataset interface	ESA	UoL	DARD, section 4.2
Ifc-3.2	AVHRR MetOp match-up dataset interface SEVIRI match-up dataset interface	Eumetsat	CMS	DARD, section 4.5, 4.8
Ifc-3.3	AVHRR Pathfinder match-up dataset interface	RSMAS	RSMAS	DARD, section 4.6
Ifc-3.4	In-situ data interface	various	FTP pull	DARD, section 6
Ifc-3.5	Inter-comparison data interface	various	FTP pull	DARD, section 7
Ifc-4	**ECV consistency check data interface** Data from OC, SL, sea ice thickness, sea ice concentration, aerosol optical depth, clouds	ESA CCI	FTP pull	DARD, section 8 NetCDF-4 files of complete data records
Ifc-5	**MMS data exchange interface**	SST CCI	Email, FTP	MMD content specification [AD 6]
Ifc-5.1	MMD retrieval interface for	SST CCI	Web form/email for	MMD content

Ifc ID	Interface Name	Source	Location/Protocol	Interface item description
	Round Robin algorithm developers		access information, FTP pull from ECDF	specification [AD 6]
Ifc-5.2	MMD transfer interface for Round Robin algorithm developers	SST CCI	FTP push to UoL	MMD content specification [AD 6]
Ifc-6	**Climate user and SST user interface**	SST CCI	HTTP, FTP, email	see below
Ifc-6.1	SST CCI Web interface	SST CCI	HTTP	Section 3.1 above
Ifc-6.2	SST CCI feedback interface	SST CCI	email	Section 3.4 above
Ifc-6.3	CRDP retrieval interface	SST CCI	Web form/email for access information, FTP pull from CEMS	PSD [AD 5] SST CCI Level 2, Level 3, Level 4 products

The main internal interfaces of the prototype are those for data exchange between the SST CCI prototype subsystems at ECDF, CMS and MetOffice. Table 3-2 lists them with references to the description of the exchanged data items.

Table 3-2: Internal interfaces of the SST CCI prototype

Ifc ID	Interface Name	Source	Location/Protocol	Interface item description
Ifc-7	**CMS to ECDF intermediate data interface** AVHRR MetOp work files	CMS	FTP push to ECDF	IODD [AD 7], section 3.3, 3.4
Ifc-8	**ECDF to MetOffice intermediate data interface** SST CCI Level 2 and Level 3 products	ECDF	rsync/sftp pull	PSD [AD 5]

Data from CMS were generated in real-time and are routinely downloaded from the CMS ftp server (meteo-spatiale.fr) to ECDF. The ECDF NAS storage is externally accessible using secure file transfer protocols (SFTP). Data transfer to/from the Met Office is achieved by running *rsync* on the Met Office servers which synchronises the archives on the two systems.

3.7 The SST processing chain

An overview of the SST CCI processing chain is illustrated in Figure 3-2. The figure illustrates the two variants of the processing chain: the demonstrator (also referred to as short-term) and the long-term SST processing systems. Both variants share several common components.

The processing occurs at different locations: ECDF carries out the Level-2 and Level-3 processing of NOAA AVHRR GAC, (A)ATSR and passive microwave (PMW) satellite data, CMS produces Level-3 SST from MetOp AVHRR and SEVIRI satellite data, and MetOffice produces Level-4 SST from Level-2 and Level-3 data produced by ECDF.

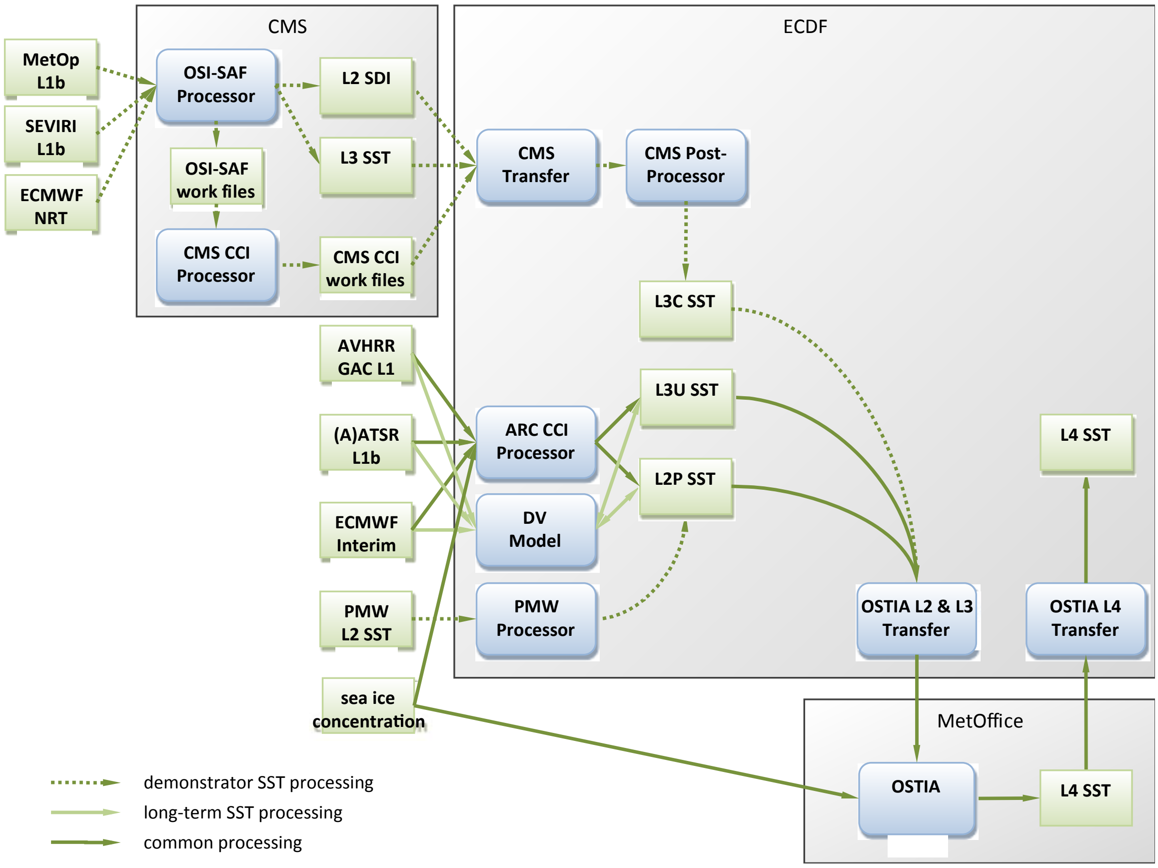

Figure 3-2: SST CCI processing chain

3.8 Distributed subsystems of the SST prototype

Subsystems of the prototype are distributed to infrastructures of CMS, UoE ECDF, and UK MetOffice.

3.8.1 CMS

The processing at CMS consists of the OSI-SAF and CMS CCI processors. MetOp AVHRR and SEVIRI L1b satellite data are the main input of the OSI-SAF processor, with ECMWF NRT auxiliary data. The OSI-SAF processor produces Level-2 SDI and Level-3 SST. The CMS CCI processor interfaces with the OSI-SAF processor and produces Level-3 MetOp work files. All inputs required for the processing are available at CMS.

3.8.2 ECDF

The processing at ECDF mainly consists of the ARC CCI processor, the diurnal variability (DV) model, and the PMW processor. AVHRR GAC Level-1 and (A)ATSR Level 1b satellite data are the main input of the ARC CCI processor, with ECMWF Interim and sea ice concentration auxiliary data. The outputs of the ARC CCI processor are SST CCI L2P and L3U SST. The DV model uses the same inputs as the ARC CCI processor in order to modify the L2P produced by the ARC CCI. It also produces L3U SST. The PMW processor converts PMW Level-2 SST data into SST CCI L2P SST format. The L3C processor produces SST CCI L3C SST from SST CCI L2P and L3U SST. All inputs required for the processing are hosted in high performance fibre channel disk storage at ECDF.

Minor components in the ECDF processing for converting outputs fetched from CMS into SST CCI L3U SST and for the transfer of inputs to and outputs from MetOffice.

3.8.3 MetOffice

The processing at MetOffice essentially consists of the OSTIA processor, which receives SST CCI L2P and L3U SST from ECDF and produces SST CCI L4 SST. Sea ice concentration auxiliary data are used.

4. MULTI-SENSOR MATCH-UP SYSTEM (MMS)

This section describes the MMS prototype implemented in SST CCI phase 1 to generate one complete multi-sensor matchup dataset of about 6 million matchups for the years 1991 to 2010. For the SST CCI round robin exercise the MMD has been split into training dataset, input dataset, validation dataset, and the MMS further has generated a processed dataset with ARC CCI outputs.

4.1 MMS Overview

Main concepts of the MMS are multi-sensor matchups, their determination, and their use in algorithm improvement.

Multi-sensor match-up

Traditional SST retrieval algorithm development and validation has relied on single-sensor match-up datasets (MD), where the SST retrieved from a single satellite instrument is matched to a single type of in-situ measurement.

ESA's SST-CCI project required the development of a multi-sensor match- up system (MMS) for generating a novel multi-sensor match-up dataset (MMD) of temporal and spatial coincidences between multi-sensor L1 and L2 satellite retrievals and time series of SST from *in-situ* sensors. From 20 years of inputs from ATSR-1, ATSR-2, AATSR, NOAA AVHRR GAC, METOP AVHRR, SEVIRI, AMSR-E, TMI, GOME-2, OSI SAF, and time series of *in-situ* measurements recorded by buoys, drifters, moorings and ships the MMS produced a multi-sensor match-up dataset (MMD), which has several critical applications in SST-CCI.

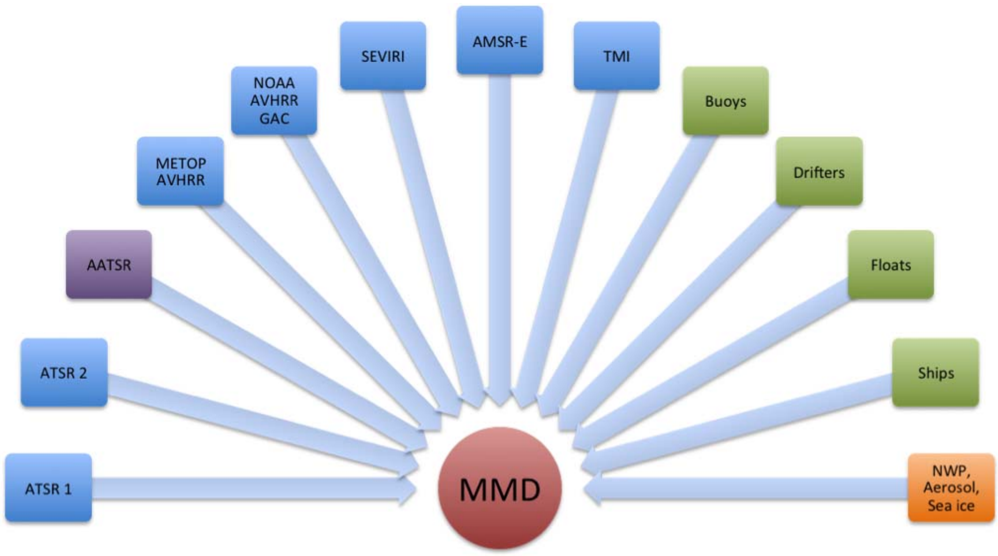

Figure 4-1: Satellite, ancillary and in-situ data used in the MMS

How to build a multi-sensor match-up?

Building a multi-sensor match-up record is quite straightforward. The necessary steps are illustrated in Figure 4-2 and explained below.

1. A single-sensor (A)ATSR MD match-up defines the reference location and time. The MD match-up includes a single (A)ATSR pixel
2. A sub-scene of a METOP AVHRR image is extracted from a single-sensor METOP MD record. The sub-scene overlaps with the reference location and the acquisition has been made within a certain interval of the reference time
3. A sub-scene of a SEVIRI image is extracted from a matching single-sensor SEVIRI MD record
4. A large sub-scene of 101 pixels square is extracted from the AATSR L1b product that corresponds to the AATSR MD match-up.

5. Sub-scenes of 25 pixels square are extracted from matching AVHRR GAC products obtained from all NOAA AVHRR satellites
6. Extracted sub-scene images are added to the MMD record. Images are cropped and filled with no-data pixels, if necessary, so that the reference location is always at the centre of the image

Note that METOP and SEVIRI MD files have been used only because the corresponding satellite data products have not been available to the project. In addition to the satellite data shown in Figure 4-2, passive microwave data from TMI and AMSR-E are extracted as well. NWP data are interpolated onto the sub-scene pixels and added to the record, as well as *in-situ* data. Additional rules are applied to remove duplicates. The content of the MMD files is described in [AD 6].

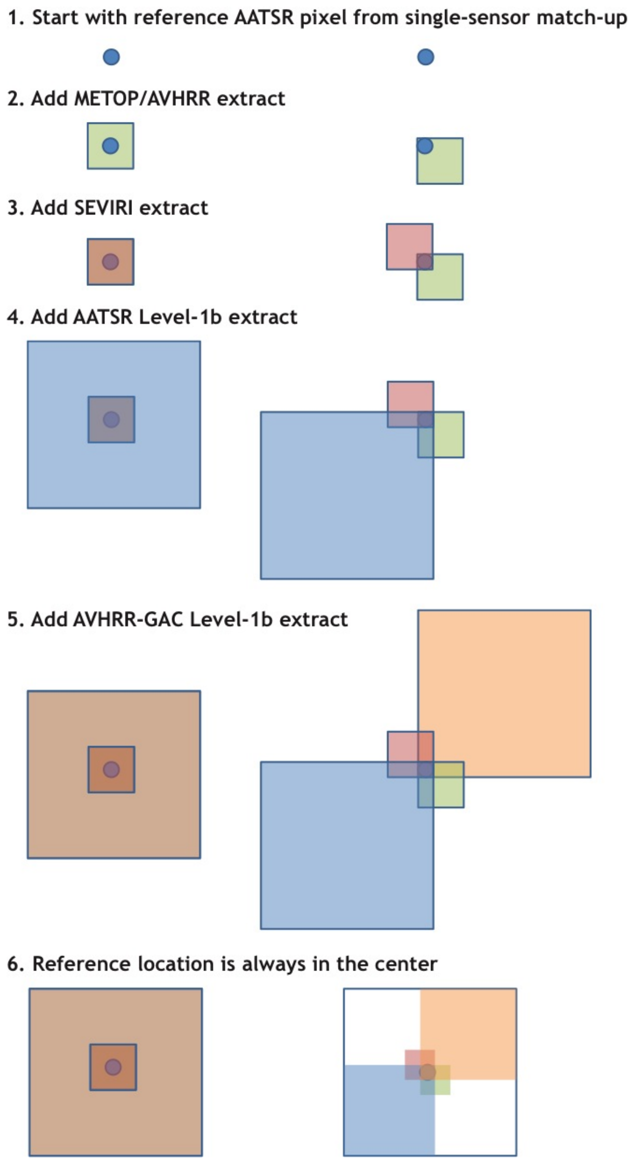

Figure 4-2: How to build a multi-sensor match-up? The best case of perfect image extract alignment (left lane) and the worst case are illustrated

Algorithm improvement cycle

The purpose of the MMS is explicit support of algorithm improvement by means of continuous algorithm development, processing and validation. Figure 4-3 illustrates the activities in the algorithm improvement cycle.

The cycle is initiated by ingesting single-sensor match-up datasets (MD) and single-sensor Earth observation data files into the MMS database, along with auxiliary data files needed for producing L2 SST from single-sensor data (ingestion of input data). Each data file ingested is associated with a unique identifier. The ingested observations are stored in a database table. Information on the columns (i.e. variables) in the data file is stored in a column table. The MMS database can be fed with additional in-situ or Earth observation data at any time.

In the matching activity coincidences between the reference observations from MD files and Earth observation data from different sensors (including in-situ data) are found. The coincidences found are stored in a database table.

In the MMD extraction activity the actual Earth observation sub-scene data are extracted for specified match-up records and columns. Extracted data are written to an MMD file [AD 6].

In the processing activity the SST CCI prototype processor is executed to produce L2 SST from the Earth observation data extracted into MMD files. The L2 SST processing results are written to result MMD files [AD 6].

Result MMD files are then ingested into the MMS database (ingestion of results MMD). New columns of the result MMD files are added to the table of columns in the database.

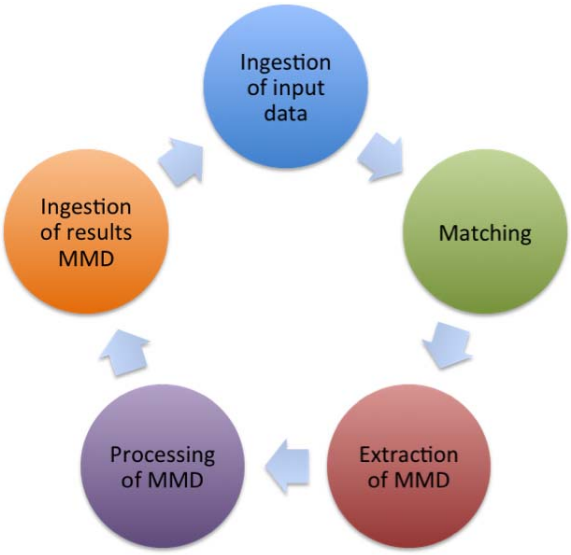

Figure 4-3: Activities in the algorithm improvement cycle

4.2 MMS Infrastructure

The MMS prototype runs on a dedicated virtual machine, which has access to the Edinburgh Compute and Data Facility (ECDF) cluster running a Sun Grid Engine (SGE). The grid engine is employed for extracting image sub-scenes and for retrieving SST by means of the modified ATSR Reprocessing for Climate (ARC) software, which constitutes the SST CCI prototype. Data ingestion and match-up calculations are carried out on the virtual machine. The MMS uses the SST CCI prototype infrastructure and file hierarchy for the input data.

The database implementation is based on the open-source relational database management system PostgreSQL with PostGIS extension. BEAM and a Java persistence API binding have been used to implement satellite and in-situ data readers and transformations of input data in order to harmonise the heterogeneous units and data formats in the inputs. The transformation rules can be fully can be fully customised.

4.3 Data structures and organisation

While the input data mainly remains stored in files in their original format, the reference information and matchup identification is stored in a database.. Database tables define and relate the information on ingested data files and observations and computed match-ups.

On a technical level the Apache OpenJPA Java persistence API binding is used to bind the database tables to Java objects: each database table is represented by a Java class, and each record of a database table is represented by an instance of this class. The OpenJPA API facilitates direct interaction with the database by means of specific Java API and forwarding native SQL commands. The database tables are described further in the paragraphs below, the corresponding Java classes and there relations are illustrated in Figure 4-4.

The **Matchup** table represents match-ups of reference observations with one or many coinciding non-reference observations. Besides the reference observation, a match-up record contains a list of coincidences, a binary pattern, and a flag for marking the match-up as invalid. The pattern is used for quickly finding out from which sensors the observations in the coincidence list are originating.

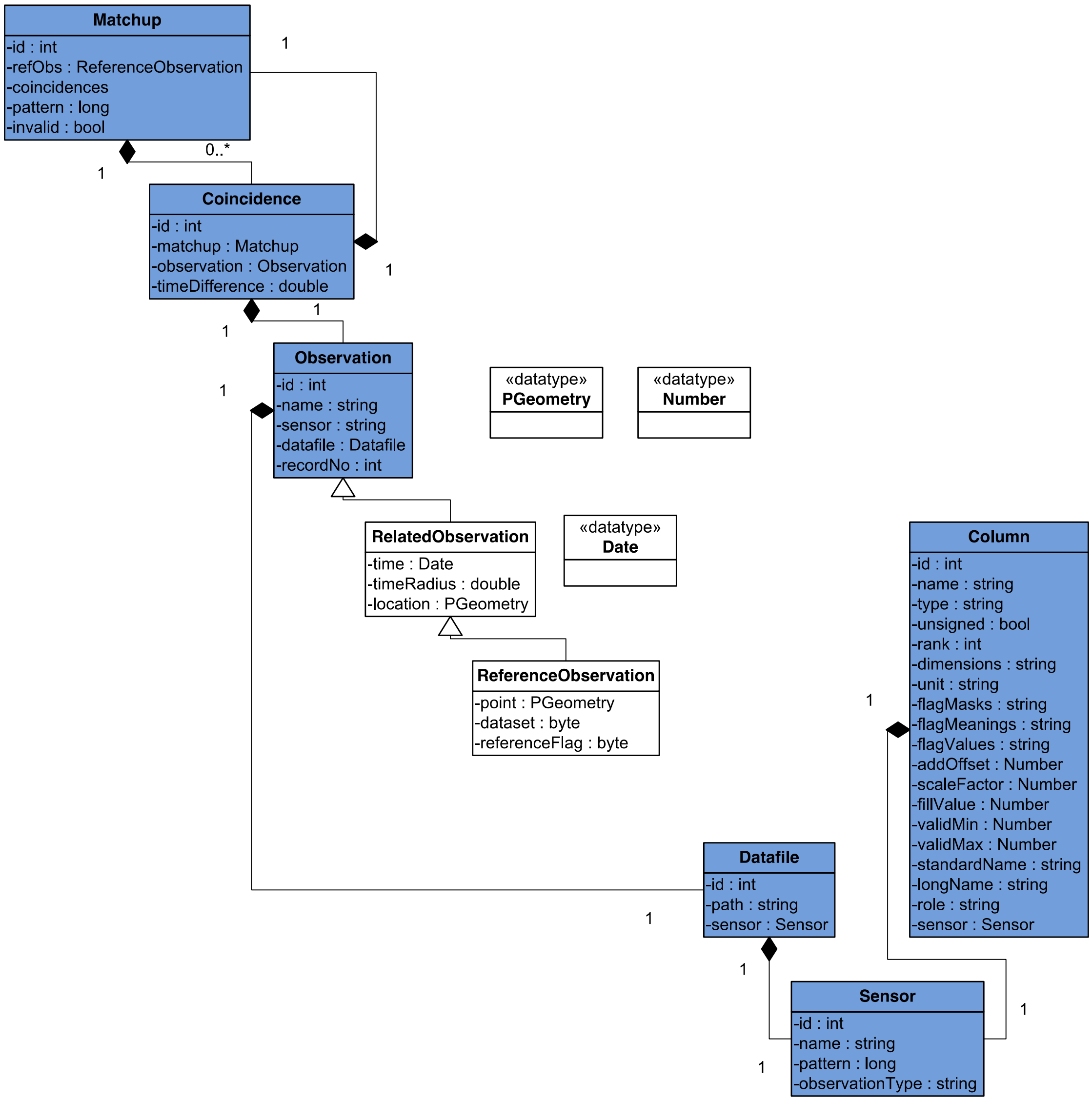

Figure 4-4: Database tables and their representation in Java

A record in the **Coincidence** table relates a match-up record to a single coinciding non-reference observation. Besides the non-reference observation and the match-up record, a coincidence has a field for storing the time difference between the reference observation of the match-up record and the coinciding non-reference observation.

A record in the **Observation** table represents an observation read from a data file. Besides a reference to the data file, an observation has a record number that identifies the observation within the data file, a field for naming the originating sensor, and an optional field for naming the observation. Special observations are the global observation (exhibits a time stamp), the related observation (exhibits a time stamp, a time interval, and a geographic boundary), and the reference observation (in addition to the related observation exhibits a punctual reference location, and a dataset marker for indicating the type of reference observation, e.g. drifter, mooring, ship, etc.). An index is created for the sensor name. Another index is created for the timestamp.

A record in the **Datafile** table represents a data file containing observations. The record stores the file path and a reference to the originating sensor.

A record in the **Sensor** table exhibits the sensor name, the pattern bit mask used to indicate the origin of coinciding observations in the pattern field of a match-up record, and the observation type (observation, global observation, related observation, etc.).

The records in the **Column** table represent the columns, which are available for the MMD file extraction. The fields in the column records are filled during ingestion, according to the information contained in the input data files. Note that a column entry contains the description of the column with the attribute name and the information how to determine the value in the original input files, but not the data actual values. Data file readers retrieve the column values when needed.

4.4 Ingestion of input data

For ingesting input data files into the MMS database specific readers are used. All readers implement a **Reader** interface, which defines methods for reading observations and extracting subscene data (Figure 4-5). Readers for satellite data files (including Aerosol and sea ice data products) are implemented by using the BEAM product reader API; readers for MD and in-situ data files are implemented on basis of the NetCDF API.

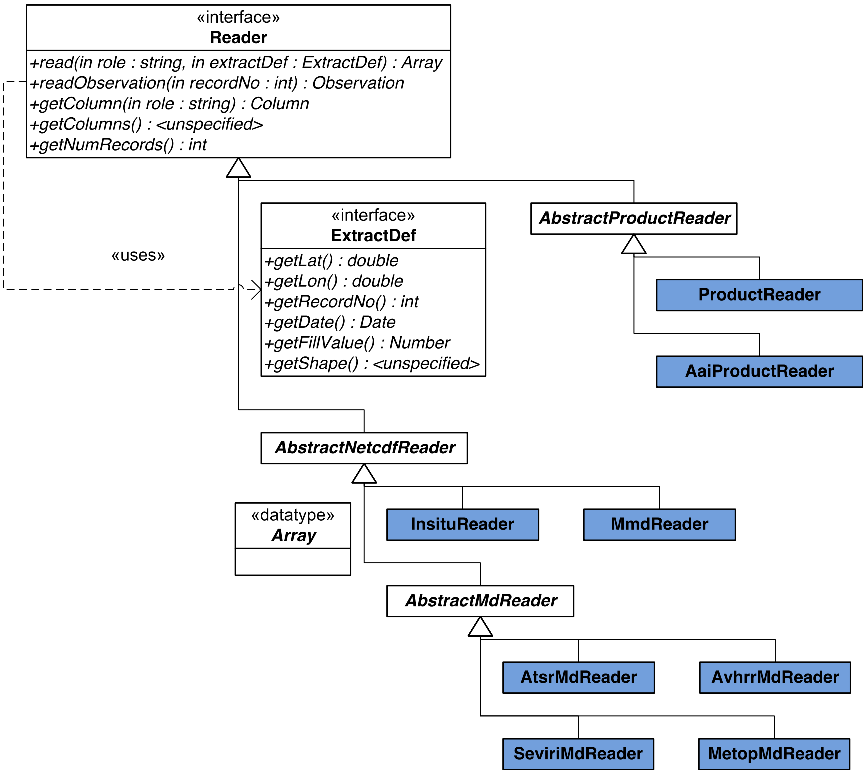

Figure 4-5: Readers for ingesting observations and extracting sub-scene data

For the actual ingestion activity a command line tool is used to automate the overall process. A call of the tool registers input data of a certain time period in the database creating observation entries, data file entries, and sensor and column entries if missing. The time period to be considered, the sensor names and file name patterns for the sensors to be considered along with further sensor properties (reader to be used, value of the pattern bit mask) are configured in a property file supplied as command line argument. An example property file is shown below. In the prototype the ingestion is carried out in monthly chunks after input data staging. The individual single-month ingestion jobs are submitted to the ECDF cluster for concurrent execution.

```
# File: mms-config.properties
#
###############################
# Database settings
###############################

openjpa.ConnectionDriverName = org.postgresql.Driver
openjpa.ConnectionURL        = ...
openjpa.ConnectionUserName   = ...
openjpa.ConnectionPassword   = ...
openjpa.Log                  = DefaultLevel=INFO,SQL=INFO
openjpa.jdbc.SynchronizeMappings = buildSchema

###############################
# Ingestion parameters
###############################

mms.archive.rootdir = /exports/work/...
mms.initialcleanup  = true

mms.source.startTime = 1990-01-01T00:00:00Z
mms.source.endTime   = 2012-01-01T00:00:00Z

mms.source.0.inputDirectory  = atsr_md/v01
mms.source.0.filenamePattern = a[atsr12]{4}_md_.*\\.nc
mms.source.0.sensor          = atsr_md

mms.source.1.inputDirectory  = metop_md/v01
mms.source.1.filenamePattern = mdb1_metop02.*\\.nc\\.gz
mms.source.1.sensor          = metop

mms.source.2.inputDirectory  = seviri_md/v01
mms.source.2.filenamePattern = sstmdb1_meteosat09.*\\.nc\\.gz
mms.source.2.sensor          = seviri

mms.source.3.inputDirectory  = avhrr_md/v01
mms.source.3.filenamePattern = avhrr_md.*\\.nc
mms.source.3.sensor          = avhrr_md

mms.source.11.inputDirectory  = atsr.1/v1
mms.source.11.filenamePattern = AT1.*\\.E1
mms.source.11.sensor          = atsr_orb.1

mms.source.12.inputDirectory  = atsr.2/v1
mms.source.12.filenamePattern = AT2.*\\.E2
mms.source.12.sensor          = atsr_orb.2

mms.source.13.inputDirectory  = atsr.3/v1
mms.source.13.filenamePattern = ATS.*\\.N1
mms.source.13.sensor          = atsr_orb.3

mms.source.21.inputDirectory  = avhrr.m02/v1
mms.source.21.filenamePattern = .*\\.gz
mms.source.21.sensor          = avhrr_orb.m02

mms.source.22.inputDirectory  = avhrr.n10/v1
mms.source.22.filenamePattern = .*\\.gz
mms.source.22.sensor          = avhrr_orb.n10

mms.source.23.inputDirectory  = avhrr.n11/v1
mms.source.23.filenamePattern = .*\\.gz
```

```
mms.source.23.sensor           = avhrr_orb.n11

mms.source.24.inputDirectory  = avhrr.n12/v1
mms.source.24.filenamePattern = .*\\.gz
mms.source.24.sensor          = avhrr_orb.n12

mms.source.25.inputDirectory  = avhrr.n14/v1
mms.source.25.filenamePattern = .*\\.gz
mms.source.25.sensor          = avhrr_orb.n14

mms.source.26.inputDirectory  = avhrr.n15/v1
mms.source.26.filenamePattern = .*\\.gz
mms.source.26.sensor          = avhrr_orb.n15

mms.source.27.inputDirectory  = avhrr.n16/v1
mms.source.27.filenamePattern = .*\\.gz
mms.source.27.sensor          = avhrr_orb.n16

mms.source.28.inputDirectory  = avhrr.n17/v1
mms.source.28.filenamePattern = .*\\.gz
mms.source.28.sensor          = avhrr_orb.n17

mms.source.29.inputDirectory  = avhrr.n18/v1
mms.source.29.filenamePattern = .*\\.gz
mms.source.29.sensor          = avhrr_orb.n18

mms.source.30.inputDirectory  = avhrr.n19/v1
mms.source.30.filenamePattern = .*\\.gz
mms.source.30.sensor          = avhrr_orb.n19

mms.source.41.inputDirectory  = amsr-e/v05
mms.source.41.filenamePattern = [0-9]*-AMSRE-REMSS-L2P-amsr_.*\\.nc
mms.source.41.sensor          = amsre

mms.source.42.inputDirectory  = tmi/v04
mms.source.42.filenamePattern = [0-9]*-TMI-REMSS-L2P-tmi_.*\\.nc
mms.source.42.sensor          = tmi

mms.source.43.inputDirectory  = aerosol-aai/v01
mms.source.43.filenamePattern = aai_.*\\.nc
mms.source.43.sensor          = aai

mms.source.44.inputDirectory  = sea-ice/v01
mms.source.44.filenamePattern = ice_conc_[ns]h_.*
mms.source.44.sensor          = seaice

mms.source.45.inputDirectory  = insitu-history/v01
mms.source.45.filenamePattern = insitu_WMOID.*\\.nc
mms.source.45.sensor          = history

###############################
# Sensor properties
###############################

mms.reader.atsr_md           = AtsrMdReader
mms.reader.metop             = MetopReader
mms.reader.seviri            = SeviriReader
mms.reader.avhrr_md          = AvhrrMdReader
mms.reader.aai               = AaiProductReader
mms.reader.history           = InsituReader
mms.reader.avhrr.n10         = MmdReader
mms.reader.avhrr.n11         = MmdReader
mms.reader.avhrr.n12         = MmdReader
mms.reader.avhrr.n14         = MmdReader
mms.reader.avhrr.n15         = MmdReader
```

```
mms.reader.avhrr.n16          = MmdReader
mms.reader.avhrr.n17          = MmdReader
mms.reader.avhrr.n18          = MmdReader
mms.reader.avhrr.n19          = MmdReader
mms.reader.avhrr.m02          = MmdReader
mms.reader.nwp_atsr.1         = MmdReader
mms.reader.nwp_atsr.2         = MmdReader
mms.reader.nwp_atsr.3         = MmdReader
mms.reader.nwp_avhrr.n10      = MmdReader
mms.reader.nwp_avhrr.n11      = MmdReader
mms.reader.nwp_avhrr.n12      = MmdReader
mms.reader.nwp_avhrr.n14      = MmdReader
mms.reader.nwp_avhrr.n15      = MmdReader
mms.reader.nwp_avhrr.n16      = MmdReader
mms.reader.nwp_avhrr.n17      = MmdReader
mms.reader.nwp_avhrr.n18      = MmdReader
mms.reader.nwp_avhrr.n19      = MmdReader
mms.reader.nwp_avhrr.m02      = MmdReader

mms.observationType.atsr_md  = ReferenceObservation
mms.observationType.metop    = ReferenceObservation
mms.observationType.seviri   = ReferenceObservation
mms.observationType.avhrr_md = ReferenceObservation
mms.observationType.aai      = GlobalObservation
mms.observationType.history  = InsituObservation

mms.pattern.atsr_md           =                1
mms.pattern.metop             =                2
mms.pattern.seviri            =                4
mms.pattern.atsr_orb.1        =                8
mms.pattern.atsr_orb.2        =               10
mms.pattern.atsr_orb.3        =               20
mms.pattern.atsr.1            =                8
mms.pattern.atsr.2            =               10
mms.pattern.atsr.3            =               20
mms.pattern.avhrr_orb.n10     =      80000000000
mms.pattern.avhrr_orb.n11     =     100000000000
mms.pattern.avhrr_orb.n12     =     200000000000
mms.pattern.avhrr_orb.n14     =     800000000000
mms.pattern.avhrr_orb.n15     =    1000000000000
mms.pattern.avhrr_orb.n16     =    2000000000000
mms.pattern.avhrr_orb.n17     =    4000000000000
mms.pattern.avhrr_orb.n18     =    8000000000000
mms.pattern.avhrr_orb.n19     =   10000000000000
mms.pattern.avhrr_orb.m02     =   20000000000000
mms.pattern.avhrr.n10         =              800
mms.pattern.avhrr.n11         =             1000
mms.pattern.avhrr.n12         =             2000
mms.pattern.avhrr.n14         =             8000
mms.pattern.avhrr.n15         =            10000
mms.pattern.avhrr.n16         =            20000
mms.pattern.avhrr.n17         =            40000
mms.pattern.avhrr.n18         =            80000
mms.pattern.avhrr.n19         =           100000
mms.pattern.avhrr.m02         =           200000
mms.pattern.amsre             =           400000
mms.pattern.tmi               =           800000
mms.pattern.avhrr_md          =          1000000
mms.pattern.aai               = 1000000000000000
mms.pattern.seaice            = 2000000000000000
mms.pattern.history           = 4000000000000000
```

4.5 Matching

After ingestion matchups are computed in a separate activity. This can be done using the information in the database without reading input data.

In the prototype, a reference observation is always either an ATSR MD record, a METOP MD record, or a SEVIRI MD record, because single-sensor match-ups are considered as good initial candidates. For the matching of reference observations to non-reference observations the first step is to find coinciding observations from MD files. Pseudo code for this procedure is shown in Figure 4-6.

```
for each atsr_md observation
    search for matching metop observation
    search for matching seviri observation
    search for matching ... observations
    create matchup
for each metop observation not matched above
    search for matching seviri observation
    search for matching ... observations
    create matchup
for each seviri observation not matched above
    search for matching ... observations
    create matchup
```

Figure 4-6: Matching of MD observations

After matching of MD observations, coinciding observations from satellite and auxiliary data files are searched and added to the lists of coincidences in the match-ups. The search for coinciding observations is carried out in chunks of single days. The coincidence time interval is twelve hours. Pseudo code for this procedure is shown in Figure 4-7. A database query for finding coinciding observations is shown in Figure 4-8

```
for each matchup with reference observation timestamp within day
    for each sensor
        search for coinciding sensor observations, order by time difference
        add coincidence with first sensor observation in the list to matchup
```

Figure 4-7: Matching of non-reference observations

```
select obs from  Observation obs,  Observation oref
    where obs.sensor = ’metop’ and oref.id = 87654
    and oref.time - '12:00:00' <= obs.time and obs.time < oref.time + '12:00:00'
    and st_intersects(obs.location, oref.location)
    order by abs(obs.time - oref.time)
```

Figure 4-8: Example database query for finding coinciding observations (example)

For the actual matching activity a command line tool is used for automating the overall process. Parameters (matching period and sensors to be considered) have to be specified in a property file supplied as command line argument. An example property file is shown below (only properties not included in the example for the ingestion activity are listed). The matching is carried out in monthly chunks; individual matching jobs are submitted to the ECDF cluster.

```
# File: mms-config.properties
#
mms.matchup.startTime = 2010-06-01T12:00:00Z
mms.matchup.stopTime = 2010-06-03T12:00:00Z

mms.matchup.cleanup             = false
mms.matchup.cleanupinterval     = false
mms.matchup.markduplicates      = true
```

```
mms.matchup.dropduplicates     = true
mms.matchup.atsr_md            = true
mms.matchup.metop              = true
mms.matchup.avhrr_md           = true

mms.matchup.0.sensor           = atsr_md
mms.matchup.1.sensor           = metop
mms.matchup.2.sensor           = seviri
mms.matchup.3.sensor           = avhrr_md
mms.matchup.11.sensor          = atsr_orb.1
mms.matchup.12.sensor          = atsr_orb.2
mms.matchup.13.sensor          = atsr_orb.3
mms.matchup.21.sensor          = avhrr_orb.m02
mms.matchup.22.sensor          = avhrr_orb.n10
mms.matchup.23.sensor          = avhrr_orb.n11
mms.matchup.24.sensor          = avhrr_orb.n12
mms.matchup.25.sensor          = avhrr_orb.n14
mms.matchup.26.sensor          = avhrr_orb.n15
mms.matchup.27.sensor          = avhrr_orb.n16
mms.matchup.28.sensor          = avhrr_orb.n17
mms.matchup.29.sensor          = avhrr_orb.n18
mms.matchup.30.sensor          = avhrr_orb.n19
mms.matchup.41.sensor          = amsre
mms.matchup.42.sensor          = tmi
mms.matchup.43.sensor          = aai
mms.matchup.44.sensor          = seaice
mms.matchup.45.sensor          = history
```

4.6 Extraction of MMD

The extraction of MMD files from the MMS database requires considerable effort of reformatting. This input to output format transformation is one of the computational services of MMD extraction. A hierarchy of reformatting rules has been implemented in order to change column's names, number formats, units, netCDF attributes, etc. that are different in the ingested input and the MMD output (Figure 4-9). The format of the MMD output columns is fully configurable.

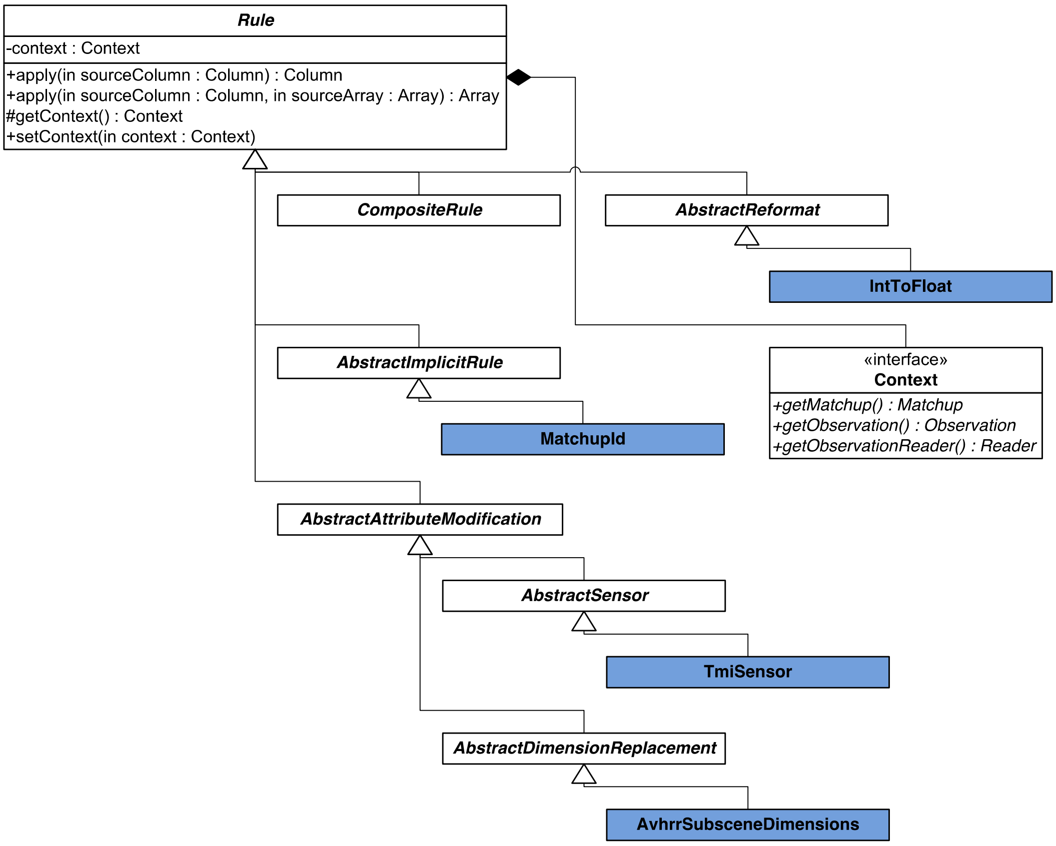

Figure 4-9: Example rules for transforming input columns into MMD columns

For the actual MMD extraction activity a command line tool is used to facilitate automation. Parameters (mainly sub-scene dimensions and input to output transformation rules) have to be specified in a property file supplied as command line argument. A composite of rules is usually applied to a single input variable to get an output variable. The rules are applied to declarations and to values to transform them accordingly. An example property file is shown below (only properties not included in the above examples are listed). Extraction is carried out in monthly chunks; the single-month extraction jobs are submitted to the ECDF cluster.

```
# File: mms-config.properties
#
mms.target.startTime          = 2010-06-02T00:00:00Z
mms.target.stopTime           = 2010-06-03T00:00:00Z
mms.target.filename           = mmd.nc
mms.target.dimensions         = config/mmd-dimensions.properties
mms.target.variables          = config/mmd-columns.config

# File: mmd-dimensions.properties
#
matchup                = unlimited
callsign_length        = 16
filename_length        = 80
matchup.nwp.time       = 25
matchup.nwp.nx         = 1
matchup.nwp.ny         = 1
matchup.nwp.an.time  = 13
matchup.nwp.fc.time  = 25
metop.nx               = 11
metop.ny               = 11
metop.nwp.nx           = 1
metop.nwp.ny           = 1
```

```
metop.nwp.nz         = 60
seviri.nx            = 3
seviri.ny            = 3
atsr.nx              = 11
atsr.ny              = 11
atsr.nwp.nx          = 1
atsr.nwp.ny          = 1
atsr.nwp.nz          = 60
avhrr.nx             = 3
avhrr.ny             = 5
avhrr.nwp.nx         = 1
avhrr.nwp.ny         = 1
avhrr.nwp.nz         = 60
amsre.nx             = 1
amsre.ny             = 1
tmi.nx               = 1
tmi.ny               = 1
seaice.nx            = 1
seaice.ny            = 1
aai.nx               = 1
aai.ny               = 1
insitu.time          = 48

# File: mmd-columns.config
#
# column 1: target name
# column 2: source name
# column 3: rule specification
#
matchup.id          Implicit  MatchupDimension,MatchupId
matchup.time        Implicit  MatchupDimension,TimeType,ReferenceTime
matchup.latitude    Implicit  MatchupDimension,LatType,MatchupLat
matchup.longitude   Implicit  MatchupDimension,LonType,MatchupLon
# ...
atsr.1.time  Implicit  Atsr1Sensor,MatchupDimension,TimeType,ObservationTime
atsr.1.nwp.seaice_fraction           nwp_atsr.1.atsr.1.nwp.CI  ...
atsr.1.nwp.snow_albedo               nwp_atsr.1.atsr.1.nwp.ASN  ...
atsr.1.nwp.sea_surface_temperature   nwp_atsr.1.atsr.1.nwp.SSTK  ...
atsr.1.nwp.total_column_water_vapour nwp_atsr.1.atsr.1.nwp.TCWV  ...
atsr.1.nwp.mean_sea_level_pressure   nwp_atsr.1.atsr.1.nwp.MSL ...
atsr.1.nwp.total_cloud_cover         nwp_atsr.1.atsr.1.nwp.TCC  ...
atsr.1.nwp.10m_east_wind_component   nwp_atsr.1.atsr.1.nwp.U10  ...
atsr.1.nwp.10m_north_wind_component  nwp_atsr.1.atsr.1.nwp.V10  ...
atsr.1.nwp.2m_temperature            nwp_atsr.1.atsr.1.nwp.T2  ...
# ...
# further transformation rules, about 1000 lines of rule specifications in total
```

4.7 SST Processing of MMD

The processing of MMD files is achieved by calling the ARC SST CCI prototype with the MMD file as input. For the actual processing activity there is a command line tool that generates the necessary calls and job submission commands for the ECDF cluster. This script is then copied to a suitable location on the ECDF file system and executed there by means of an *ssh* remote command.

Processing with CDO and ARC-CCI is performed on ECDF using the SGE infrastructure. Jobs are generated and submitted by MMS for concurrent processing, monitored, and results are stored in proper locations in the processing archive. The prototype has processed the complete 20 years matchup dataset.

4.8 Ingestion of results MMD

Ingesting results MMD files into the MMS database is simpler than ingesting Earth observation data files. Since each record in an MMD file exhibits a match-up ID, each MMD record (i.e. observation) is added to the list of coincidences in the corresponding match-up entry stored within the match-up table of the database.

For the actual ingestion of result MMDs activity a command line tool is used. Parameters have to be specified in a property file supplied as command line argument. An example property file is shown below (properties already listed in previous examples are not listed).

```
#
# File: mms-config.properties
#
mms.reingestion.filename = NSS.GHRR.NL.D10152.S1908.E2052.B4996667.GC.MMM.nc
mms.reingestion.sensor = avhrr.n15
mms.reingestion.pattern = 10000
```

5. SST CCI PROTOTYPE CMS SUBSYSTEM

5.1 SST prototype infrastructure

The SST CCI prototype subsystem at CMS is based on two elements: the OSI SAF processor, running in near real-time in the CMS operational environment, and the CMS CCI processor, running in the CMS Research and Development environment. The following diagram provides an overview of the CMS operational environment:

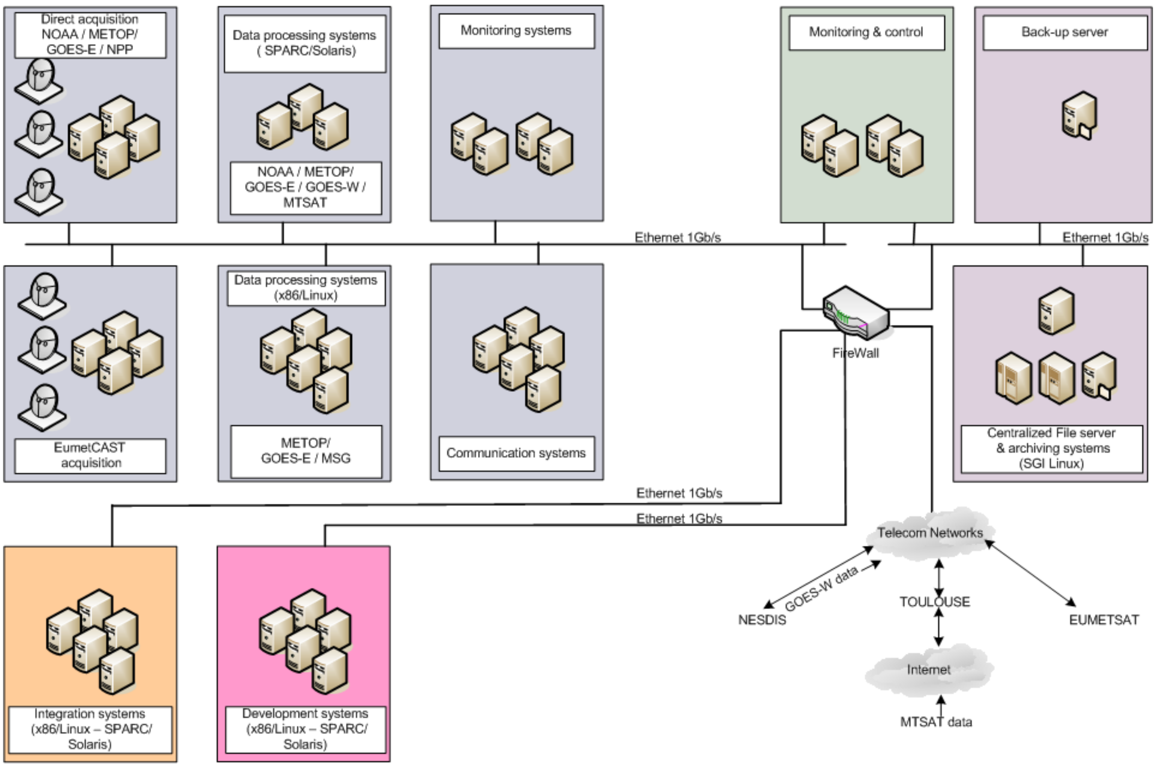

Figure 5-1: Overview of the operational architecture at CMS

The OSI SAF processor includes several operational processing chains, managed by the Archi-PEL software, which are running on x86/Linux servers ("Data processing systems" box in Figure 5-1).

The CMS CCI processor is running on one of the development servers of the CMS Research and Development team: a DELL PowerEdge 6950 server, with 4 Opteron 8216 processors at 2.4 GHz (memory: 16 Gb, disks: 1 Tb), running under Linux RedHat 5.1 operating system.

5.2 External and internal interfaces

The near real-time satellite data used by the OSI SAF processor (MSG/SEVIRI and MetOp/AVHRR Level 1b data) are received from EUMETSAT at CMS through the EUMETCast dissemination system. EUMETCast is a EUMETSAT service, based on telecommunication satellites and Digital Video Broadcast standard. ECMWF near real-time model outputs used by the OSI SAF and the CMS CCI processors are received at Météo-France Toulouse for its operational duties through an operational link (GTS/RMDCN), and are then pushed to CMS though Météo-France private network.

All output products from SST CCI prototype CMS subsystem are pushed to an external FTP server (meteo-spatiale.fr), hosted by a private company contracted by CMS, from where they can be pulled by ECDF).

5.3 Data management

The OSI SAF processor, as part of the CMS operational architecture, relies for data storage on a file server system, including a SGI Linux cluster with 2 nodes, a 24 Tb Raid6 disk bay and a LTO4 tape library with 577 slots and 6 tape drives. The file management software used is DMF.

The output products from the CMS CCI processor are stored on an EMC NAS NS351 server (10 Tb), including SATA Raid5 500 Gb disks, and based on the DART 5.5 software. The files are stored directly (no data base management system) in one directory per month.

5.4 Production control

The OSI SAF processor, as all applications running in near real-time in the CMS operational environment, is managed by in-house software called ArchiPEL. This software is essentially a task scheduler, triggering applications as soon as input data are available in a dedicated spool. But it offers also additional services for supervision, for products dissemination and for the installation of processing chains, which are used by the 24 h/24 operators at CMS to monitor the status of processing chains, and to switch to back-up equipment in case of failure.

The CMS CCI processor is based on a simple *crontab* mechanism, triggering twice per day the production of the CCI AVHRR MetOp internal work files. In case of problems in the processing, an electronic mail is generated, and the production can be re-launched, after the problem has been identified and solved (working days only). Basic quick looks and statistics are produced from every file to verify its content.

5.5 Operational scenarios

The operational software configuration management at CMS is using the RAZOR software management tool. When ready for operational integration, all software updates are first tested on the integration systems, and test processing chains are run in parallel, during a period of time which is depending on the importance and the expected impact of the changes.

In the Research and Development environment, the software configuration is managed under developer's responsibility, using (or not) a software configuration management tool of his choice. In any case, the disk space of all Research and Development team developers is automatically saved on a daily basis, using the TSM (Tivoli Storage Manager) software.

6. SST CCI PROTOTYPE ECDF SUBSYSTEM

6.1 SST prototype infrastructure

The SST CCI prototype uses the Edinburgh Compute and Data Facility (ECDF). The ECDF is a high-performance cluster of servers including around 1500 processors and up to 250 TB of data capacity (increasing over time). The cluster (see Figure 6-1) is connected to a large amount of high performance fibre channel disk storage via a number of dedicated machines. These machines then share the data across the cluster using a parallel file system. The ECDF offers the establishment of a processing grid via the Sun Grid Engine (SGE), which is a suite of software tools supporting Distributed Resource Management (DRM). The SGE software is able to distribute workload across the ECDF resource pool by an efficient scheduling of jobs. It can monitor and manage the jobs, the CPU cores and system memory used, as well as the mounted file systems. Each processing node is equipped with two quad-core CPUs and is specified with two GB of memory per core. Access to ECDF is via a scheduler and batch system.

The ECDF storage is split into two parts: a low cost area for storage of data and easy external access (NAS), and a high-performance area (HPC) for processing data which is accessed by the worker nodes. Both use IBM's General Parallel File System (GPFS), but as the NAS area cannot be directly access from the worker nodes it is necessary to stage (copy) the data from the NAS to HPC. The SST CCI prototype is currently assigned 60 TB of NAS and 3 TB of HPC storage.

The core of the processor is based on the GBCS program [RD 181], which is written in Fortran 95 with some Fortran 2003 extensions, mainly for binding the BEAM Envisat Product Reader (EPR) API. Minor components of the processor are written in C; processing scripts are written in Python, and job submission use shell scripts. Required compilers comprise IFC 12 and GCC 4. Required libraries include netCDF, the radiative transfer model RTTOV-10.2, and the CDO package for pre-processing ECMWF data.

Figure 6-1: ECDF cluster

6.2 External and internal interfaces

For the processing at ECDF the majority of the input data are historical and were copied from various data archives using FTP or tape transfer.

- (A)ATSR data – ftp from the (A)ATSR Multimission archive at NEODC
- AVHRR data – transferred via tape from NOAA
- AMSRE / TMI data – ftp from Remote Sensing Systems
- ECMWF-interim data – ftp from the BADC archive

Data from CMS were generated in real-time and are routinely downloaded from the CMS ftp server (meteo-spatiale.fr) using a simple shell script executed once per day by a *crontab*.

The ECDF NAS storage is externally accessible using secure file transfer protocols: SSHFS, SFTP, or SSH. Data transfer to/from the Met Office is achieved by running the *rsync* program on the Met Office servers which synchronises the archives on the two systems.

Internal data transfer between the NAS (for storage and external access) and HPC (for cluster processing) is performed using the *rsync* program to copy data between the two file systems. The control scripts allow individual sensors to be staged with a granularity of one month. In practice the current 3 TB allocation of HPC storage allows one year of data to be staged for processing at once.

6.3 Data management

External data downloaded to the ECDF cluster are stored in two directory hierarchies: the first is a direct copy of the structure used by the external archive from which the data were copied, the second using a common structure allows all data sources to be accessed in a consistent manner. The two hierarchies reference the same underlying files using hard links (i.e. the same copy of the data is accessed through two directory structures), this permits easy updating of the data files by syncing with external archives while internal access is through a single common structure.

The internal directory structure used is:

{archive root}/{dataset}/v{version}/{YYYY}/{MM}/{DD}/

Data generated for the CCI project are organised following the CCI directory guidelines:

{archive_root}/cciout/{cat}/{level}/{sensor}/{version}/{YYYY}/{MM}/{DD}/

Where:

{cat} is the category, either DM (demonstrator) or LT (long term)

{level} is the processing level one of: L2P, L3U, L3C, L4

{sensor} indicates the sensor providing the data (not applicable to L4)

{version} is the version of the particular dataset

6.4 Production control

The ECDF cluster uses the Sun Grid Engine, which provides standard tools for job submission, monitoring and control. Jobs submitted to the Grid Engine are held in a queue until there is an available resource to execute them. Each job can request a different type of resource, i.e. a simple worker node for standard processing tasks or a special staging node for transferring data between NAS and HPC storage. Array jobs are supported where the same processing is applied to multiple inputs and will execute in parallel when sufficient worker nodes are available. The Grid Engine also supports job dependencies, so tasks will be scheduled after the appropriate dependencies have completed processing. For the SST CCI processing three types of Grid Engine jobs are created:

Staging – these jobs will copy a month of data from the NAS to HPC storage

Processing – these are array jobs which process a month of input data to L2P/L3U outputs

Destaging – these jobs copy the SST CCI output files from HPC to NAS and clean-up the staged input data

All three jobs types are submitted with appropriate resource requirements and dependency information by a top level bash shell script. The production can be initialised for blocks of one month or one year. The larger year long production runs are more efficient as it all three job types can be running in parallel (i.e. March could be staging, while February is processing and January is destaging)

The staging and destaging jobs are both bash shell scripts, which execute the appropriate copy and delete commands to perform the required staging or destaging.

The processing job is a python script, which executes the various steps detailed in the DPM [AD 8]. The python script will process a list of input files. The script executes the appropriate components depending on the input file type, and writes outputs to the appropriate directories. The top-level user interface is via the command line:

ccil2p.py /path/to/input_data

Critical processing errors – i.e. failure to process an orbit due to a corrupted file – will result in the loss of that orbit and subsequent orbits from the same sensor and day. All other sensors and days will be unaffected and continue processing. Critical scheduling errors – i.e. where a task fails to complete – will result in any dependent tasks being held in the queue until the issue is resolved.

Script to stage one month of ECMWF data (stage-ecmwf.sh)

```
#!/bin/bash

# Grid Engine options
#$ -cwd
#$ -l h_rt=06:00:00
#$ -pe staging 1
#$ -j y

. /etc/profile.d/modules.sh
. ~/.bash_profile

extra_days() {
date -d "$1-$2-01 -2 day" +"%Y/%m/%d"
date -d "$1-$2-01 -1 day" +"%Y/%m/%d"
date -d "$1-$2-01 +1 month" +"%Y/%m/%d"
}

year=$1
month=$(printf '%02d' "$((10#$2))")

echo "`date -u +%Y%m%d-%H%M%S` staging of ECMWF $year/$month ..."

if [ "$year" = "" -o "$month" = "" ]; then
    echo "missing parameter, use $0 year month"
    exit 100
fi

if [ "$MMS_ARCHIVE" = "$MMS_NAS" ]; then
    echo "archive and nas are identical: $MMS_ARCHIVE"
    exit 100
fi

cd $MMS_NAS

if [ -e ecmwf-era-interim/v01/$year/$month ]
then
    mkdir -p $MMS_ARCHIVE/ecmwf-era-interim/v01/$year/$month
    rsync -av ecmwf-era-interim/v01/$year/$month/ $MMS_ARCHIVE/ecmwf-era-
interim/v01/$year/$month
    if [ "$?" -ne 0 ]; then exit 100; fi
else
    echo "`date -u +%Y%m%d-%H%M%S` skipping non-existing ecmwf-era-
interim/v01/$year/$month"
fi

for extra in `extra_days $year $month`
do
    mkdir -p $MMS_ARCHIVE/ecmwf-era-interim/v01/$extra
    rsync -av ecmwf-era-interim/v01/$extra/ $MMS_ARCHIVE/ecmwf-era-
interim/v01/$extra
done

echo "`date -u +%Y%m%d-%H%M%S` staging of $year/$m
```

Script to stage one month of input satellite data (stage-input.sh)

```
#!/bin/bash

# Grid Engine options
#$ -cwd
#$ -l h_rt=24:00:00
#$ -pe staging 1
#$ -j y

. /etc/profile.d/modules.sh
. ~/.bash_profile

year=$1
month=$(printf '%02d' "$((10#$2))")
sensor=$3

echo "`date -u +%Y%m%d-%H%M%S` staging of input $sensor $year/$month ..."

if [ "$year" = "" -o "$month" = "" ]; then
    echo "missing parameter, use $0 year month"
    exit 100
fi

if [ "$MMS_ARCHIVE" = "$MMS_NAS" ]; then
    echo "archive and nas are identical: $MMS_ARCHIVE"
    exit 100
fi

cd $MMS_NAS

if [ -e $sensor/v1/$year/$month ]
then
    echo "`date -u +%Y%m%d-%H%M%S` staging $sensor $year/$month ..."
    mkdir -p $MMS_ARCHIVE/$sensor/v1/$year
    rsync -av $sensor/v1/$year/$month $MMS_ARCHIVE/$sensor/v1/$year
    if [ "$?" -ne 0 ]; then exit 100; fi

    if [[ "$sensor" == avhrr* ]]
    then
    clavrx=${sensor/avhrr/clavrx}
    echo "`date -u +%Y%m%d-%H%M%S` staging $clavrx $year/$month ..."
    mkdir -p $MMS_ARCHIVE/$clavrx/v1/$year
    rsync -av $clavrx/v1/$year/$month $MMS_ARCHIVE/$clavrx/v1/$year
    if [ "$?" -ne 0 ]; then exit 100; fi
    fi
else
    echo "`date -u +%Y%m%d-%H%M%S` file not found"
fi


echo "`date -u +%Y%m%d-%H%M%S` staging of $sensor $year/$month ... done"
```

Script to process one month of a satellite (process_month.sh)

```
#!/bin/bash

. eddie_funcs.sh

year=$1
month=$(printf '%02d' "$((10#$2))")

scr=`pwd`
cd logs

sensors=`sensor_list $year`
```

```
pnwp=`mysub $scr/stage-ecmwf.sh $year $month`

for sensor in $sensors
do
    name=`short_name $sensor`-${year: -2:2}$month
    prep=`mysub -N s-$name $scr/stage-input.sh $year $month $sensor`
    proc=`mysub -hold_jid $pnwp,$prep -N p-$name $scr/ccil2p.sh $year $month
$sensor`
    dest=`mysub -hold_jid $proc -N d-$name $scr/destage.sh $year $month
$sensor`
done
```

Script to process one day of a satellite (ccil2p.sh)

```
#!/bin/bash

# Grid Engine options
#$ -t 1-31
#$ -cwd
#$ -l h_rt=06:00:00,sages_1ppn=1
#$ -j y

. /etc/profile.d/modules.sh
. ~/.bash_profile

year=$1
month=$2
sensor=$3
day=$(printf '%02d' "$SGE_TASK_ID")

echo "`date -u +%Y%m%d-%H%M%S` ccil2p $year/$month/$day $sensor ..."

if [ "$year" = "" -o "$month" = "" -o "$day" = "" -o "$sensor" = "" ]; then
    echo "missing parameter, use $0 year month sensor"
    exit 100
fi

path=$MMS_ARCHIVE/$sensor/v1/$year/$month/$day

export CCI_ECMWF_PATH=/exports/work/geos_gc_sst_cci/stagingarea/ecmwf-era-
interim/v01/{Y}/{m}/{d}
export
CCI_L2P_PATH=/exports/work/geos_gc_sst_cci/stagingarea/ccioout/lt/l2p/{prod}/
v01/{Y}/{m}/{d}
export
CCI_L3U_PATH=/exports/work/geos_gc_sst_cci/stagingarea/ccioout/lt/l3u/{prod}/
v01/{Y}/{m}/{d}

export UDUNITS_PATH=
ulimit -s unlimited

$CCI_HOME/python/ccil2p.py $path/*
if [ "$?" -ne 0 ]; then exit 1; fi

echo "`date -u +%Y%m%d-%H%M%S` ccil2p $year/$month/$day $sensor ... done"
```

6.4.1.1 Script to copy output data back to NAS and clean-up (destage.sh)

```
#!/bin/bash

# Grid Engine options
#$ -cwd
#$ -l h_rt=00:30:00
#$ -pe staging 1
#$ -j y
```

```
. /etc/profile.d/modules.sh
. ~/.bash_profile

set -o nounset

ghrsst_prod() { echo $1 | sed '
s/atsr\.1/ATSR1/g
s/atsr\.2/ATSR2/g
s/atsr\.3/AATSR/g
s/avhrr\.n10/AVHRR10_G/g
s/avhrr\.n11/AVHRR11_G/g
s/avhrr\.n12/AVHRR12_G/g
s/avhrr\.n14/AVHRR14_G/g
s/avhrr\.n15/AVHRR15_G/g
s/avhrr\.n16/AVHRR16_G/g
s/avhrr\.n17/AVHRR17_G/g
s/avhrr\.n18/AVHRR18_G/g
s/avhrr\.n19/AVHRR19_G/g
s/avhrr\.m02/AVHRRMTA_G/g'
}

year=$1
month=$(printf '%02d' "$((10#$2))")
sensor=$3

echo "`date -u +%Y%m%d-%H%M%S` de-staging $year/$month $sensor ..."

if [ "$year" = "" -o "$month" = "" -o "$sensor" = "" ]; then
    echo "missing parameter, use $0 year month sensor"
    exit 100
fi

if [ "$MMS_ARCHIVE" = "$MMS_NAS" ]; then
    echo "archive and nas are identical: $MMS_ARCHIVE"
    exit 100
fi

product=`ghrsst_prod $sensor`

cd $MMS_ARCHIVE
for root in cciout/*/*
do
    path=$root/$product/v01/$year/$month
    if [ -e $path ]
    then
    echo "`date -u +%Y%m%d-%H%M%S` export $path ..."
    mkdir -p $MMS_NAS/$path
    rsync -av $MMS_ARCHIVE/$path/ $MMS_NAS/$path/
    if [ "$?" -eq 0 ]
    then
        echo "Clean: $MMS_ARCHIVE/$path"
        rm -rf $MMS_ARCHIVE/$path
        echo "Clean: $MMS_ARCHIVE/$sensor/v1/$year/$month"
        rm -rf $MMS_ARCHIVE/$sensor/v1/$year/$month
        if [[ "$sensor" == avhrr* ]]
        then
        clavrx=${sensor/avhrr/clavrx}
        echo "Clean: $MMS_ARCHIVE/$clavrx/v1/$year/$month"
        rm -rf $MMS_ARCHIVE/$clavrx/v1/$year/$month
        fi
    else
        echo "`date -u +%Y%m%d-%H%M%S` error destaging data ..."
        exit 100
    fi
    else
    echo "`date -u +%Y%m%d-%H%M%S` skipping $path ..."
```

```
    fi
done

echo "`date -u +%Y%m%d-%H%M%S` de-staging of $year/$month ... done"
```

6.5 Operational scenarios

There are two operational scenarios for the prototype processor. The long term ECV that covers the period August 1991 through December 2010 uses data from ATSR and AVHRR instruments. Long-term (LT) processing is the most resource intensive and requires almost 24 hours to process the standard one-year batch of input data. For systematic processing the yearlong batches are submitted daily after a manual check of the previous days processing logs. When a more rapid development cycle is needed jobs are submitted in three or six-month batches for overnight processing.

The Demonstration ECV comprises two three-month periods. The first covering June to August 2007 includes AATSR, AMSR-E and TMI data; the second covering January to March 2012 includes AATSR, MetOp (processed at CMS), and SEVIRI (processed at CMS) data. The diurnal model (DM) processing is less resource intensive as only the AATSR data needs to be processed from Level 1 and is submitted to the Grid Engine as two three-months batch jobs.

Script to submit processing job for one year of data (process_year.sh)

```
#!/bin/bash

. eddie_funcs.sh

year=$1

scr=`pwd`
cd logs

sensors=`sensor_list $year`

deps=""

for month in {01..12}
do
    pnwp=`mysub $deps $scr/stage-ecmwf.sh $year $month`
    next_deps="-hold_jid $pnwp"
    for sensor in $sensors
    do
    name=`short_name $sensor`-${year: -2:2}$month
    prep=`mysub $deps -N s-$name $scr/stage-input.sh $year $month $sensor`
    proc=`mysub -hold_jid $pnwp,$prep -N p-$name $scr/ccil2p.sh $year $month
$sensor`
    dest=`mysub -hold_jid $proc -N d-$name $scr/destage.sh $year $month
$sensor`
    next_deps="$next_deps,$prep"
    done
    deps="$next_deps"
done
```

Subroutine to submit a single processing job (eddie_funcs.sh)

```
mysub() { echo `qsub $@` | sed 's/[^0-9]*\([0-9]*\).*/\1/'; }

short_name() { echo $1 | sed '
s/atsr\.1/at1/g
s/atsr\.2/at2/g
s/atsr\.3/ats/g
s/avhrr\.//g'
}

sensor_list() {
```

```
case "$1" in
    199[1234]) echo 'atsr.1 avhrr.n12' ;;
    199[56])   echo 'atsr.1 atsr.2 avhrr.n12 avhrr.n14' ;;
    1997)      echo 'atsr.2 avhrr.n12 avhrr.n14' ;;
    1998)      echo 'atsr.2 avhrr.n12 avhrr.n14 avhrr.n15' ;;
    1999)      echo 'atsr.2 avhrr.n14 avhrr.n15' ;;
    2000)      echo 'atsr.2 avhrr.n14 avhrr.n15' ;;
    2001)      echo 'atsr.2 avhrr.n15 avhrr.n16' ;;
    2002)      echo 'atsr.2 avhrr.n15 avhrr.n16 avhrr.n17' ;;
    2003)      echo 'atsr.3 avhrr.n16 avhrr.n17' ;;
    2004)      echo 'atsr.3 avhrr.n16 avhrr.n17' ;;
    2005)      echo 'atsr.3 avhrr.n16 avhrr.n17 avhrr.n18' ;;
    *)         echo 'atsr.3 avhrr.n17 avhrr.n18 avhrr.m02' ;;
esac
}
```

7. SST CCI PROTOTYPE METOFFICE SUBSYSTEM

OSTIA is a near-real time system for producing a daily L4 analysis using various input data sources. For the SST CCI prototype, the OSTIA reanalysis system is used, which is scientifically very similar to the near-real time system. However, the systems are run in different ways: the operational one is run routinely every day whereas the reanalysis system is run on a one-off basis producing a long time series of daily fields. Figure 5-1 shows an overview of the system used in the CCI SST prototype.

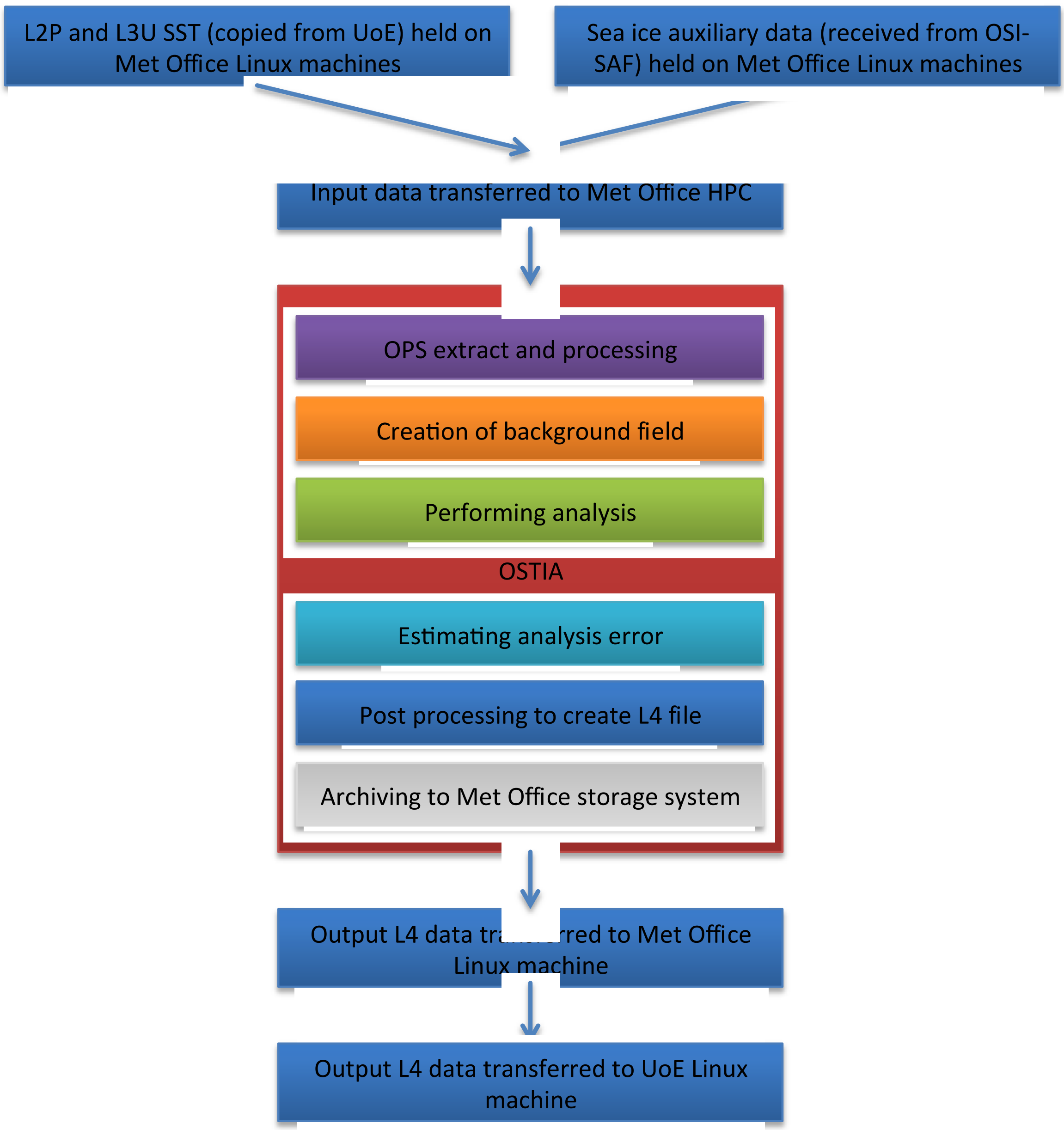

Figure 7-1: Overview of the OSTIA reanalysis system

In order to run the OSTIA reanalysis system efficiently, it is run in a 'leap-frog' fashion in which the 'Observation Processing System extract and processing' stage is run in parallel with the analysis and error estimation procedures. This means the observations are pre-processed a number of days ahead of the SST analysis day.

7.1 SST prototype infrastructure

The OSTIA processor is run on the High Performance Computing (HPC) facility at the Met Office. The HPC consists of approximately 1200 IBM Power 7 nodes, each with 32 CPUs. The nodes are spread evenly between two computing halls in physically separate locations.

All jobs on the Met Office HPC are managed using the Load Leveller batch subsystem. The same computing facility is available for research and development experiments, pre-operational testing, production of reanalyses, and routine operational jobs. Each type of experiment is allocated its own priority level, with the operational system having dedicated nodes at specific times of the day.

7.2 External and internal interfaces

For phase 1, the input SST data will be copied from the University of Edinburgh to the Met Office over the Super Janet network via a staging server. The auxiliary sea ice data is already held at the Met Office following transfer from OSI-SAF for a previous study.

The OSTIA processor uses the Met Office's Observation Processing System (OPS) to read in the required input SST data for the correct time frame and to process and quality check the input data. This produces a number of observation files in an internal Met Office file format. A background file is also created based on the analysis of the previous day with relaxation to climatology. The observations and background files are then passed to the main analysis system, which performs an objective analysis. The main output of this analysis is the L4 SST product. For phase 1, the output L4 files will be copied back to the University of Edinburgh using the Super Janet network.

7.3 Data management

The Met Office HPC facility has been configured with approximately 1500 TB of global storage space. This storage is split between the two computer halls and is shared out to all nodes in the system using IBM's General Parallel File System (GPFS). Disk space is also available on Linux machines within the Met Office computing halls, and it is possible to copy information to those machines from the HPC (and vice versa). While the OSTIA CCI processing is being carried out, the input daily SST and sea ice data will be stored on the Linux machines and copied to the HPC when required using an automated process.

As part of the OSTIA system, all the files that are required for re-running the system and the output files are automatically archived to the Met Office data archive system (the Managed Archive Storage System-Replacement; MASS-R). MASS-R utilises the IBM HPSS (High Performance Storage System) to provide hierarchical storage management of IBM disk and tape hardware and services for very large storage requirements. It enables high data transfer rates to move large files between storage devices and computers, and an environment that can scale to multiple petabytes. The user interface to HPSS, known as MOOSE, is written and maintained by Met Office developers. The SST CCI input files to the OSTIA reanalysis system will be manually archived to MASS-R.

7.4 Production control

The various steps in the main OSTIA system (as detailed in the DPM [AD 8]) are controlled using the Suite Control System (SCS), a tool developed within the Met Office. The system provides a graphical user interface in order to control and make changes to the various tasks. For example, the user can specify the length of the OSTIA run (from a single day to several years) in the SCS. The SCS also controls the submission of jobs to the HPC using a number of Unix scripts. There is error handling within the SCS, which allows the user to determine whether errors from different subcomponents should be fatal or not to the whole system. The user can easily restart the system if a fatal error has occurred.

Various outputs from the OSTIA reanalysis system is automatically added to an internal Met Office monitoring webpage during the run. This page is checked regularly by the scientists running the analysis and will highlight any problems.

All code and scripts used in OSTIA are configuration managed using the Flexible Configuration Management (FCM) system developed at the Met Office.

7.5 Operational scenarios

For the long-term product, the 20-year period will be split into ten 2-year periods, each run with a 2-month spin up. This allows the 2-year periods to be run in parallel, thus greatly speeding up the generation time. This was shown to work well during the production of the previous OSTIA re-analysis.

For the 3-month demo products and for test processing (typically 1 month), the whole period will be produced in a single run.

The control of which satellite SST data is used in each run is done through name lists and the SCS setup.

As stated in Section 5.4, the OSTIA code is maintained under the FCM system and is therefore easy to develop and update in the system (via the SCS). The development cycle involves updating and reviewing the code, making any necessary changes to name lists and the SCS setup, and then using the SCS to run short tests. These can then be compared to a control run.

The final L4 product will be copied directly back to the EDCF.