IDF Format and Content Specification

Author	OceanDataLab
Version/Revision	1.2
Date of issue	2019, January 21th

Revision Change log

Issue	Date	Type	Change description
0.0	2016 May	Initial draft	
1.0	2016 October 21st	Version 1	
1.1	2016 November 14th	update	Add mandatory global attributes
1.2	2019 January 21th	update	Add details about data serialization and GCP density

Table of contents

1 Introduction

1.1 Purpose and scope

The objective of this technical note is to define how data is described and stored in the IDF format. This format focuses on extracting file contents efficiently and fast enough to be streamed by webservices to client applications for interactive visualization, animation or on-the-fly data processing.

1.2 Applicable documents

The following documents contain requirements and information applicable to this document:

[AD-1]	netCDF Climate and Forecast (CF) Metadata Conventions version 1.6 available from: http://cfconventions.org
[AD-2]	COARDS Conventions available from http://ferret.wrc.noaa.gov/noaa_coop/coop_cdf_profile.html
[AD-3]	[AD-1] UDUNITS-2 package available from http://www.unidata.ucar.edu/software/udunits/udunits-2/udunits2.html
[AD-4]	[AD-2] ISO 8601, The International Standard for the representation of dates and times: http://www.iso.org/iso/date_and_time_format
[AD-5]	[AD-3] Unidata Attribute Conventions for Dataset Discovery (ACDD), available from http://www.unidata.ucar.edu/software/netcdf-java/formats/DataDiscoveryAttConvention.html
[AD-6]	[AD-4] Current version (CF-1.6) of the standard name table can be found at: http://cfconventions.org/standard-names.html
[AD-7]	[AD-5] NASA Global Change Master directory (GCMD) Science Keywords and Associated Directory Keywords, available at: http://gcmd.nasa.gov/Resources/valids/archives/keyword_list.html

1.3 Acronyms and abbreviations

IDF	Intermediate Data Format
GCP	Ground Control Point

2 IDF overview

2.1 Motivations

SEAScope is a 3D visualization application for satellite, in-situ and numerical model data. It offers advanced rendering functionalities that ease the detection of synergies between several sources of observations and simulations. It also features tools to help users design and test algorithms on a large variety of data with immediate visual feedback.

Thanks to the 3D rendering engine, SEAScope does not require any specific projection in its input. Direct access to raw data is however judged inadequate for performance issues as raw data formats are highly heterogeneous. Also, some data may need a time and resource consuming pre-processing (eg. resampling or scientific algorithm) for which it is preferable to store the result in a common format. It leads to the definition of the Intermediate Data Format (IDF): the use of a unique file format as well as the standardization of its data content (ie. geolocation patterns or storage of geophysical variables) allow a better and easier optimization for accessing and streaming data.

The definition of an IDF format is also an opportunity to provide data with a certain degree of self description and user friendliness in case of data distribution or sharing.

During this process, we have paid a special attention for preserving the data geometry since no preliminary projection is required. Keeping a good dynamic range of the geophysical variables in the IDF while streaming data as light as possible was also a main concern to allow subsequent calculations and manipulations.

2.2 NetCDF-4 classic file format

IDF data files use the **NetCDF-4 Classic** format. A major advantage to the use of NetCDF-4 format products from the producer's perspective is that no additional metadata records are required when using this format since one can easily extract it from the files without having to decompress the entire file.

The IDF formatted data sets must comply with the Climate and Forecast (CF) Conventions, v1.6 or later because these conventions provide a practical standard for storing oceanographic data in a robust, easily-preserved for the long-term, and interoperable manner.

2.3 Multiresolution support

Multiresolution support is required for high resolution data in order to stream the appropriate amount of data according to the current viewport zoom in SEAScope. It avoids to load (or resample on the fly) a high amount of data when the viewport is far away from data and it allows nice transition while zooming inside data.

IDF format is naturally supporting multiresolution by allowing to generate multiple independent IDF files from the original data, one at each different spatial sampling. For each IDF file, the associated spatial spacing in meters is indicated in global attributes (see idf_spatial_resolution in 3.2 Global attributes) in order for SEAScope to know which IDF file to use according to the viewport. These multiresolution IDF files are also arranged with an index, called here subsampling index, which goes from 0 (lowest spatial spacing) to n (highest spatial spacing). This index is specified in the IDF filename and as a global attribute (see idf_subsampling_factor in sections 3.1 File naming and 3.2 Global attributes).

Although not mandatory for SEAScope use, the most simple/common case is to apply a downsampling factor of 2 between two successive IDF files in terms of spatial sampling. Then, the associated spatial spacing is each time multiplied by 2 and the subsampling index is equivalent to the exponent of 2 used from IDF file with the lowest spatial spacing (which is not necessarily the raw data spacing if resampling occurs during conversion into IDF). In other words, from an IDF file with a subsampling index of 0, a subsampling index of 1 indicates a downsampling by a factor 2, a subsampling index of 2 indicates a downsampling by a factor 4 and so on.

2.4 Geolocation with GCP

Ground Control Points (GCP) are a set of control points relating coordinates (typically latitude and longitude but also eventually depth or time) and positions (ie. indices) in geophysical variables. In most cases, data geolocation patterns can be sufficiently described with a subset of coordinates. Thus, a correct geographic mapping can be performed in SEAScope without transmitting coordinates for each data point. Subsetting coordinates reduces as well the size of IDF files. GCPs are also intended to be used in the spatio-temporal indexation scheme of SEAScope.

Currently, the concept of GCPs is not widely used: among geospatial tools we can cite GDAL which has defined GCPs in its model and among remote sensing data we can cite Sentinel-1 data format for which the full lat/lon coordinates are not given. As a consequence, no convention has been defined yet regarding the GCPs dimensions and variables naming. This document defines this naming for each conventional datamodel, see section 3.5 Coordinate variables. The main rules are the following:

- ➢ GCP dimensions are named from the main dimensions of the datamodel in use, with the suffix “_gcp”. As an example, for the swath datamodel, the main dimensions are “row” and “cell” then the GCP dimensions (subset of the main dimensions) are “row_gcp” and “cell_gcp”. The suffix “_gcp” allows to discriminate the main dimensions used for the geophysical variables and the GCP dimensions used for the subset of coordinates.
- ➢ GCP coordinate variables are named from the classic coordinates names with the suffix “_gcp” : lon_gcp, lat_gcp, eventually time_gcp or depth_gcp. It seems preferable to add this suffix in order to not mislead softwares or users expecting full coordinates in variables named lon or lat.
- ➢ GCP position variables are named from the main dimensions of the datamodel in use, with the prefix “index_” and the suffix “_gcp”. With the swath datamodel as an example, we obtain “index_row_gcp” and “index_cell_gcp”. These variables provide index location of GCPs in the datamodel dimensions.

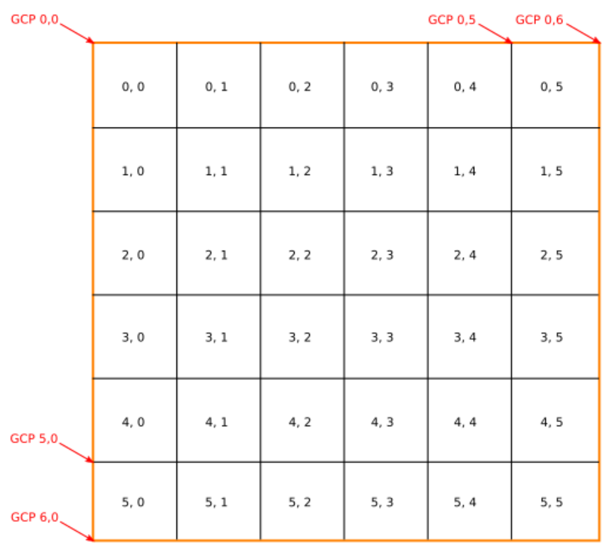
GCP density shall be as minimal as possible to keep the IDF file size small but still to allow interpolation of the intermediate positions at a precision better than the IDF spatial spacing. It depends on data resolution, nature and geometry but also on the purpose of the IDF file (visualization only, analysis, etc…) so it is not possible to provide a formula that computes the GCP density automatically. Here are some hints that can help with the tuning of GCP density:

- If the IDF is only used for visualization, start from a low number of GCPs (a 4x4 grid), check that the granule is displayed at the right location and that it is not distorted by interpolation (use coastline and islands as reference if they are available). If not double GCP density and repeat.
- GCP density must never exceed the resolution of the data (it provides no meaningful information and makes the files heavier). The only exception is for IDF files generated to be visualized in SEAScope as wind barbs or arrows: the symbols are placed above GCPs on the globe, so to increase the barbs/arrows density you have to increase the GCPs density too.
- For trajectories, GCPs must have the same resolution/density as data.

Another important point to note is that GCPs must be placed between data points. SEAScope currently requires this to detect the outer shape of the data.

Placing the GCPs at the center of the data points would generate a distorted rendering for the granule because the application would still display the same amount of data over an area which is actually one half-pixel sorter on each side.

This constraint is not very intuitive and should be removed in future revisions of this document once the application evolves to compute the outer shape of the granules automatically.



3 IDF NetCDF-4 file structure

In the context of the IDF format, a 'granule' is defined as the smallest aggregation of data which is independently managed (i,e., described, inventoried, retrievable). This concept is of importance to choose an adequate granularity for data partitioning.

3.1 File naming

3.1.1 Unique granule in the original file

If one granule is stored in the file to be converted, the IDF file name is following the original name of the file following by the extension `_idf_' and a subsampling index. Keeping the original file name enables users to easily retrieve the original data. The subsampling index is a two digits starting at 00 for the higher resolution (more details can be found in section 2.3 Multiresolution support).

< Original File name > _idf_<Subsampling index>.<File type>

3.1.2 Several granules in one file

When the original file name cannot be used as a unique name (e.g. the file needs to be splitted into several time steps, orbits, platforms ...), the name of the idf is not self explanatory and should be reconstructed following this standard name (file naming convention components are described in Table 3.1):

<Product_string>_<Indicative Date><Indicative Time>_<Processing Level>_v< Product Version>_fv<File Version>_idf_<Subsampling index>.<File Type>

Table 3.1: File naming convention components

Name	Definition	Description
<Indicative Date>	YYYYMMDD	The identifying date for this data set, using the format YYYYMMDD, where YYYY is the four-digit year,MM is the two-digit month from 01 to 12, and DD is the two-digit day of month from 01 to 31. The date used should best represent the observation date for the dataset.
<Indicative Time>	HHMMSS	The identifying time for this data set in UTC, using the format HHMMSS, where HH is the two-digit hour from 00 to 23, MM is the two-digit minute from 00 to 59, and SS is the two-digit second from 00 to 59. The time used is dependent on the <Processing Level> of the data set: L2P: start time of granule L3U: start time of granule L3C and L3S: centre time of the collation window L4: nominal time of analysis All times should be given in UTC and should be chosen to best represent the observation time for this dataset. Note: providers should ensure the applications they use to determine UTC properly account for leap seconds.

<Processing Level>	The data processing level code	The data processing level code (L2P, L3U, L3C, L3S, or L4).
<Product String>	A character string identifying the current product set. The string is used uniquely.	The unique “name” of the product line.
< Product Version>	nn.n	Version number of the product. A different version number must be associated with a product at each processor algorithm or configuration change. For example, “02.0”.
<File Version>	xx.x	Version number for the file, for example, “01.0”.
<Subsampling index>	sub<byte> (ie sub0)	Index indicating the spatial resolution of the IDF file. In most cases, it represents a subsampling of a factor exponent of 2 from the IDF file with the lowest spatial spacing.
<File Type>	netCDF data file suffix (nc)	Indicates this is a NetCDF file containing data.
<Platform>	The identifier of the measuring platform.	The identifier of the acquisition platform: if it as a WMO identifier, it shall be labelled WMO<identifier> otherwise it will use the provider identifier.

For most drifter databases, one file includes data from all platforms and time.

There is one IDF file per platform, instrument and time range to be displayed (month for most drifters, day for drifters at high resolution like those from CARTHE experiment). For platforms having different onboard instruments, all data (from different on board sensor) can be stored in the same file if they have similar resolution and coordinate dimension. If the different instruments have different sampling times and often or different availability time ranges, they are stored in separate IDF files.

If there is a significant interruption time (to be defined, more than one day for instance) or a significant change in the acquisition process (ex: maintenance change on the instrument, or replacement), then the data product for a specific platform/instrument may be split into two or more chronologically consecutive files within the current reference time range.

A file containing the data from one platform (buoy, ship,...) during one specific time range will be named as follow:

<Product String>_<Start Date><Start Time>_<End Date><End Time>_<Platform>_v<Product Version>_fv<File Version>_idf_<Subsampling index>.<File Type>

A file containing the data from a set of similar buoys (surface drifters, ARGO floats ...) during one specific time range will be split so that there is one file per buoy. The naming is similar to the one for fixed station/platform.

For along track L3 products where data are collated daily (several passes are stored in one file), the pass and the cycle is specified during the naming process of the IDF file so that there is a unique granule stored in each file.

<Product String>_<Start Date><Start Time>_<End Date><End Time>_c<cycle>_p<pass>_v< Product Version_-fv<File Version>_idf_<Subsampling index>.<File Type>

3.2 Global attributes

Table 3.2 below summarizes the global attributes that are mandatory or useful for IDF data file. More details on the CF-mandated attributes (as indicated in the Source column) are available at: http://cfconventions.org/Data/cf-conventions/cf-conventions-1.6/build/cf-conventions.html#attribute-appendix and information on the ACDD recommendations is available at http://www.unidata.ucar.edu/software/netcdf-java/formats/DataDiscoveryAttConvention.html. Some attributes (in orange) are required in order to be opened by SEAScope.

The content of this table may change to take into account new IDF data or SEAScope functionalities.

Table 3.2: Mandatory global attributes for IDF data file (attributes in orange are required by SEAScope, attributes in blue can be displayed and thus are of use for SEAScope, attributes in purple can be displayed by SEAScope depending on the type of data, attributes in green are highly recommended for CF compliancy, all the other are optional).

Global Attribute Name	Format	Description	Source
idf_granule_id	string	Text string, unique identifier for the granule in SEAScope. For most cases this attribute is the name of the IDF file without the subsampling index, idf suffix and file extension.	IDF
idf_subsampling_factor	int	Subsampling index integer	IDF
idf_spatial_resolution	float	Geophysical spatial resolution of the variable, used in SEAScope to determine the resolution to use as a function of the zoom. Should be specified in meter.	IDF
idf_spatial_resolution_units	string	Unit for spatial resolution, has to be meter (‘m’)	
time_coverage_start	string	Representative date and time of the start of the granule in the ISO 8601 compliant format of “yyyymmddThhmmss.ssssssZ”. The exact meaning of this attribute depends the type of granule: · L2P: first measurement in granule	ACDD

		(identical to 'time' netCDF variable) · L3U: start time of granule · L3C and L3S: representative start time of first measurement in the collation · L4: representative start time of the analysis (start_time and stop_time together represent the valid period of the L4 granule)	
time_coverage_end	string	Representative date and time of the end of the granule in the ISO 8601 compliant format of "yyyymmddThhmmss.ssssssZ". The exact meaning of this attribute depends the type of granule: · L2P: last measurement in granule · L3U: stop time of granule · L3C and L3S: representative stop time of last measurement in collation · L4: representative stop time of the analysis (start_time and stop_time together represent the valid period of the L4 granule)	ACDD
Conventions	string	A text string identifying the netCDF conventions followed. This attribute should be set to the version of CF used and should also include the ACDD. For example: "CF-1.4, Unidata Observation Dataset v1.0".	CF
title	string	The "title" attribute gives a brief description of the collection. Its use is highly recommended and its value will be used by THREDDS and as the title of the collection visible in SEASCOPE client. It therefore should be human readable and reasonable to display in a list of such names.	CF, ACDD
summary	string	The "summary" attribute gives a longer description of the dataset. Its use is highly recommended. In many discovery systems, the title and the summary will be displayed in the results list from a search. It should therefore capture the essence of the dataset it describes. For instance, we recommend this field include information on the type of data	ACDD

		contained in the dataset, how the data was created (e.g., instrument X; or model X, run Y), the creator of the dataset, the project for which the data was created, the geospatial coverage of the data, and the temporal coverage of the data. This should just be a summary of this information, more detail should be provided in the recommended creator attributes, the recommended geospatial attributes, and the recommended temporal attributes.	
references	string	Published or web-based references that describe the data or methods used to produce it.	CF
institution	string	Institution which the data originally come from. If a dataset is a simple reformatting without any modification the source institution is to be used. See table 4.2for available codes.	CF, ACDD
institution_abbreviation	string	Abbreviation of the above full institution name.	CF, ACDD
history	string	The "history" attribute provides an audit trail for modifications to the original data. It should contain a separate line for each modification with each line including a timestamp, user name, modification name, and modification arguments. Its use is recommended and its value will be used by THREDDS as a history-type documentation. The "history" attribute is recommended by the NetCDF Users Guide and the CF convention.	CF, ACDD
comment	string	Miscellaneous information about the data or methods used to produce it.	CF, ACDD
license	string	Describe any restrictions to data access, use, and distribution. IDF data sets should be freely and openly available to comply with the R/GTS framework, with no restrictions. However, if a user should submit a simple registration via a web form, for example, the URL could be given here. Default to “ IDF protocol describes data use as free and open.”	ACDD

id	string	The unique character string for this product.	ACDD
naming_authority	string	The "id" and "naming_authority" attributes are intended to provide a globally unique identification for each dataset. The "id" value should attempt to uniquely identify the dataset. The naming authority allows a further refinement of the "id". The combination of the two should be globally unique for all time. We recommend using reverse-DNS naming for the naming authority.	ACDD
product_version	string	The product version of this data file	
processing_software	string	Name and version of the processing software	
uuid	string	A Universally Unique Identifier (UUID). Numerous, simple tools can be used to create a UUID, which is inserted as the value of this attribute. See http://en.wikipedia.org/wiki/Universally_Unique_Identifier for more information and tools.	
netcdf_version_id	string	Version of NetCDF libraries used to create this file. For example, “"4.1.1 of Dec 22 2011 16:33:39 $"”	
date_created	string	The date and time the data file was created in the form “yyyymmddThhmmssZ”. This time format is ISO 8601 compliant.	ACDD
date_modified	string	Product last modification date in the form “yyyymmddThhmmssZ”. This time format is ISO 8601 compliant.	ACDD
file_quality_level	integer	A code value: 0 = unknown quality 1 = extremely suspect (frequent problems, e.g. with known satellite problems) 2 = suspect (occasional problems, e.g. after launch) 3 = excellent (no known problems)	
spatial_resolution	string	A string describing the approximate resolution of the product. For example, “1.1km at nadir”	
time_coverage_resolution	string	The resolution provides an idea of the density of the data inside the time range and should	ACDD

		be an ISO8601 duration string (e.g., "P10D").	
geospatial_lat_max	float	Decimal degrees north, range -90 to +90.	ACDD
geospatial_lat_min	float	Decimal degrees north, range -90 to +90.	ACDD
geospatial_lon_max	float	Decimal degrees east, range -180 to +180.	ACDD
geospatial_lon_min	float	Decimal degrees east, range -180 to +180.	ACDD
geospatial_vertical_min		Use the min attribute to describe a simple vertical minimum depth or height. If **geospatial_vertical_units** is not used, vertical is assumed to be in meters above ground. Further refinement of the geospatial bounding box can be provided by using the units and resolution attributes.	ACDD
geospatial_vertical_max		Use the max attribute to describe a simple vertical maximum depth or height. If **geospatial_vertical_units** is not used, vertical is assumed to be in meters above ground. Further refinement of the geospatial bounding box can be provided by using the units and resolution attributes.	ACDD
geospatial_vertical_units	string	meters	ACDD
geospatial_vertical_positive	string	The geospatial_vertical_positive attribute indicates which direction is positive (a value of "up" means that z increases up, like units of height, while a value of "down" means that z increases downward, like units of pressure or depth).	ACDD
nominal_latitude	float	Location latitude of acquisition platform. Only used in case of a fixed station (ex: moored buoy). This may different from the actual GPS location provided in lat/lon variable. Decimal degrees north, range -90 to +90.	
nominal_longitude	float	Location longitude of acquisition platform. Only used in case of a fixed station (ex: moored buoy). This may different from the actual GPS location provided in lat/lon variable. Decimal degrees east, range -180 to +180.	
source	string	Comma separated list of all source data present in this file. List current sources first,	CF

		followed by Auxiliary sources.	
source_version	string	The version of the source data used, as provided by the original producer	
wmo_id	string	WMO identifier, if any. Use the source provider id if no WMO identifier is existing for this platform.	
buoy_network	string	Identifier of the observation network to which the platform belongs to.	
station_name	string	Full name of the station (usually a location)	
station_id	string	Identifier of the station in network (this is the provider id which may be different from the WMO id).	
sea_floor_depth_below_sea_level	float	Positive value in meters.	
site_elevation	float	Positive value in meters (for platforms on land such as lighthouses).	
platform	string	Satellite(s) used to create this data file. Select from the entries found in the Satellite Platform column of table 3.5 and provide as a comma separated list if there is more than one.	
platform_type	string	Type of platform.	
sensor	string	Sensor(s) used to create this data file. Select from the entries found in the Satellite Sensor column of table 3.5 and provide as a comma separated list if there is more than one.	
band	string	Name of the sensing band used. Can be: • a band name for microwave sensors (C, Ku, Ka, L,...) • a wavelength for optical or infra-red sensors • a derived product such as Chl-a, SST, SPM, ...)	
sensor_description	string	Free description of the instrument	
sensor_manufacturer	string	Name of the company responsible for the instrument construction	
sensor_serial_number	string	Serial number of the instrument, following the manufacturer convention.	
sensor_install_date	string	Date and time at which the instrument was	

		installed on the platform (buoy or ship), in the ISO 8601 compliant format of “yyyymmddThhmmssZ”.	
sensor_height	string	Height (positive) or depth (negative) above sea surface at which the instrument is fixed, in meters.	
sensor_sampling_period	string	Sampling period of the instrument, in seconds. Applicable for instance to anemometer and wave sensors.	
sensor_sampling_rate	string	Sampling rate of the instrument, In Hz	
sensor_calibration_date	string	Date and time at which the sensor was last calibrated, in the ISO 8601 compliant format of “yyyymmddThhmmssZ”.	
sensor_history	String list	List of events or interventions on the sensor in the time interval covered by the data file. Specify a date and time in the ISO 8601 compliant format of “yyyymmddThhmmssZ” for each entry in this table. Includes for instance any maintenance operation or malfunction event.	
sensor_type	string	Type of sensor.	
Metadata_Conventions	string	Unidata Dataset Discovery v1.0	ACDD
metadata_link	string	Link to collection metadata record at archive	ACDD
keywords	string	The "keywords" attribute lists key words and phrases that are relevant to the dataset. Its use is highly recommended. The values in the list may be taken from a controlled list of keywords (e.g., the AGU Index list or the GCMD Science Keywords). If a controlled list is used, the "keywords_vocabulary" attribute may be used to identify the list.	ACDD
keywords_vocabulary	string	The "keywords_vocabulary" attribute identifies the controlled list of keywords from which the values in the "keywords" attribute are taken. If you are following a guideline for the words/phrases in your "keywords" attribute, put the name of that guideline here. The use of this attribute is recommended and its value will be used by THREDDS to identify the vocabulary from which the keywords come. “NASA Global Change	ACDD

		Master Directory (GCMD) Science Keywords” as defined in [AD-7] are recommended.	
standard_name_vocabulary	string	“NetCDF Climate and Forecast (CF) Metadata Convention”	ACDD
acknowledgement	string	A place to acknowledge various type of support for the project that produced this data.	ACDD
creator_name	string	Provide a name for the most relevant point of contact at the producing organization or agency.	ACDD
creator_email	string	provide the email relevant to the creator name	ACDD
creator_url	string	provide the url relevant to the creator name and this data set	ACDD
publisher_name	string	Name of the distributing agency if different from institution attribute	ACDD
publisher_url	string	URL of the distributing agency if different from institution url attribute	ACDD
publisher_email	string	Email of help desk or contact point of the distributing agency	ACDD
processing_level	string	GHRSST definitions are the options: L2 L3U, L3C, L3S, L4	ACDD,
cdm_data_type	string	Data feature type (point, station, swath, grid,...) as defined by Unidata CDM model: http://www.unidata.ucar.edu/software/netcdf-java/CDM/	ACDD

The following tables provide the reference codes for some of the above attributes (for which a reference is made to these tables).

Table 3.3: list of currently identified types of satellite sensors

Sensor code	**Description**
altimeter	Onboard Jason-x, Envisat, Cryosat-2,...
sar	Synthetic Aperture Radar: onboard ERS, Envisat, Sentinel-1,...
infrared radiometer	Onboard NOAA-xx, MSG-xx, METOP-x,...
microwave radiometer	AMSR-E, AMSR2, GMI, SMOS,...

Table 3.4: list of currently identified types of observation platforms

Platform code	Description
leo satellite	Low earth orbit satellite, including for instance all polar orbiting satellites: ERS, EnviSAT, METOP, ...
geostationary satellite	Satellite in geostationary orbit:
moored buoy	
drifting buoy	Surface drifters from Coriolis or AOML database
ship	
argo	Float from ARGO program

3.3 Variable attributes

Table 3.5: Variable attributes for IDF data files

Variable Attribute Name	Format	Description	Source
_FillValue	Must be the same as the variable type	A value used to indicate array elements containing no valid data. This value must be of the same type as the storage (packed) type; should be set as the minimum or maximum value for this type. Required for the majority of variables except coordinates, gcps, mask and flags.	CF
units	string	Text description of the units, preferably S.I., and must be compatible with the Unidata UDUNITS-2 package. For a given variable (e.g. wind speed), these must be the same for each dataset. Required for the majority of variables except mask, quality_level, and l2p_flags.	CF, ACDD
scale_factor	float	multiplication factor to get real data value in units.	
add_offset	float	number to be added to the data to get real data value in units. If both scale_factor and add_offset attributes are present, the data	

		are first scaled before the offset is added.	
long_name	string	A free-text descriptive variable name.	CF, ACDD
valid_min	Expressed in same data type as variable	Minimum valid value for this variable (in storage type). The fill value should be outside this valid range. Values outside of **valid_min** and **valid_max** will be treated as missing values. Required for all variables except variable time.	CF
valid_max	Expressed in same data type as variable	Maximum valid value for this variable (in storage type). The fill value should be outside this valid range. Required for all variables except variable time.	CF
standard_name	string	Where defined, a standard and unique description of a physical quantity. For the complete list of standard name strings, see [AD-6].	CF, ACDD
comment	string	Miscellaneous information about the variable or the methods used to produce it.	CF
axis	String	For use with coordinate variables only. The attribute 'axis' may be attached to a coordinate variable and given one of the values "X", "Y", "Z", or "T", which stand for a longitude, latitude, vertical, or time axis respectively. See: http://cf-pcmdi.llnl.gov/documents/cf-conventions/1.4/cf-conventions.html#coordinate-types	CF
positive	String	For use with a vertical coordinate variable only. May have the value "up" or "down". For example, if an oceanographic netCDF file encodes the depth of the surface as 0 and the depth of 1000 meters as 1000 then the axis would set positive to "down". If a depth of 1000 meters was encoded as -1000, then positive would be set to "up".	CF
coordinates	String	Identifies auxiliary coordinate variables, label variables, and alternate coordinate variables. See the section on coordinate-system in. This attribute must be provided if the data are on a non-regular lat/lon grid (map projection or swath data).	CF
flag_meanings	String	Space-separated list of text descriptions associated in strict order with conditions	CF

		set by either flag_values or flag_masks. Words within a phrase should be connected with underscores.	
`flag_values`	Must be the same as the variable type	Comma-separated array of valid, mutually exclusive variable values (required when the bit field contains enumerated values; i.e., a "list" of conditions). Used primarily for **`quality_level`** variable.	CF
`flag_masks`	Must be the same as the variable type	Comma-separated array of valid variable masks (required when the bit field contains independent Boolean conditions; i.e., a bit "mask"). Used primarily for **`l2p_flags`** variable. *Note: CF allows the use of both flag_masks and flag_values attributes in a single variable to create sets of masks that each have their own list of flag_values (see http://cf-pcmdi.llnl.gov/documents/cf-conventions/1.5/ch03s05.html#id2710752 for examples), but this practice is discouraged.*	CF
`depth`	String	Use this to indicate the depth for which the current data are valid. Follow the same rule as for parameter string to expressed depth, as defined in section	
`height`	String	Use this to indicate the height for which the wind (or any atmosphere) data are specified.	

Variable attributes in orange are required for SEAScope platform, those in green are highly recommended. All the others are optional and depends on the variable.

3.4 Definitions of storage types within IDF

Computer storage types referred to in this document follow those used in netCDF. The preferred storage type is chosen to minimize the variable and idf file size: **aside from space-time coordinates related to GCPs, all variables should be stored as unsigned bytes with scale_factor and add_offset attributes.**

More details can be found in the following section.

3.5 Space Time coordinate variables

NetCDF coordinate variables provide location in space and time for multidimensional data arrays. Thus, they must be included for all dimensions that can be identified as spatio-temporal axes. Coordinate arrays (longitude and latitude) are used to geolocate data arrays on non-orthogonal grids, such as images in the original pixel/scan line space, or complicated map projections. ‘unit’ attribute is required.

GCPs dimensions define the size of spatial and temporal coordinates. GCP variables refer to the position of each pixel on original data. Coordinates arrays need to be monotonically ordered. Data storage types are 32 bits float for longitudes/latitudes, 64 bits for time and 32 bits signed integer for GCPs. If variables are time dependent, time values are stored as seconds since 1970-01-01.

For non time related variables, reference time depends on the <Processing Level> of data and is defined as follows:

- L2P: start time of granule;
- L3U: start time of granule;
- L3C and L3S: center time of the collation window;
- L4: nominal time of the analysis

Adding the time dimension into variables facilitates aggregation of all files of a given dataset along the time axis with such tools as THREDDS and LAS (this applies to gridded data only).

If variables are stored in 2D in the file, columns and lines grid dimensions are referred either as:

- ‘lat’ and ‘lon’ if data are mapped on a regular grid (some geostationary products).
- 'x’ and 'y' are used if data are mapped on a projected non-regular grid
- 'cell' and 'row' for sensor scanning pattern (scan line, swath). “cell`'` must be used for the across-track dimension and `'row'` for the along-track dimension.

Coordinate vectors are used for data arrays located on orthogonal (but not necessarily regularly spaced) grids, such as a geographic (lat-lon) map projections. The only required attribute is units. The elements of a coordinate vector array should be in monotonically increasing or decreasing order.

A coordinate variable in 2D must be provided if data are on a non-regular lat/lon grid (map projection or swath data).

A grid_mapping ("projection name") must be provided if data are mapped following a projection. Refer to the CF convention for standard projection names.

The definition of coordinate variables and dimensions is highly dependent on data geometry. The following datamodels can be defined:

- Regular latitude/longitude grids
- Non regular or unstructured grids
- Swath grids
- Time dependent variables in 1D

The datamodel list only refer to data that have already been converted into IDF. It will of course evolve when more data are converted into IDF.

3.5.1 Regular latitude/longitude grids

This is the simplest case. Many L3 and L4 products as well as some geostationary L2 products are provided on a regular lat/lon grid. On such a projection, an IDF file is not necessary as the grid is constant and GCPs can be computed or stored if there is enough information in the original file. On one hand, it avoids us to duplicate already well-formated data, on the other hand one have to write specific routines to convert these non IDF data in SEASCOPE.

The **`time`** dimension be specified as **`unlimited`**. Although in L3 and L4 granules there is only one time dimension (**`time=1`**) and variable **`time`** has only one value (seconds since 1970-01-01), setting an unlimited dimension for **`time`** will allow netCDF tools and utilities to easily concatenate (and average for example) a series of time consecutive granules. The following CDL is provided as an example:

```
netcdf example {
dimensions:
      lat = 1801 ;
      lon = 3600 ;
      gcp_lon = 180;
      gcp_lat = 360 ;
      time = UNLIMITED ;
variables:
      …
}
```

For regular lat/lon grid, the following dimension and coordinate variables are used:

```
dimensions:
      lon = 7200 ;
      lat = 521 ;
      lon_gcp = 720;
      lat_gcp = 52 ;
      time = UNLIMITED ;
   double time(time) ;
       time:long_name = "time" ;
       time:standard_name = "time" ;
       time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
       time:calendar = "standard" ;
       time:axis = "T" ;
    float lat_gcp(lat_gcp) ;
       lat_gcp:long_name = "ground control points latitude" ;
       lat_gcp:standard_name = "latitude" ;
       lat_gcp:units = "degrees_north" ;
       lat:axis = "Y" ;
       lat:comment = "geographical coordinates, WGS84 projection" ;
    float lon_gcp(lon_gcp) ;
       lon_gcp:long_name = "ground control points longitude" ;
       lon_gcp:standard_name = "longitude" ;
       lon_gcp:units = "degrees_east" ;
```

```
      lon_gcp:axis = "X" ;
      lon_gcp:comment = "geographical coordinates, WGS84 projection" ;
   int index_lat_gcp(lat_gcp) ;
       index_lat_gcp:long_name = "index of ground control points in lat
dimension" ;
      index_lat_gcp:comment = "index goes from 0 (start of first pixel) to
dimension value (end of last pixel)" ;
   int index_lon_gcp(lon_gcp) ;
       index_lon_gcp:long_name = "index of ground control points in lon
dimension" ;
      index_lon_gcp:comment = "index goes from 0 (start of first pixel) to
dimension value (end of last pixel)" ;
```

The dimensions of a variable is (time, lat, lon):

```
   ubyte variable(time, lat, lon) ;
      variable:_FillValue = 255UB ;
      variable:long_name = "variable long name" ;
      variable:standard_name = "variable_standard_name" ;
      variable:units = "variable Unit" ;
      variable:add_offset = offset (float);
      variable:scale_factor = offset (float) ;
```

3.5.2 Non-regular or unstructured grids (projection or finite element mesh)

In this datamodel, coordinates are in some projection, x (resp y) varies with y (resp x). GCPs are no longer regularly spaced in this situation and thus are required in the IDF.

As in the previous model, the **`time`** dimension is specified as **`unlimited`**, there is only one time dimension (**`time=1`**) and variable **`time`** has only one value (seconds since 1970-01-01), setting an unlimited dimension for **`time`** will allow netCDF tools and utilities to easily concatenate (and average for example) a series of time consecutive granules given that they have similar dimension in x and y. The following CDL is provided as an example:

```
netcdf example {
dimensions:
      lat = 1801 ;
      lon = 3600 ;
      gcp_lon = 180;
      gcp_lat = 360 ;
      time = UNLIMITED ;
variables:
      …
}
```

For projected y/x grid, the following dimension and coordinate variables are used:

```
dimensions:
      y = 7200 ;
      x = 521 ;
      y_gcp = 720;
      x_gcp = 52 ;
      time = UNLIMITED ;
variables:
    double time(time) ;
       time:long_name = "time" ;
       time:standard_name = "time" ;
       time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
       time:calendar = "standard" ;
       time:axis = "T" ;
    float lat_gcp(y_gcp, x_gcp) ;
       lat_gcp:long_name = "ground control points latitude" ;
       lat_gcp:standard_name = "latitude" ;
       lat_gcp:units = "degrees_north" ;
       lat_gcp:comment = "geographical coordinates, WGS84 projection" ;
    float lon_gcp(y_gcp, x_gcp) ;
       lon_gcp:long_name = "ground control points longitude" ;
       lon_gcp:standard_name = "longitude" ;
       lon_gcp:units = "degrees_east" ;
       lon_gcp:comment = "geographical coordinates, WGS84 projection" ;
    int index_y_gcp(y_gcp) ;
        index_y_gcp:long_name = "index of ground control points in y
dimension" ;
       index_y_gcp:comment = "index goes from 0 (start of first pixel) to
dimension value (end of last pixel)" ;
    int index_x_gcp(x_gcp) ;
        index_x_gcp:long_name = "index of ground control points in x
dimension" ;
       index_x_gcp:comment = "index goes from 0 (start of first pixel) to
dimension value (end of last pixel)" ;
```

The dimension of a variable is (time, y, x) :

```
    ubyte variable(time, y, x) ;
       variable:_FillValue = 255UB ;
       variable:long_name = "variable long name" ;
       variable:standard_name = "variable standard" ;
       variable:units = "variable unit" ;
       variable:add_offset = offset (float) ;
       variable:scale_factor = scale_factor (float) ;
```

```
        variable:valid_min = 0UB ;
        variable:valid_max = 254UB ;
```

3.5.3 Swath grids

In this case where data are gridded following the sensor pattern, no projection can be associated and lat/lon data have to be stored in 2-D arrays. Dimensions row and cell should be used to describe the swath. As a best practice, the cell dimension should refer to the cross-track direction and the dimension row should refer to the along-track direction of a polar orbiting (or similar) satellite sensor swath.

Some metadata variables (such as incidence angle in SAR or view/sun angles in visible products) can be defined on the GCP grid (using gcp dimensions) rather than on the product variable grid.

For these cases, the following dimension and coordinate variables is used:

```
dimensions:
    time = 1 ;
    row = 2030 ;
    cell = 2361 ;
    row_gcp = 21 ;
    cell_gcp = 25 ;
variables:
    double time(time) ;
       time:long_name = "time" ;
       time:standard_name = "time" ;
       time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
       time:calendar = "standard" ;
       time:axis - "T" ;
    float lat_gcp(row_gcp, cell_gcp) ;
       lat_gcp:long_name = "ground control points latitude" ;
       lat_gcp:standard_name = "latitude" ;
       lat_gcp:units = "degrees_north" ;
       lat_gcp:comment = "geographical coordinates, WGS84 projection" ;
    float lon_gcp(row_gcp, cell_gcp) ;
       lon_gcp:long_name = "ground control points longitude" ;
       lon_gcp:standard_name = "longitude" ;
       lon_gcp:units = "degrees_east" ;
       lon_gcp:comment = "geographical coordinates, WGS84 projection" ;
    int index_row_gcp(row_gcp) ;
       index_row_gcp:long_name = "index of ground control points in row
dimension" ;
       index_row_gcp:comment = "index goes from 0 (start of first pixel) to
dimension value (end of last pixel)" ;
    int index_cell_gcp(cell_gcp) ;
       index_cell_gcp:long_name = "index of ground control points in cell
dimension" ;
       index_cell_gcp:comment = "index goes from 0 (start of first pixel) to
dimension value (end of last pixel)" ;
```

The dimension of a variable is (time, row, cell):

```
ubyte variable(time, row, cell) ;
       variable:_FillValue = 255UB ;
       variable:long_name = "variable long name" ;
       variable:standard_name = "variable standard name" ;
       variable:units = "variable unit" ;
       variable:add_offset = offset (float) ;
       variable:scale_factor = scale factor (float) ;
       variable:valid_min = 0UB ;
       variable:valid_max = 254UB ;dimensions:
```

3.5.4 Time dependent variables (in 1D)

This data pattern corresponds to any fixed platform, drifting buoy or along track sensors such as altimeters.

In this case where data are not gridded but vectorized along a time axis, no projection can be associated and lat/lon data have to be stored in 1-D arrays. Dimensions cannot be referred to as lat/lon anymore. Instead the main coordinate dimension is **`time`** and data are stored in chronological order along this time axis. For fixed platform or station, latitude, longitude and depth (which is optional) have one value (they have the dimension of station = 1) as they are not changing with time. Latitude, longitude and depth (also optional) are defined for each time GCP for moving sensors. The time variable is in seconds since 1970-01-01.

For these cases, the following dimension and coordinate variables shall be used:

```
dimensions:
      time = 1231 ;
      station = 1;
      time_gcp = 123 ;
variables:
      double time(time_gcp) ;
            time:long_name = "time" ;
            time:standard_name = "time" ;
            time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
            time:calendar = "standard" ;
            time:axis: “T”
      float lat_gcp(time_gcp) ;
            lat_gcp:long_name = "ground control points latitude" ;
            lat_gcp:standard_name = "latitude" ;
            lat_gcp:units = "degrees_north" ;
            lat_gcp:comment = "geographical coordinates, WGS84 projection" ;
      float lon_gcp(time_gcp) ;
            lon_gcp:long_name = "ground control points longitude" ;
            lon_gcp:standard_name = "longitude" ;
```

```
            lon_gcp:units = "degrees_east" ;
            lon_gcp:comment = "geographical coordinates, WGS84 projection" ;
      int index_time_gcp(time_gcp) ;
            index_time_gcp:long_name = "index of ground control points in
time dimension" ;
            index_time_gcp:comment = "index goes from 0 (first pixel) to
value dimension" ;
```

The dimensions of a variable will then be (time):

```
ubyte variable(time) ;
      variable:_FillValue = 255UB ;
      variable:long_name = “variable long name” ;
      variable:standard_name = “variable standard name” ;
      variable:unit = “variable unit" ;
      variable:add_offset = offset (float) ;
      variable:scale_factor = scale (float) ;
      variable:valid_min = 0UB ;
      variable:valid_max = 254UB ;
```

3.6 Geophysical variables

3.6.1 Naming convention

All these variables need to be uniquely and consistently named across all products if some products provide the same measured quantity. This section list all the identified variables.

The rules behind the naming of these variables are as follow:

- avoid ambiguous naming such as 'u' and 'v' (for vector components). Explicit and unambiguous names are recommended.
- vectors should be stored as northward and eastward components (not module and direction). Explicit module variable can be provided in addition. For better integration in visualization tools, vector components must be named as:
  - northward_<*geophysical quantity*>
  - eastward_<*geophysical quantity*>
- some instruments can only measure a single component of the current vector. Use one of the following prefix:
  - `acrosstrack` for a vector quantity measured along a satellite track (altimetry derived velocities)
  - `lineofsight` for the radial component of the velocity (SAR derived velocities)
- avoid naming variables after the methodology (“mcc_current”) or instrument (“sar_doppler_current”) used to retrieve it. Currents are a geophysical quantity and a proper naming needs to be found to describe the component measured.
- For geophysical quantity other than currents, use the conventions agreed in other projects (ex: GHRSST for sea surface temperature)
- use “_bias” and “_stddev” suffixes to express respectively the bias and standard deviation of a quantity
- use “_error” suffix to express the estimation error on a quantity

4 Examples of IDF

4.1 Lat/Lon grid datamodel

4.1.1 ECMWF wind

```
netcdf ECMWF_20141229T00Z_idf_00 {
dimensions:
    time = UNLIMITED ; // (1 currently)
    lat = 1409 ;
    lon = 2880 ;
    lat_gcp = 178 ;
    lon_gcp = 361 ;
variables:
    double time(time) ;
       time:long_name = "time" ;
       time:standard_name = "time" ;
       time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
       time:calendar = "standard" ;
    float lat_gcp(lat_gcp) ;
       lat_gcp:long_name = "ground control points latitude" ;
       lat_gcp:standard_name = "latitude" ;
       lat_gcp:units = "degrees_north" ;
    float lon_gcp(lon_gcp) ;
       lon_gcp:long_name = "ground control points longitude" ;
       lon_gcp:standard_name = "longitude" ;
       lon_gcp:units = "degrees_east" ;
    int index_lat_gcp(lat_gcp) ;
       index_lat_gcp:long_name = "index of ground control points in lat
dimension" ;
       index_lat_gcp:comment = "index goes from 0 (start of first pixel) to
dimension value (end of last pixel)" ;
    int index_lon_gcp(lon_gcp) ;
       index_lon_gcp:long_name = "index of ground control points in lon
dimension" ;
       index_lon_gcp:comment = "index goes from 0 (start of first pixel) to
dimension value (end of last pixel)" ;
    ubyte u10m(time, lat, lon) ;
       u10m:_FillValue = 255UB ;
       u10m:long_name = "u component of horizontal wind" ;
       u10m:standard_name = "eastward_wind" ;
       u10m:units = "m s-1" ;
       u10m:add_offset = -25.4 ;
       u10m:scale_factor = 0.2 ;
       u10m:valid_min = 0UB ;
       u10m:valid_max = 254UB ;
    ubyte v10m(time, lat, lon) ;
```

```
        v10m:_FillValue = 255UB ;
        v10m:long_name = "v component of horizontal wind" ;
        v10m:standard_name = "northward_wind" ;
        v10m:units = "m s-1" ;
        v10m:add_offset = -25.4 ;
        v10m:scale_factor = 0.2 ;
        v10m:valid_min = 0UB ;
        v10m:valid_max = 254UB ;

// global attributes:
        :idf_granule_id = "ECMWF_20141229T00Z" ;
        :time_coverage_start = "2014-12-28T22:30:00.000000Z" ;
        :time_coverage_end = "2014-12-29T01:30:00.000000Z" ;
        :idf_subsampling_factor = 0 ;
        :idf_spatial_resolution = 13875. ;
        : spatial_resolution_units = “m” ;
        :title = "ECMWF wind" ;
        :institution = "Cep; OceanDataLab" ;
        :comment = "" ;
        :id = "ECMWF_20141229T00Z_idf_00" ;
        :product_version = "1.0" ;
        :geospatial_lat_max = "88" ;
        :geospatial_lat_min = "-88" ;
        :geospatial_lon_max = "180" ;
        :geospatial_lon_min = "-180" ;
        :creator_email = “contact@oceandatalab.com”
}
```

4.1.2 GlobCurrent L4

```
netcdf \20141229000000-GLOBCURRENT-L4-CURgeo_0m-ALT_OI-v02.0-fv01.0_idf_00 {
dimensions:
    time = UNLIMITED ; // (1 currently)
    lat = 1280 ;
    lon = 2880 ;
    lat_gcp = 161 ;
    lon_gcp = 361 ;
variables:
    double time(time) ;
       time:long_name = "time" ;
       time:standard_name = "time" ;
       time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
       time:calendar = "standard" ;
    float lat_gcp(lat_gcp) ;
       lat_gcp:long_name = "ground control points latitude" ;
       lat_gcp:standard_name = "latitude" ;
       lat_gcp:units = "degrees_north" ;
    float lon_gcp(lon_gcp) ;
```

```
        lon_gcp:long_name = "ground control points longitude" ;
        lon_gcp:standard_name = "longitude" ;
        lon_gcp:units = "degrees_east" ;
    int index_lat_gcp(lat_gcp) ;
        index_lat_gcp:long_name = "index of ground control points in lat dimension" ;
        index_lat_gcp:comment = "index goes from 0 (start of first pixel) to dimension value (end of last pixel)" ;
    int index_lon_gcp(lon_gcp) ;
        index_lon_gcp:long_name = "index of ground control points in lon dimension" ;
        index_lon_gcp:comment = "index goes from 0 (start of first pixel) to dimension value (end of last pixel)" ;
    ubyte eastward_geostrophic_current_velocity(time, lat, lon) ;
        eastward_geostrophic_current_velocity:_FillValue = 255UB ;
        eastward_geostrophic_current_velocity:long_name = "Absolute geostrophic velocity: zonal component" ;
        eastward_geostrophic_current_velocity:units = "m s-1" ;
        eastward_geostrophic_current_velocity:add_offset = -5.08 ;
        eastward_geostrophic_current_velocity:scale_factor = 0.04 ;
        eastward_geostrophic_current_velocity:valid_min = 0UB ;
        eastward_geostrophic_current_velocity:valid_max = 254UB ;
    ubyte northward_geostrophic_current_velocity(time, lat, lon) ;
        northward_geostrophic_current_velocity:_FillValue = 255UB ;
        northward_geostrophic_current_velocity:long_name = "Absolute geostrophic velocity: meridian component" ;
        northward_geostrophic_current_velocity:units = "m s-1" ;
        northward_geostrophic_current_velocity:add_offset = -5.08 ;
        northward_geostrophic_current_velocity:scale_factor = 0.04 ;
        northward_geostrophic_current_velocity:valid_min = 0UB ;
        northward_geostrophic_current_velocity:valid_max = 254UB ;

// global attributes:
        :idf_granule_id = "20141229000000-GLOBCURRENT-L4-CURgeo_0m-ALT_OI-v02.0-fv01.0" ;
        :time_coverage_start = "2014-12-28T12:00:00.000000Z" ;
        :time_coverage_end = "2014-12-29T12:00:00.000000Z" ;
        :idf_subsampling_factor = 0 ;
        :idf_spatial_resolution = 13875. ;
        :idf_spatial_resolution_units = "m" ;
        :title = "Globcurrent geostrophic surface velocity" ;
        :institution = "CLS; OceanDataLab" ;
        :comment = "Geostrophic current from mapped ADT altimeters … " ;
        :id = "20141229000000-GLOBCURRENT-L4-CURgeo_0m-ALT_OI-v02.0-fv01.0_idf_00" ;
        :product_version = "1.0" ;
        :geospatial_lat_max = "90" ;
        :geospatial_lat_min = "-90" ;
        :geospatial_lon_max = "180" ;
```

```
        :geospatial_lon_min = "-180" ;
        :sensor = “HY-2A; Jason-2; Altika; Cryosat-2”
        :creator_email = “contact@oceandatalab.com”;
}
```

4.1.3 Odyssea SST

```
netcdf \20141228-IFR-L4_GHRSST-SSTfnd-ODYSSEA-SAF_002-v2.0-fv1.0_idf_00 {
dimensions:
    time = UNLIMITED ; // (1 currently)
    lat = 1750 ;
    lon = 2250 ;
    lat_gcp = 36 ;
    lon_gcp = 46 ;
variables:
    double time(time) ;
        time:long_name = "time" ;
        time:standard_name = "time" ;
        time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
        time:calendar = "standard" ;
    float lat_gcp(lat_gcp) ;
        lat_gcp:long_name = "ground control points latitude" ;
        lat_gcp:standard_name = "latitude" ;
        lat_gcp:units = "degrees_north" ;
    float lon_gcp(lon_gcp) ;
        lon_gcp:long_name = "ground control points longitude" ;
        lon_gcp:standard_name = "longitude" ;
        lon_gcp:units = "degrees_east" ;
    int index_lat_gcp(lat_gcp) ;
        index_lat_gcp:long_name = "index of ground control points in lat
dimension" ;
        index_lat_gcp:comment = "index goes from 0 (start of first pixel) to
dimension value (end of last pixel)" ;
    int index_lon_gcp(lon_gcp) ;
        index_lon_gcp:long_name = "index of ground control points in lon
dimension" ;
        index_lon_gcp:comment = "index goes from 0 (start of first pixel) to
dimension value (end of last pixel)" ;
    ubyte analysed_sst(time, lat, lon) ;
        analysed_sst:_FillValue = 255UB ;
        analysed_sst:long_name = "analysed sea surface temperature" ;
        analysed_sst:standard_name = "sea_surface_temperature" ;
        analysed_sst:units = "K" ;
        analysed_sst:add_offset = 271.05 ;
        analysed_sst:scale_factor = 0.15 ;
        analysed_sst:valid_min = 0UB ;
        analysed_sst:valid_max = 254UB ;

// global attributes:
```

```
       :idf_granule_id = "20141228-IFR-L4_GHRSST-SSTfnd-ODYSSEA-SAF_002-v2.0-fv1.0" ;
       :time_coverage_start = "2014-12-27T12:00:00.000000Z" ;
       :time_coverage_end = "2014-12-28T12:00:00.000000Z" ;
       :idf_subsampling_factor = 0 ;
       :idf_spatial_resolution = 2220. ;
       :idf_spatial_resolution_units = "m" ;
 :title = "ODYSSEA Sea Surface temperature Analysis in Aghulas" ;
       :institution = "Ifremer; OceanDataLaba " ;
       :comment = "Analysed Sea Surface temperature computed using optimal interpolation with Odyssea processor 2.0" ;
       :id = "20141228-IFR-L4_GHRSST-SSTfnd-ODYSSEA-SAF_002-v2.0-fv1.0_idf_00" ;
       :product_version = "1.0" ;
       :geospatial_lat_max = "-10" ;
       :geospatial_lat_min = "-45" ;
       :geospatial_lon_max = "10" ;
       :geospatial_lon_min = "55" ;
       :creator_email = “contact@oceandatalab.com”;
       :sensor = “AVHRR;AVHRR;MODIS_A;AATSR”;
       :platform = “METOP-A; NOAA18;NOAA19;AQUA;ENVISAT”;

}
```

4.2 Projected y/x grid datamodel

4.2.1 AMSR sea ice concentration

```
netcdf asi-AMSR2-n6250-20141201-v5_idf_00 {
dimensions:
    time = UNLIMITED ; // (1 currently)
    y = 1792 ;
    x = 1216 ;
    y_gcp = 113 ;
    x_gcp = 77 ;
variables:
    double time(time) ;
       time:long_name = "time" ;
       time:standard_name = "time" ;
       time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
       time:calendar = "standard" ;
    float lat_gcp(y_gcp, x_gcp) ;
       lat_gcp:long_name = "ground control points latitude" ;
       lat_gcp:standard_name = "latitude" ;
       lat_gcp:units = "degrees_north" ;
    float lon_gcp(y_gcp, x_gcp) ;
       lon_gcp:long_name = "ground control points longitude" ;
       lon_gcp:standard_name = "longitude" ;
       lon_gcp:units = "degrees_east" ;
    int index_y_gcp(y_gcp) ;
```

```
       index_y_gcp:long_name = "index of ground control points in y dimension"
;
       index_y_gcp:comment = "index goes from 0 (start of first pixel) to
dimension value (end of last pixel)" ;
    int index_x_gcp(x_gcp) ;
       index_x_gcp:long_name = "index of ground control points in x dimension"
;
       index_x_gcp:comment = "index goes from 0 (start of first pixel) to
dimension value (end of last pixel)" ;
    ubyte sea_ice_concentration(time, y, x) ;
       sea_ice_concentration:_FillValue = 255UB ;
       sea_ice_concentration:long_name = "ASI Ice Concentration" ;
       sea_ice_concentration:units = "percent" ;
       sea_ice_concentration:add_offset = 0. ;
       sea_ice_concentration:scale_factor = 0.393700787401575 ;
       sea_ice_concentration:valid_min = 0UB ;
       sea_ice_concentration:valid_max = 254UB ;

// global attributes:
       :idf_granule_id = "asi-AMSR2-n6250-20141201-v5" ;
       :time_coverage_start = "2014-12-01T00:00:00.000000Z" ;
       :time_coverage_end = "2014-12-02T00:00:00.000000Z" ;
       :idf_subsampling_factor = 0 ;
       :idf_spatial_resolution = 6250. ;
       :idf_spatial_resolution_units = "m" ;
}
```

4.3 Swath datamodel

4.3.1 Modis SST

```
netcdf A2014290131500.L2_LAC_SST_idf_00 {
dimensions:
    time = 1 ;
    row = 2030 ;
    cell = 2361 ;
    row_gcp = 21 ;
    cell_gcp = 25 ;
variables:
    double time(time) ;
       time:long_name = "time" ;
       time:standard_name = "time" ;
       time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
       time:calendar = "standard" ;
    float lat_gcp(row_gcp, cell_gcp) ;
       lat_gcp:long_name = "ground control points latitude" ;
       lat_gcp:standard_name = "latitude" ;
       lat_gcp:units = "degrees_north" ;
    float lon_gcp(row_gcp, cell_gcp) ;
       lon_gcp:long_name = "ground control points longitude" ;
```

```
        lon_gcp:standard_name = "longitude" ;
        lon_gcp:units = "degrees_east" ;
    int index_row_gcp(row_gcp) ;
        index_row_gcp:long_name = "index of ground control points in row dimension" ;
        index_row_gcp:comment = "index goes from 0 (start of first pixel) to dimension value (end of last pixel)" ;
    int index_cell_gcp(cell_gcp) ;
        index_cell_gcp:long_name = "index of ground control points in cell dimension" ;
        index_cell_gcp:comment = "index goes from 0 (start of first pixel) to dimension value (end of last pixel)" ;
    ubyte denoised_sst(time, row, cell) ;
        denoised_sst:_FillValue = 255UB ;
        denoised_sst:long_name = "denoised sea surface temperature" ;
        denoised_sst:standard_name = "sea_surface_temperature" ;
        denoised_sst:units = "K" ;
        denoised_sst:add_offset = 283.200097855866 ;
        denoised_sst:scale_factor = 0.0511811023622047 ;
        denoised_sst:valid_min = 0UB ;
        denoised_sst:valid_max = 254UB ;

// global attributes:
        :idf_granule_id = "A2014290131500.L2_LAC_SST" ;
        :time_coverage_start = "2014-10-17T13:15:08.000000Z" ;
        :time_coverage_end = "2014-10-17T13:20:07.000000Z" ;
        :idf_subsampling_factor = 0 ;
        :idf_spatial_resolution = 1000. ;
        :idf_spatial_resolution_units = "m" ;
}
```

4.3.2 S-1 sea surface roughness

```
netcdf s1a-ew-grd-hh-20141202t183501-20141202t183605-003544-0042c3-001-F64E_idf_00 {
dimensions:
    time = 1 ;
    row = 5339 ;
    cell = 5181 ;
    row_gcp = 23 ;
    cell_gcp = 21 ;
variables:
    double time(time) ;
        time:long_name = "time" ;
        time:standard_name = "time" ;
        time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
        time:calendar = "standard" ;
    float lat_gcp(row_gcp, cell_gcp) ;
        lat_gcp:long_name = "ground control points latitude" ;
```

```
        lat_gcp:standard_name = "latitude" ;
        lat_gcp:units = "degrees_north" ;
    float lon_gcp(row_gcp, cell_gcp) ;
        lon_gcp:long_name = "ground control points longitude" ;
        lon_gcp:standard_name = "longitude" ;
        lon_gcp:units = "degrees_east" ;
    int index_row_gcp(row_gcp) ;
        index_row_gcp:long_name = "index of ground control points in row dimension" ;
        index_row_gcp:comment = "index goes from 0 (start of first pixel) to dimension value (end of last pixel)" ;
    int index_cell_gcp(cell_gcp) ;
        index_cell_gcp:long_name = "index of ground control points in cell dimension" ;
        index_cell_gcp:comment = "index goes from 0 (start of first pixel) to dimension value (end of last pixel)" ;
    ubyte sea_surface_roughness(time, row, cell) ;
        sea_surface_roughness:_FillValue = 255UB ;
        sea_surface_roughness:long_name = "sea surface roughness" ;
        sea_surface_roughness:units = "1" ;
        sea_surface_roughness:add_offset = 0. ;
        sea_surface_roughness:scale_factor = 0.00787401574803l5 ;
        sea_surface_roughness:valid_min = 0UB ;
        sea_surface_roughness:valid_max = 254UB ;

// global attributes:
        :idf_granule_id = "s1a-ew-grd-hh-20141202t183501-20141202t183605-003544-0042c3-001-F64E" ;
        :time_coverage_start = "2014-12-02T18:35:01.251000Z" ;
        :time_coverage_end = "2014-12-02T18:36:05.342000Z" ;
        :idf_subsampling_factor = 0 ;
        :idf_spatial_resolution = 80. ;
        :idf_spatial_resolution_units = "m" ;
}
```

4.4 Trajectory datamodel

No subsampling has been performed yet on trajectories as spatial resolution is difficult to define. It will be done for the next version of the document.

4.4.1 Surface Drifter from AOML

```
netcdf AOML_2015_12_139922_idf_00 {
dimensions:
      time = 123 ;
      time_gcp = 123 ;
variables:
      double time(time_gcp) ;
            time:long_name = "time" ;
            time:standard_name = "time" ;
            time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
```

```
            time:calendar = "standard" ;
      float lat_gcp(time_gcp) ;
            lat_gcp:long_name = "ground control points latitude" ;
            lat_gcp:standard_name = "latitude" ;
            lat_gcp:units = "degrees_north" ;
      float lon_gcp(time_gcp) ;
            lon_gcp:long_name = "ground control points longitude" ;
            lon_gcp:standard_name = "longitude" ;
            lon_gcp:units = "degrees_east" ;
      int index_time_gcp(time_gcp) ;
            index_time_gcp:long_name = "index of ground control points in time
dimension" ;
            index_time_gcp:comment = "index goes from 0 (first pixel) to value
dimension" ;
      ubyte current(time) ;
            current:_FillValue = 255UB ;
            current:unittype = "m/s" ;
            current:add_offset = 0.02351f ;
            current:scale_factor = 0.003844449f ;
            current:valid_min = 0UB ;
            current:valid_max = 254UB ;
      ubyte temp(time) ;
            temp:_FillValue = 255UB ;
            temp:unittype = "K" ;
            temp:add_offset = 302.565f ;
            temp:scale_factor = 0.006456693f ;
            temp:valid_min = 0UB ;
            temp:valid_max = 254UB ;

// global attributes:
            :idf_granule_id = "AOML_2015_12_139922" ;
            :time_coverage_start = "2015-12-01T06:00:00.000000Z" ;
            :time_coverage_end = "2015-12-31T18:00:00.000000Z" ;
            :platform_code = "139923" ;
            :idf_subsampling_factor = 0 ;
            :idf_spatial_resolution = 1.e+07f ;
            :idf_spatial_resolution_units = "m" ;
 :title = "Temperature and curent from SVP data filtered by AOML" ;
       :institution = "AOML; OceanDataLaba " ;
       :comment = "Positions have been filtered using Lumpkin algorithm" ;
       :id = "AOML_2015_12_139922_idf_00" ;
       :product_version = "1.0" ;
       :geospatial_lat_max = "-9.00" ;
       :geospatial_lat_min = "-9.792" ;
       :geospatial_lon_min = "179.917" ;
       :geospatial_lon_max = "-178.3" ;
       :creator_email = “contact@oceandatalab.com”;
       :platform_code = “139923”;

}
```

4.4.2 L3 along track Jason 2 SLA

```
netcdf dt_global_j2_sla_vfec_20151201_20160610_c0273_p024_idf_00 {
```

```
dimensions:
    time = 1201 ;
    time_gcp = 1201 ;
variables:
    double time(time_gcp) ;
       time:long_name = "time" ;
       time:standard_name = "time" ;
       time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
       time:calendar = "standard" ;
    float lat_gcp(time_gcp) ;
       lat_gcp:long_name = "ground control points latitude" ;
       lat_gcp:standard_name = "latitude" ;
       lat_gcp:units = "degrees_north" ;
    float lon_gcp(time_gcp) ;
       lon_gcp:long_name = "ground control points longitude" ;
       lon_gcp:standard_name = "longitude" ;
       lon_gcp:units = "degrees_east" ;
    int index_time_gcp(time_gcp) ;
       index_time_gcp:long_name = "index of ground control points in time dimension" ;
       index_time_gcp:comment = "index goes from 0 (first pixel) to value dimension" ;
    ubyte SLA(time) ;
       SLA:_FillValue = 255UB ;
       SLA:unittype = "m" ;
       SLA:add_offset = -0.451f ;
       SLA:scale_factor = 0.003622047f ;
       SLA:valid_min = 0UB ;
       SLA:valid_max = 254UB ;

// global attributes:
       :idf_granule_id = "dt_global_j2_sla_vfec_20151201_20160610_c0273_p024" ;
       :time_coverage_start = "2015-12-01T00:17:42.520577Z" ;
       :time_coverage_end = "2015-12-01T01:00:51.112577Z" ;
       :cycle = 273 ;
       :orbit = 24 ;
       :idf_subsampling_factor = 0 ;
       :idf_spatial_resolution = 1.e+07f ;
       :idf_spatial_resolution_units = "m" ;
       :title = "Sea Level Anomaly" ;
       :institution = "AVISO; OceanDataLaba " ;
       :comment = "" ;
       :id = "dt_global_j2_sla_vfec_20151201_20160610_c0273_p024" ;
       :product_version = "1.0" ;
       :geospatial_lat_max = "61" ;
       :geospatial_lat_min = "-61" ;
       :geospatial_lon_min = "-92.05706" ;
```

```
        :geospatial_lon_max = "-9.022436" ;
        :creator_email = “contact@oceandatalab.com”;
        :sensor = “Jason2”;
}
```

5 Annex

5.1 Open points

5.1.1 Flexibility to use non-IDF netCDF-4 data

Corresponding section: 2.2 NetCDF-4 classic file format

NetCDF-4 format is already used by some data producers and the number of NetCDF-4 formatted data will probably increase in the future. A direct streaming of such data instead of converting them into IDF would avoid data duplication and facilitate data "ingestion" into SEAScope. It is the general idea but it has to be seen case by case because of potential blocking points:

- Data needs a special pre-processing preferably done off-line.
- High resolution data needs multiresolution support
- Data geolocation pattern is outside the scope of IDF geolocation patterns

Some other blocking points need to be investigated along with the implementation of SEAScope data streaming service:

- Storage data type compliant with required fast access and streaming?
- Multiple granules in a file? Multiple files for a granule?

5.1.2 NetCDF-4 specific settings (chunk size, digit precision and data compression)

Corresponding section: 2.2 NetCDF-4 classic file format

NetCDF-4 format has some settings for optimizing data access and reducing file size. For now, no specific chunk size nor data compression are used and the geophysical variables are simply stored with 8 bits without digit precision (hence the use of a scale factor and an add offset). These settings will be investigated along with the implementation of SEAScope data streaming service.

5.1.3 Multiresolution guidelines

Corresponding section: 2.3 Multiresolution support

We should indicate clear rules for multiresolution:

- When multiresolution is needed?
- If needed, until which spatial sampling IDF files need to be generated?
- How to define spatial spacing in meters? (eg. irregular trajectory like drifters)

5.1.4 Tiling from GCPs and GCPs density

Corresponding section: 2.4 Geolocation with GCP

GCPs serve two main purposes: data geolocation and virtual tiling. For now, GCPs have been used with a focus on data geolocation since SEAScope tiling is not effective yet. The use of GCPs for tiling will be addressed along with the implementation of SEAScope data tiling. Then some guidelines about GCPs will be added in the document. It will probably add some constraints about where to position GCPs in the data space. Also, it makes more sense to wait for tiling before giving definitive rules about GCPs density.

5.1.5 Unique product identifier

Corresponding section: 3 IDF NetCDF-4 file structure

The product unique identifier is used to fill in the global attribute id described in table 3.2. If a product unique identifier is already defined in the original product, one should use it to redefine and id similar to:

<Original id>-<Subsampling index>

If no id is defined in the original file, one should define a product unique identifier using the following rules.

Best Practice for Establishing Product Identifiers

A best practice has been established for defining the text strings to be used as product unique identifier. While a rigid standard for the text strings is not possible, the following best practice should be applied to the extent possible.

<Producer>-<Processing Level>-<Product String>-v<Product Version>-<subsampling index>

The definitions of the components match the definitions from the file naming convention, found in previous sections. The component <Product Version> is used to distinguish different versions of the same dataset and should be of the form *x.y* where *x* is the major and *y* is the minor version.

The component <Producer> corresponds to the producing agency using the shortname or acronym.

Unique product identifier has not been defined yet in IDF global attributes. Recommended practices regarding its definition may evolve when this issue is tackled.

5.1.6 Data Model

Corresponding section: 3.5 Coordinate variables

Section 3.5 is giving an overview of possible datamodels. This section is going to evolve when various types of data are converted into IDF.

New datamodel shall be defined such as Time Series for fixed sensor, as the only coordinate that varies is the time. There is no need to store longitude and latitude as a function of time.