EGO gliders user's manual

NetCDF conventions and reference tables

Version 1.1

December 20th, 2013

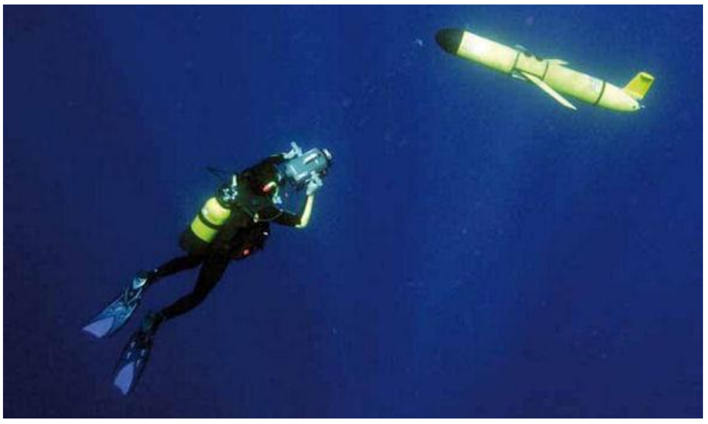

Table of contents

TABLE OF CONTENTS 2

HISTORY 4

CHANGES TO IMPLEMENT AFTER SAN-DIEGO'S MEETING, DECEMBER 2013 4

1 EGO GLIDERS DATA-MANAGEMENT PRINCIPLES 6

1.1 About EGO 6

1.2 About this document 6

1.3 EGO data management structure and data access 6

1.4 User Obligations 6

1.5 Disclaimer 7

1.6 Further Information Sources and Contact Information 7

1.7 Useful links, tools 7
1.7.1 EGO matlab scripts to generate NetCDF files 7
1.7.2 EGO file format checker 7

1.8 Open issues, known problems 8
1.8.1 Warning on CTD thermal lag errors 8
1.8.2 Time sampling issues on Slocum gliders 8

2 EGO GLIDER NETCDF DATA FORMAT VERSION 1.1 9

2.1 NetCDF standards and conventions 9

2.2 Trajectory and profile data formats 10

2.3 Data file dimensions 10

2.4 Global attributes 11

2.5 Variables 16
2.5.1 Coordinate variables 16
2.5.2 Coordinate quality control variables 18
2.5.3 Profile, phase and direction management 20
2.5.4 Positioning method 20
2.5.5 Data variables 22
2.5.6 History information 25

2.6 Gliders metadata: platform – deployment – instrument - sensor 27
2.6.1 Platform metadata container 27
2.6.2 Glider deployment information 28
2.6.3 Glider sensor information 29
2.4.7 Glider parameter information 30
2.6.4 Glider parameter derivation and calibration information 30

2.7 Gliders technical data 32

3 REFERENCE TABLES 33

3.1 Reference tables 1: data type 33

3.2 Reference table 2: Variable quality control flag scale 33
3.2.1 Reference table 2.2: cell methods 33

3.3 Reference table 3: EGO parameter dictionary 34

3.3.1 Convention for parameter names, standard names and units 34
3.3.2 EGO parameter list 34
3.3.3 References 35

3.4 Reference table 4: Data Assembly Center Codes 35

3.5 Reference table 5: Location classes 35

3.6 Reference table 7: History action codes 36

3.7 Reference table 8: Instrument types 36

3.8 Reference table 9: Phases of the glider trajectory 36

3.9 Reference table 10: Positioning method 37

3.10 Reference table 11: QC test binary IDs 37

3.11 Reference table 12: History steps codes 37

3.12 Reference table 19: Data mode 38

3.13 Reference table 20: Sensor mount characteristics 38

3.14 Reference table 21: Sensor orientation characteristics 39

3.15 Reference table 22: type of glider 39

4 USING THE HISTORY SECTION OF THE EGO NETCDF STRUCTURE 40

4.1 Recording information about the Delayed Mode QC process 40

4.2 Recording processing stages 40

4.3 Recording QC Tests Performed and Failed 41

4.4 Recording changes in values 42

5 GDAC ORGANIZATION 44

5.1 File naming convention 44

5.2 Index of glider deployments files 45

6 DATA DISTRIBUTION 47

6.1 DAC to GDAC data distribution 47

6.2 DAC to GTS data distribution 47

7 GLOSSARY, DEFINITIONS 48

7.1 Observatory 48

7.2 Deployment 48

7.3 Glider 48

7.4 Sensor 48
7.4.1 Parameter measured by the sensor 48
7.4.2 Calibration of the parameter measured by the sensor 48

7.5 Principal Investigator (PI) 48

7.6 Global Data Assembly Centre (GDAC) 48

7.7 Data Assembly Centre (DAC) 49

History

Version	Date	Comment
0.9	08/10/2012	TC: initialization of the document based o EGO user's manual version 1.2
0.99	15/10/2012	TC : updates after Paris Groom meeting §2.3.3 : Profile, phase and direction management §2.3.4 : Positioning method §2.3.5 : add a coordinates attribute to <PARAM> §2.4.4 : Configuration parameters §2.4.6 : calibration : note on derived parameters such as PSAL, note on pre-deployment calibrations §2.5 : Gliders technical data §5 : Glossary, definitions
0.999	23/10/2012	GB : use TIME:units = "days since 1970-01-01T00:00:00Z"; instead of 1950 for compatibility with old version software such as ferret
1.0	12/12/2012	TC : use TIME variable (seconds since 01/01/1970) and JULD (days since 01/01/1950) Remove <PARAM>_dm Remove uncertainty as an attribute Remove qc_indicator_attribute Remove EGO glider catalogue Remove configuration parameters chapter DERIVATION instead of CALIBRATION Add a data distribution chapter 6 Add a chapter 1.8 on CTD thermal lag error Manage technical data as standard variables
1.01	03/06/2013	JPR: various updates
1.1	04/06/2013	**Trieste meeting updates** §6.2: GTS tesac distribution : minimal length of profile set to 10 decibars §2.5.5: Add a standard_name attribute to QC flags §2.2: Use EVENT instead of TIME dimension §2.5.1: Manage separately the events recorded by different clocks : TIME_SCIENCE, TIME_NAVIGATION, TIME_GPS
1.1	30/11/2013	Jean-Philippe Rannou and Thierry Carval • Typo corrections • §2.2 information on trajectory and profile file
1.1	07/12/2013	Thierry Carval §2.6 : glider metadata are stored in containers instead of variables. §2.5.5 : add _XXX to discriminate variable names of the same parameter (San-Diego action 2)
1.1	17/12/2013	Justin Buck : ACDD convention implementation

Changes to implement after San-Diego's meeting, December 2013

Changes to implement after San-Diego's meeting, December 2013
Action 1: Derrick to suggest similar new attributes for uom_uri

Action 2: Add the OceanSITES text to the combined manual and/or list of recommendations.

Action 3: Bart will provide Thierry with the STSM report with Lucas. Thierry will make the document available on the web and will provide the link to it.

Action 4: Derrick will add the updated ACDD and IOOS attributes to the comparison spreadsheet. Starting with ACDD_Working.
Thierry/Bart/Seb/Justin will add EGO and ANFOG attributes.

Action 5: Determine the complete list of variable attributes to be included with each metadata container variable.
Thierry and Derrick

Action 6: complete the parameters list on chapter “3.3.2” EGO parameter list. Use the Argo and bio-Argo list as a first choice. Add the parameters that are glider specific (ex: yo, pitch).
Use parameter_uri attribute instead of parameter_urn
Thierry and Justin

Action 7: review time and coordinate axis in chapter 2.5.3
Derrick checks that this chapter is compatible with OOI practices.

Action 8: revisit chapter 2.4 “Global attributes”, should be compliant with the new ACDD version 2.
Justin

Action 9: history section is recommended, but not mandatory on §2.5.6
Thierry

Action 10: Bart will apply the QC tests on the Ifremer Groom data files available on: XXX
Bart will push the QC script on the Groom SVN (XXX)

1 EGO gliders data-management principles

1.1 About EGO

Everyone's Gliding Observatories - EGO is dedicated to the promotion of the glider technology and its applications.

The EGO group promotes glider applications through coordination, training, liaison between providers and users, advocacy, and provision of expert advice.

We intend to favor oceanographic experiments and the operational monitoring of the oceans with gliders through scientific and international collaboration. We provide news, support, information about glider projects and glider data management, as well as resources related to gliders.

All EGO data are publicly available. More information about the project is available at: http://www.ego-network.org

1.2 About this document

This document specifies the NetCDF file format of EGO-gliders that is used to distribute glider data, metadata and technical data. It documents the standards used therein; this includes naming conventions as well as metadata content.

It was initiated in October 2012, based on OceanSITES, Argo and ANFOG user's manuals.

1.3 EGO data management structure and data access

The data flow within EGO is carried out through three organizational units: PIs, DACs and GDACs.

The **Principal Investigator (PI)**, typically a scientist at a research institution, maintains the observing platform and the sensors that deliver the data. He or she is responsible for providing the data and all auxiliary information to a **Data Assembly Center (DAC)**.

The **DAC** assembles EGO-compliant files from this information and delivers these to the two **Global Data Assembly Centers (GDACs)**, where they are made publicly available.

The **GDAC** distributes the best copy of the data files. When a higher quality data file (e.g. calibrated data) is available, it replaces the previous version of the data file.
The user can access the data at either GDAC, cf. section "GDAC organization".

1.4 User Obligations

A user of EGO data is expected to read and understand this manual and the documentation about the data as contained in the "attributes" of the NetCDF data files, as these contain essential information about data quality and accuracy.

A user of EGO data must comply with the requirements set forth in the attributes "distribution_statement" and "citation" of the NetCDF data files.

Unless stated otherwise, a user must acknowledge use of EGO data in all publications and products where such data are used, preferably with the following standard sentence:

"These data were collected and made freely available by the international EGO project and the national programs that contribute to it."

1.5 Disclaimer

EGO data are published without any warranty, express or implied.

The user assumes all risk arising from his/her use of EGO data.

EGO data are intended to be research-quality and include estimates of data quality and accuracy, but it is possible that these estimates or the data themselves contain errors.

It is the sole responsibility of the user to assess if the data are appropriate for his/her use, and to interpret the data, data quality, and data accuracy accordingly.

EGO welcomes users to ask questions and report problems to the contact addresses listed in the data files or on the EGO internet page.

1.6 Further Information Sources and Contact Information

- EGO website: http://www.ego-network.org
- For further information about the benefits and distributing data onto the GTS, please refer to: http://www.jcommops.org/dbcp/gts or contact the EGO Project Office on projectoffice@EGO.org.
- For information about unique numbering of EGO Gliders and Gliders on the GTS see: http://www.wmo.int/pages/prog/amp/mmop/wmo-number-rules.html

1.7 Useful links, tools

1.7.1 EGO matlab scripts to generate NetCDF files

These scripts, ideally one per glider type will be freely available.

1.7.2 EGO file format checker

The EGO file format checker is a java software freely available at:

http://projets.ifremer.fr/coriolis/Observing-the-ocean/Observing-system-networks/EGO/Access-to-data

1.8 Open issues, known problems

1.8.1 Warning on CTD thermal lag errors

The glider CTD data may be affected by a thermal lag error problem. In area with strong thermal gradient such as in the thermocline, the salinity may be overestimated.

This problem is not detected by the existing real-time quality controls.

A new real-time QC test is under study to flag as probably bad (but correctable) the salinity data affected by this problem.

In parallel, method to correct this problem is under study.

1.8.2 Time sampling issues on Slocum gliders

Slocum gliders are equipped with 2 central units (CPU): a navigation CPU and a scientific CPU (to manage sensors).

Observations are reported either from the navigation CPU or from the scientific CPU.

Observations are time stamped with the CPU clock.

Both clocks are not necessarily synchronized.

The time difference between navigation and scientific CPU is difficult to manage.

A time difference may result in observations with identical time stamps.

These problems may be identified in level 0 data files where no correction on time reference is performed.

In level 1 data files, these duplicate time samples are removed.

2 EGO glider NetCDF data format version 1.1

2.1 NetCDF standards and conventions

EGO uses the NetCDF (network Common Data Form) system, a set of software libraries and machine-independent data formats. Our implementation of NetCDF is based on the community-supported Climate and Forecast (CF) specification, which supplies a standard vocabulary and some metadata conventions.

EGO layers several more conventions above the CF standard. These are intended to make it easier to share in-situ data, to make it simpler for the GDACs to aggregate data from multiple sites, and to ensure that the data can be created and understood by the basic NetCDF utilities.

- EGO includes standard terms for the short name of both coordinate and data variables (measurements).
- File names are created using a standard, described in section 5.1.

An EGO data file contains measurements such as temperature and salinity, continuously performed at different levels on a glider as well as other recorded parameters, derived variables and complete location, time, and provenance information.

The requirements are drawn almost exclusively from the NetCDF Style Guide:

- Units are compliant with CF/COARDS/Udunits;
- The time parameter is encoded as recommended by COARDS and CF;
- Parameters are given standard names from the CF table;
- Where time is specified as an attribute, the ISO8601 standard is used.

For more information on NetCDF, Udunits, COARDS, CF and ISO8601 see:

- NetCDF: http://www.unidata.ucar.edu/software/netcdf/docs/BestPractices.html
- Udunits: http://www.unidata.ucar.edu/software/udunits/
- COARDS: http://www.ferret.noaa.gov/noaa_coop/coop_cdf_profile.html
- CF: http://cf-pcmdi.llnl.gov/
- ISO8601: http://en.wikipedia.org/wiki/ISO_8601

Note on format version

In October 2012, the EGO data format version **1.0** was released.

In July 2013, the first data files in version **1.1** were distributed.
The User’s manual may be updated with clarifications, recommendations, additional optional attributes without changing the data format version.

2.2 Trajectory and profile data formats

The data, metadata and technical data from an EGO glider deployment are recorded in a trajectory file.

From the trajectory file, vertical profiles of ocean parameters can be extracted into profile files. The profile files are a product derived from the core trajectory file.

2.3 Data file dimensions

Name	Example	Comment
EVENT_SCIENCE	EVENT_SCIENCE=unlimited	Number of events recorded from the scientific board.
EVENT_NAVIGATION	EVENT_NAVIGATION=1547	Number of events recorded from the scientific board.
EVENT_GPS	EVENT_GPS=358	Number of events recorded from the scientific board.
DATE_TIME	DATE_TIME = 14;	This dimension is the length of an ASCII date and time value. Date_time convention is : YYYYMMDDHHMISS YYYY : year MM : month DD : day HH : hour of the day (as 0 to 23) MI : minutes (as 0 to 59) SS : seconds (as 0 to 59) Date and time values are always in universal time coordinates (UTC). Examples : 20010105172834 : January 5th 2001 17:28:34 19971217000000 : December 17th 1997 00:00:00
STRING1024 STRING256 STRING64 STRING32 STRING16 STRING8 STRING4 STRING2	STRING1024 = 1024; STRING256 = 256; STRING64 = 64; STRING32 = 32; STRING16 = 16; STRING8 = 8; STRING4 = 4; STRING2 = 2;	String dimensions from 2 to 1024.

2.4 Global attributes

The global attribute section of a NetCDF file contains metadata that describes the contents of the file overall, and allows for data discovery. All fields should be human-readable, and should be of character type, not numeric, even if the information content is a number. EGO recommends that all of these attributes be used and contain meaningful information unless there are technical reasons rendering this impossible. However, files that do not at least contain the attributes listed as "mandatory" will not be considered EGO-compliant. In EGO, global attribute names are in lower-case letters (except "Convention").

Global attributes can be thought of as conveying five kinds of information:

- What: what are the data in this dataset;
- Where: the spatial coverage of the data;
- When: the temporal coverage of the data;
- Who: who produced the data;
- How: how were the data produced and made available.

The global attributes specification follows the recommendations of Unidata NetCDF Attribute Convention for Dataset Discovery (ACDD), at:

~~http://www.unidata.ucar.edu/software/netcdf-java/formats/DataDiscoveryAttConvention.html~~

http://wiki.esipfed.org/index.php/Attribute_Convention_for_Data_Discovery_(ACDD)

Name	Example	definition	ACDD compliant
WHAT			
data_type	data_type=" EGO glider time-series data"	Type of data contained in the file. The list of acceptable data types is in reference table 1. Example: "EGO glider time-series data" **This attribute is mandatory.**	
format_version	format_version="1.0"	File format version Example: "0.9". **This attribute is mandatory.**	yes
platform_code	platform_code="pytheas"	Glider unique code within EGO project. The use of lower case is recommended. Only basic ASCII letters, numbers and "_" (no accent). **This attribute is mandatory.**	
date_modified	date_modified="2006-04-11T08:35:00Z"	File update or creation date (UTC). See note on time format below. **This attribute is mandatory.**	yes
wmo_platform_code	wmo_platform_code=" 61864"	WMO (World Meteorological Organization) identifier. This platform number is unique within the EGO project. Example: "61864" for pytheas glider.	
ices_platform_code	ices_platform_code="FFPY"	Platform code assigned by ICES (International Council for the Exploration of the Sea).	yes
source	source="Glider observation"	Method of production of the original data. For EGO data, use one of the following: "Shipborne observation", "Glider observation"	yes

history	history= "2008-12-10T09:35:36Z Written by MATLAB script seagliderFV.m v1.2 2010-12-07T10:11:00Z data calibrated, controlled and sent to DAC, Laurent Mortier"	Audit trail for modifications to the original data. It should contain a separate line for each modification, with each line beginning with a timestamp, and including user name, modification name, and modification arguments. The time stamp should follow the format outlined in the note on time formats below.	yes
data_mode	data_mode="R"	Indicates if the file contains real-time, provisional, mixed or delayed-mode data. The list of valid data modes is in reference table 19. **This attribute is mandatory.**	yes
quality_index	quality_index="excellent"	Code value valid for the whole dataset: "unknown quality" "excellent" (no known problems, regular quality checking) "probably good" (occasional problems, validation phase) "extremely suspect" (frequent problems)	No
references	references="http://www.ego-network.org/"	Published or web-based references that describe the data or methods used to produce it. Include a reference to EGO and a project-specific reference if appropriate.	No
comment	comment="This deployment was performed during the Latex exercise"	Miscellaneous information about the data or methods used to produce it. Any free-format text is appropriate.	yes
Conventions	Conventions="CF-1.6 EGO-1.1	Name of the conventions followed by the dataset.	yes
netcdf_version	netcdf_version="3.6"	NetCDF version used for the data set.	No
title	title="Pytheas glider data on Latex deployment"	Free-format text describing the dataset. The display of these two attributes together should allow data discovery for a human reader. "title": title of the dataset. Use the file name if in doubt.	Yes
summary	summary="Oceanographic glider data from Pytheas glider deployed in gulf of Lion, North-West Mediterranean sea, in 2010. Measured properties: temperature, salinity, oxygen, turbdity."	"summary": a longer description of the dataset. A paragraph of up to 100 words is appropriate.	yes

abstract	abstract="Glider Ocean observations have been collected by EGO since 2005 and are ongoing. EGO is Everyone's Gliding Observatory. The data are Slocum, SeaGlider or Spray gliders fitted with a wide range of sensors measuring temperature, salinity, oxygen, currents, chlorophyll, nitrate, cdom and other bio-geo-chemical data. This NetCDF file was created by EGO using the EGO filenaming convention version 1 and the EGO NetCDF user's manual version 1.0."	Paragraph describing the dataset: type of data contained, how it was created, who collected it, what instruments were used, what data formatting convention was used,etc.	No
keywords	keywords="Turbidity, Chlorophyll, Organic Matter, Oxygen, Fluorescence, Scattering, Water Temperature, Conductivity, Salinity"	Comma separated list of key words and phrases.	yes
naming_authority id	naming_authority="EGO" id="GL_20100612_PYTHEAS_MooseT00_09_R"	The "id" and "naming_authority" attributes are intended to provide a globally unique identification for each dataset. For EGO data, use: naming_authority="EGO" and id=file name (without .nc suffix), which is designed to be unique. **Both attributes are mandatory.**	yes
cdm_data_type	cdm_data_type="Trajectory"	The "cdm_data_type" attribute gives the Unidata CDM (Common Data Model) data type used by THREDDS. E.g. "Point", "Trajectory", "Station", "Radial", "Grid", "Swath". More: http://www.unidata.ucar.edu/projects/THREDDS/CDM/CDM-TDS.htm	yes
WHERE			
area	area="North West Mediterranean Sea"	Geographical coverage Use vocabulary from SeaDataNet sea areas (C16). http://seadatanet.maris2.nl/v_bodc_vocab/search.asp?name=(C16)%20SeaDataNet+sea+areas&l=C16	yes
geospatial_lat_min	geospatial_lat_min="59.8"	Southernmost valid latitude, a value between -90 and 90 degrees. This is calculated from the valid latitudes in the file. Decimal degrees	Yes
geospatial_lat_max	geospatial_lat_max="62.9"	Northernmost valid latitude, a value between -90 and 90 decimal degrees. This is calculated from the valid latitudes in the file. Decimal degrees	Yes

geospatial_lon_min	geospatial_lon_min="-41.2"	Westernmost valid longitude, a value between -180 and 180 degrees. This is calculated from the valid longitudes in the file. Decimal degrees	Yes
geospatial_lon_max	geospatial_lon_max="-20.1"	Easternmost valid longitude, a value between -180 and 180 decimal degrees. This is calculated from the valid longitudes in the file. Decimal degrees	Yes
geospatial_vertical_min	geospatial_vertical_min="10.0"	Minimum valid depth or pressure for measurements in the file.	Yes
geospatial_vertical_max	geospatial_vertical_max="200"	Maximum valid depth or pressure for measurements in the file.	Yes
WHEN			
time_coverage_start	time_coverage_start="2010-07-01T00:00:00Z"	Start date of the data in UTC. See note on time format below.	Yes
time_coverage_end	time_coverage_end="2010-09-18T23:59:29Z"	Final date of the data in UTC. See note on time format below.	Yes
WHO			
Institution	institution="CNRS-LOCEAN"	Institution where the original data was produced.	Yes
institution_references	institution_references="http://www.nocs.uk"	References to data provider institution, the place to find all information on the dataset (web-based, i.e. give URLs).	No
sdn_edmo_code	sdn_edmo_code="1042"	SeaDataNet EDMO code of the institution. EDMO is the "European Directory of Marine Organisations". http://seadatanet.maris2.nl/edmo/	No
contact	contact="laurent.beguery@dt.insu.cnrs.fr"	Contact person's e-mail.	
creator	creator="Thierry Carval"	Name of the person responsible for the creation of the dataset.	yes
data_assembly_center	data_assembly_center="IF"	Data Assembly Center (DAC) in charge of this data file. The data_assembly_center are listed in reference table 4.	yes
contributor	principal_investigator="Laurent Mortier"	Name of the principal investigator in charge of the glider project.	yes
contributor_email	contributor_email="Laurent.Mortier@upmc.fr"	Principal investigator's email address.	no
Contributor_role	Contributor_role="principle_investigator"	The name and role of any individuals or institutions that contributed to the creation of this data.	yes
observatory	observatory = "North west Mediterranean sea"	A geographical area monitored with a fleet of gliders.	No, glider specific
deployment_code	deployment_code="MooseT00_19"	Deployment code. It is unique among EGO deployments. This code may be used as the local code in catalogues such as SeaDataNet Common Data Index (CDI).	No, glider specific

deployment_label	deployment_label="Moose T00_19 summer 2010 deployment"	The deployment label, a free text to describe the deployment.	No, glider specific
HOW			
license	license="Follows CLIVAR (Climate Variability and Predictability) standards, cf. http://www.clivar.org/data/data_policy.php. Data available free of charge. User assumes all risk for use of data. User must display citation in any publication or product using data. User must contact PI prior to any commercial use of data."	Statement describing data distribution policy. EGO has adopted the CLIVAR data policy, which explicitly calls for free and unrestricted data exchange.	yes
citation	citation="These data were collected and made freely available by the international EGO project and the national programs that contribute to it."	The citation to be used in publications using the dataset.	yes
update_interval	update_interval="daily"	Update interval for the file, one of the following: "hourly", "daily", "yearly", "void". Use "void" for delayed-mode or archive data that do not need continuous updating.	Yes
qc_manual	qc_manual="http://DOI.xxx"	Contains the name of the manual that describes the quality control procedure. As of now, there is no separate QC manual, so the user's manual is the appropriate reference.	yes

Note on time formats

Whenever time information is given in the global attributes, it ought to be a string of the format:

"YYYY-MM-DDThh:mm:ssZ"   (i.e. year - month - day T hour : minute : second Z)

If higher resolution than seconds is needed, any number of decimal digits (".s") for the seconds is acceptable:

"YYYY-MM-DDThh:mm:ss.sZ"

In any case, the time must be in UTC. A capital "T" separates the date and the hour information. The string must end with a capital "Z", an old indication of UTC. These formats are two (of many) described by ISO8601.

Examples:

- 2005-10-24T08:00:00Z
- 2008-01-01T22:50:02.031Z

2.5 Variables

NetCDF variables include data measured by instruments, parameters derived from the primary measurements, and coordinate variables, which may be nominal values, such as values for depth for instruments that do not directly record depth. The variable names are written in CAPITALIZED letters. Each variable has a specific set of attributes, some of which are mandatory.

The mandatory variables or attributes are in **bold characters**.

2.5.1 Coordinate variables

The coordinate variables orient the data in time and space. For this purpose, they have an "axis" attribute defining that they point in X, Y, Z, and T dimensions.

~~Default values are not allowed in coordinate variables.~~

All attributes in this section except the "comment" are mandatory.

Time axis

The time axis is the main dimension for an EGO glider data file.

There are typically 3 time sources on a glider:

- The science board
- The navigation board
- The GPS clock

Data are recorded along one of these time sources (the CTD observations times are from the science board clock, the technical data times are from the navigation board, the GPS location times are from the GPS board).

The events are recorded along one of these 3 time axes.

The data from the science board are recorded along the EVENT_SCIENCE time axis.

The data form the navigation board are recorded along the EVENT_NAVIGATION time axis.

The data from the GPS board are recorded along the EVENT_GPS time axis.

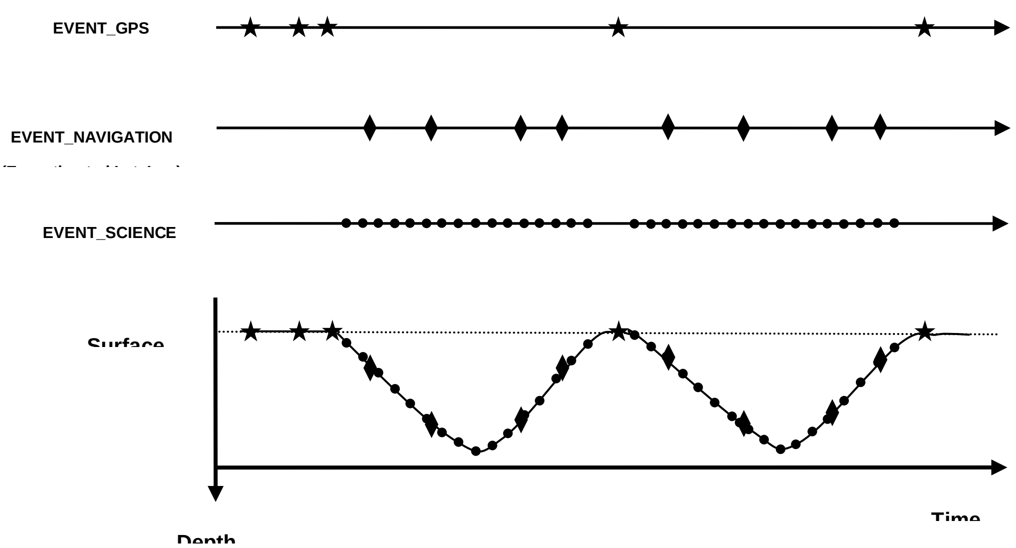

Figure 1: example of events associated to each of the 3 glider time sources

Type, name, dimension, attributes	Comment
double **TIME_SCIENCE**(EVENT_SCIENCE); TIME:long_name = "time from science board"; TIME:standard_name = "time"; TIME:units = "seconds since 1970-01-01T00:00:00Z"; TIME:valid_min = 0.0; TIME:valid_max = 90000.0; TIME:comment = "Optional comment..."; TIME:axis = "T"; TIME:ancillary_variable = "TIME_QC"; TIME:sdn_parameter_urn = "SDN:P01::XXX"; TIME:sdn_uom_urn = "SDN:P061::UTBB";	Time of the measurement in seconds since noon, 1970-01-01. Example: July 25, 2001, 19:14:00 is stored as 996088440.
double **JULD_SCIENCE**(EVENT_SCIENCE); JULD:long_name = "time"; JULD:standard_name = "time"; JULD:units = "days since 1950-01-01T00:00:00Z"; JULD:valid_min = 0.0; JULD:valid_max = 90000.0; JULD:comment = "Optional comment..."; JULD:axis = "T"; JULD:ancillary_variable = "JULD_QC"; JULD:sdn_parameter_urn = "SDN:P01::XXX"; JULD:sdn_uom_urn = "SDN:P061::UTAA";	Date and time (UTC) of the measurement in days since midnight, 1950-01-01. JULD is a duplication of TIME expressed in Julian day. JULD is used for interoperability with other groups such as Argo. Example: July 25, 2001, 19:14:00 is stored as 18833.8013889885.
float **LATITUDE_SCIENCE**(EVENT_SCIENCE); LATITUDE:long_name = "Latitude of each location"; LATITUDE:standard_name = "latitude"; LATITUDE:units = "degrees_north"; LATITUDE:_FillValue = 99999.0; LATITUDE:valid_min = -90.0; LATITUDE:valid_max = 90.0; LATITUDE:comment = "Optional comment..."; LATITUDE:axis="Y"; LATITUDE:ancillary_variable = "POSITION_QC"; LATITUDE:reference="WGS84"; LATITUDE:coordinate_reference_frame="urn:ogc:crs:EPSG::4326"; LATITUDE:sdn_parameter_urn = "SDN:P01::ALATZZ01"; LATITUDE:sdn_uom_urn = "SDN:P061::DEGN";	Latitude of the measurements. Units: degrees north; southern latitudes are negative. Example: 44.4991 for 44° 29' 56.76'' N

float **LONGITUDE_SCIENCE**(EVENT_SCIENCE); LONGITUDE:long_name = “Longitude of each location”; LONGITUDE:standard_name = “longitude”; LONGITUDE:units = “degrees_east”; LONGITUDE:_FillValue = 99999.0; LONGITUDE:valid_min = -180.0; LONGITUDE:valid_max = 180.0; LONGITUDE:comment = “Optional comment...”; LONGITUDE:axis=”X”; LONGITUDE:ancillary_variable = “POSITION_QC”; LONGITUDE:reference="WGS84"; LONGITUDE:coordinate_reference_frame="urn:ogc:crs:EPSG::4326"; LONGITUDE:sdn_parameter_urn = “SDN:P01::ALONZZ01”; LONGITUDE:sdn_uom_urn = “SDN:P061::DEGE”;	Longitude of the measurements. Unit: degrees east; western latitudes are negative. Example: 16.7222 for 16° 43’ 19.92’’ E

Note on latitude and longitude WGS84 datum

The latitude and longitude datum is WGS84. This is the default output of GPS systems.

EGO uses the EPSG coordinate reference system to describe geographical positions; the coordinate reference frame corresponding to WGS84 is : "urn:ogc:crs:EPSG::4326".

More on EPSG : http://www.epsg.org/

Note on TIME variables

~~By default, the time word represents the center of the data sample or averaging period.~~

The TIME_SCIENCE, TIME_NAVIGATION and TIME_GPS variables are recorded from different clock sources: the scientific board, the navigation board, the GPS board.

The "glider_original_parameter_name" attribute identifies precisely the origin of each time value.

TIME_NAVIGATION glider_original_parameter_name = "m_present_secs_into_mission"

TIME_SCIENCE glider_original_parameter_name = "m_present_time"

TIME_GPS glider_original_parameter_name = "m_gps_time"

2.5.2 Coordinate quality control variables

The coordinate variables have the same quality control variables as the data variables. If the quality control values are constant, the information is given in attributes of the coordinate variables. For details, see <PARAM>_QC in the section on data variables, and the note on quality control therein.

Type, name, dimension, attributes	Comment
byte **TIME_QC**(EVENT_GPS); TIME_QC:long_name = “quality flag”; TIME_QC:standard_name = “time status_flag”; TIME_QC:conventions = “EGO reference table 2”; TIME_QC:_FillValue = -128b; TIME_QC:flag_values = 0b, 1b, 2b, 3b, 4b, 5b, 8b, 9b; TIME_QC:flag_meanings = “no_qc_performed good_data probably_good_data bad_data_that_are_potentially_correctable bad_data value_changed interpolated_value missing_value”	Quality flag for each TIME value.

byte **JULD_QC**(EVENT_GPS); JULD_QC:long_name = “quality flag”; JULD_QC:standard_name = “time status_flag”; JULD_QC:conventions = “EGO reference table 2”; JULD_QC:_FillValue = -128b; JULD_QC:flag_values = 0b, 1b, 2b, 3b, 4b, 5b, 8b, 9b; JULD_QC:flag_meanings = “no_qc_performed good_data probably_good_data bad_data_that_are_potentially_correctable bad_data value_changed interpolated_value missing_value”	Quality flag for each JULD value. This QC flag is a duplication of TIME_QC.
byte **POSITION_QC**(EVENT_GPS); POSITION_QC:long_name = “quality flag”; POSITION_QC:standard_name = "latitude longitude status_flag”; POSITION_QC:conventions = “EGO reference table 2”; POSITION_QC:_FillValue = -128b; POSITION_QC:flag_values = 0b, 1b, 2b, 3b, 4b, 5b, 8b, 9b; POSITION_QC:flag_meanings = “no_qc_performed good_data probably_good_data bad_data_that_are_potentially_correctable bad_data value_changed interpolated_value missing_value”;	Quality flag for each LATITUDE and LONGITUDE value.

2.5.3 Profile, phase and direction management

A glider regularly performs various phases such as surface, descent, inflexion, subsurface drift.

For each time stamp, the following variables indicate the phase of the glider at that time.

During ascent or descent phase, the glider performs vertical profiles.

A number is associated with each phase.

The first phase of the deployment is number 0: it is the surface drift before the first dive.

The phase number is increased by 1 for each new phase.

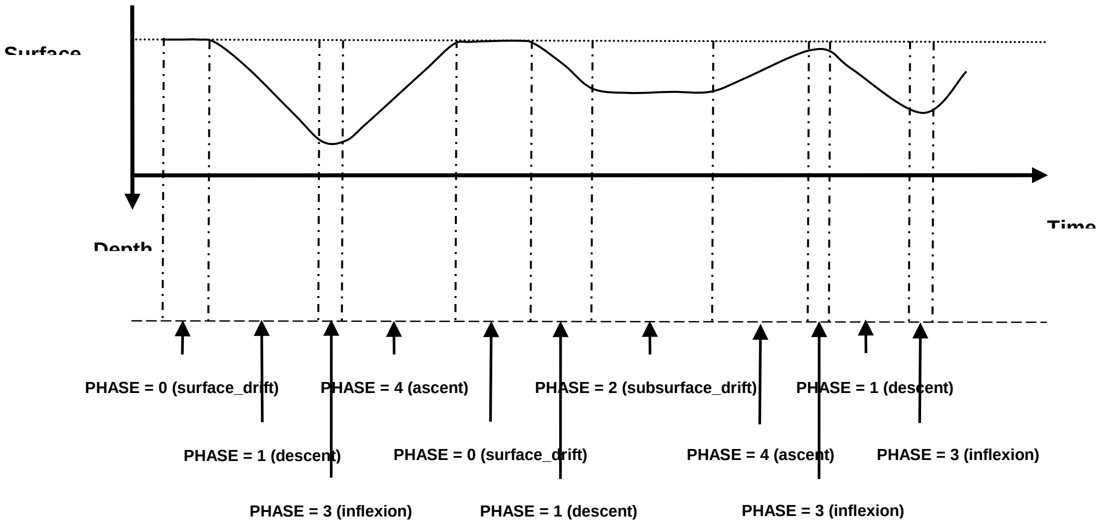

Figure 2: example of PHASE and PHASE_NUMBER assignment

Type, name, dimension, attributes	Comment
byte PHASE(EVENT_SCIENCE); PHASE:long_name = “Glider trajectory phase code”; PHASE:conventions = “EGO reference tabe 9”; PHASE:_FillValue = -128; PHASE:flag_values = 0b, 1b, 2b, 3b, 4b, 5b, 6b; PHASE:flag_meanings = “surface_drift descent subsurface_drift inflexion ascent grounded inconsistent”	Phase of the trajectory at that time, described in reference table 9.
int PHASE_NUMBER(EVENT_SCIENCE); PHASE_NUMBER:long_name = “Glider trajectory phase number”; PHASE_NUMBER:_FillValue = 99999;	A number associated with the phase. The first phase number is 0. The phase number is increased at each phase change.

2.5.4 Positioning method

The positions reported in variables latitude and longitude are reported from various sources: GPS, Argos, interpolation.

Type, name, dimension, attributes	Comment
byte POSITIONING_METHOD(EVENT_GPS); POSITIONING_METHOD:long_name = “positioning method”; POSITIONING_METHOD:_FillValue = -128; POSITIONING_METHOD:conventions = “EGO reference tabe 10”; POSITIONING_METHOD:flag_values = 0, 1, 2; POSITIONING_METHOD:flag_meanings = “gps argos interpolated”	Positioning method at that time, described in reference table 10.

2.5.5 Data variables

Data variables contain the actual measurements and indicators about their quality, uncertainty, and mode through which they were obtained. There are different options as to how the indicators are specified, whether in attributes or separate variables, which are outlined in the notes below the table.

The variable names are standardized in reference table 3; replace <PARAM> with any of the names indicated there. Mandatory attributes are marked as such, however, EGO requests that all other attributes be used and contain meaningful information unless technical reasons make this impossible.

When a given variable <PARAM> is available in different forms, a suffix _XXX discriminate the different forms of the parameter.

Example: to discriminate a raw and an adjusted salinity use the two variables PSAL (raw salinity) and PSAL_ADJUSTED (adjusted salinity).

Type, name, dimension, attributes	Comment
float **<PARAM>(**EVENT_SCIENCE); <PARAM>:**standard_name** = "<X>"; <PARAM>:**units** = "<Y>"; <PARAM>:**_FillValue** = <Y>; <PARAM>:long_name = "Y"; <PARAM>:valid_min = <Y>; <PARAM>:valid_max = <Y>; <PARAM>:comment = "<Y>"; ~~<PARAM>:sensor_mount = <X>;~~ ~~<PARAM>:sensor_orientation = <X>;~~ ~~<PARAM>:sensor_name = <Y>;~~ ~~<PARAM>:sensor_serial_number = <Y>;~~ <PARAM>:ancillary_variables = "<Y>"; <PARAM>:accuracy = <Y>; <PARAM>:precision = <Y>; <PARAM>:resolution = <Y>; <PARAM>: cell_methods = "<X>"; <PARAM>:DM_indicator = "<X>"; <PARAM>:reference_scale = "<Y>"; <PARAM>:coordinates = "TIME LATITUDE LONGITUDE PRES"; <PARAM>:sdn_parameter_urn = "XXX"; <PARAM>:sdn_uom_urn = "XXX"; <PARAM>:sdn_uom_name = "XXX"; <PARAM>:glider_original_parameter_name = "XXX";	The dimension may be EVENT_SCIENCE or EVENT_NAVIGATION <PARAM> names are defined in reference table 3. Examples: PRES, TEMP, PSAL, DOXY. These attributes are mandatory: units and _FillValue. If a standard_name exists for this variable, it is mandatory. These 8 attributes are highly desirables: valid_min, valid_max, sensor_name, sensor_serial_number accuracy, precision, resolution, DM_indicator. The other attributes are optional. <X> : standardized attributes listed in reference tables in chapter 3 <Y> : attributes whose value is set by the PI (Principal Investigator) **standard_name**: type char, see reference. table 3 **units**: type char, see reference table 3 **_FillValue**: type float, see reference table 3 **long_name**: type char, free text **valid_min**: type float. Minimum value for valid data for this dataset **valid_max**: type float. Maximum value for valid data for this dataset **comment**. type char. Any free-format text with comments as appropriate. ~~**sensor_mount** type char. See reference table 20 for sensor mounting characteristics.~~ ~~**sensor_orientation** type char. See reference table 21 for sensor orientation characteristics.~~ ~~**sensor_name** type char (if the data all come from a single sensor).~~ ~~**sensor_serial_number** type char (if the data all come from a single sensor).~~ **ancillary_variables**. type char. Other variables associated with <PARAM>, e.g. <PARAM>_QC. List as space-separated string.

	Example: TEMP:ancillary_variables=”TEMP_QC TEMP_DM TEMP_UNCERTAINTY” **accuracy**: type float. Nominal sensor accuracy. Cf. note on uncertainty below. **precision**: type float. Nominal sensor precision. Cf. note on uncertainty below. **resolution**: type float. Nominal resolution of this data parameter. **cell_methods**: type char. Specifies cell method as per CF convention. Example: TEMP:cell_methods=”TIME: mean” Values are listed in table 2.2 The boundary of the cells are described in the axis bound attribute (see CF convention for cell boundary, cell measure and cell method) **DM_indicator**: Type char. Data mode, if constant, as per reference table 19. Cf. note on data modes below. **reference_scale**: type char. For some measurements that are provided according to a standard reference scale specify the reference scale with this optional attribute. Example: ITS-90, PSS-78 **coordinates** : list of the coordinate variables **sdn_parameter_urn**: SeaDataNet parameter code **sdn_uom_urn**: SeaDataNet unit code **glider_original_parameter_name**: the parameter name as originally reported by the glider
byte **<PARAM>_QC**(EVENT); <PARAM>_QC:**long_name** = “quality flag”; <PARAM>_QC:**standard_name** = “*standard_name* status_flag”; <PARAM>_QC:**conventions** = “EGO reference table 2”; <PARAM>_QC:**_FillValue** = -128b; <PARAM>_QC:**flag_values** = 0b, 1b, 2b, 3b, 4b, 5b, 8b, 9b; <PARAM>_QC:**flag_meanings** = “no_qc_performed good_data probably_good_data bad_data_that_are_potentially_correctable bad_data value_changed interpolated_value missing_value”	Quality flags for values of associated <PARAM>. The flag scale is specified in reference table 2, and is included in the flag_meanings attribute.
float **<PARAM>_UNCERTAINTY**(TIME); <PARAM>_UNCERTAINTY:**long_name** = “uncertainty” <PARAM>_UNCERTAINTY**:_FillValue** = <X> <PARAM>:**units** = “<Y>”;	Overall uncertainty of the data given in <PARAM>. See note on uncertainty below.

Note on uncertainty, accuracy, resolution and precision

If the overall measurement uncertainty for a variable <PARAM> is reasonably well-known, it must be provided in a variable of its own, <PARAM>_UNCERTAINTY.

If it is impossible to estimate the measurement uncertainty, it is required to define at least the attribute <PARAM>:accuracy with the nominal sensor accuracy.

The attributes <PARAM>:precision and <PARAM>:resolution are optional; they contain the sensor precision and resolution if known.

Note vertical axis

The PRES variable is usually a vertical axis. Its axis attribute is "Z" : PRES:axis="Z".

It has a "positive" mandatory attribute set to "down".

Example for sea temperature measurements and associated quality flags

```
float TEMP(EVENT);
TEMP:standard_name = "sea_water_temperature";
TEMP:units = "degree_Celsius";
TEMP:_FillValue = 99999.f;
TEMP:long_name = "sea water temperature in-situ ITS-90 scale";
TEMP:valid_min = -2.0f;
TEMP:valid_max = 40.f;
TEMP:comment = "";
TEMP:sensor_depth = 1;
TEMP:sensor_mount = "mounted_on_glider_tail";
TEMP:sensor_name = "SBE41";
TEMP:sensor_serial_number = "3263";
TEMP:ancillary_variables = "TEMP_QC" ;
TEMP:accuracy = 0.01f;
TEMP:precision = 0.01f;
TEMP:resolution = 0.001f;
TEMP:cell_methods="median";
TEMP:DM_indicator="P";
TEMP:reference_scale = "ITS-90";

byte TEMP_QC(EVENT);
TEMP_QC:long_name = "quality flag";
TEMP_QC:conventions = "EGO reference table 2";
TEMP_QC:_FillValue = -128b;
TEMP_QC:flag_values = 0b, 1b, 2b, 3b, 4b, 5b, 8b, 9b;
TEMP_QC:flag_meanings = "no_qc_performed good_data probably_good_data
bad_data_that_are_potentially_correctable bad_data value_changed interpolated_value missing_value"
```

2.5.6 History information

These dimensions are specific to history section.

The history section is not mandatory, but is recommended.

Name	Definition	Comment
N_HISTORY	N_HISTORY =<int value>;	Number of history records.

This section contains history information for each action performed on the time-series by a data centre.

Each item of this section has a N_HISTORY (number of history records) dimension.

A history record is created whenever an action is performed on a part of the time-series definied by history_start_time and history_stop_time.

The recorded actions are coded and described in the history code table from the reference table 7.

Name	Definition	Comment
HISTORY_INSTITUTION	char HISTORY_INSTITUTION(N_HISTORY, STRING2); HISTORY_INSTITUTION:long_name = "Institution which performed action"; HISTORY_INSTITUTION:conventions = "EGO reference table 4"; HISTORY_INSTITUTION:_FillValue = " ";	Institution that performed the action. Institution codes are described in reference table 4. Example : BO for BODC
HISTORY_STEP	char HISTORY_STEP(N_HISTORY, STRING4); HISTORY_STEP:long_name = "Step in data processing"; HISTORY_STEP:conventions = "EGO reference table 12"; HISTORY_STEP:_FillValue = " ";	Code of the step in data processing for this history record. The step codes are described in reference table 12. Example : ARGQ : Automatic QC of data reported in real-time has been performed
HISTORY_SOFTWARE	char HISTORY_SOFTWARE(N_HISTORY, STRING8); HISTORY_SOFTWARE:long_name = "Name of software which performed action"; HISTORY_SOFTWARE:conventions = "Institution dependent"; HISTORY_SOFTWARE:_FillValue = " ";	Name of the software that performed the action. This code is institution dependent. Example : WJO
HISTORY_SOFTWARE_RELEASE	char HISTORY_SOFTWARE_RELEASE(N_HISTORY, STRING4); HISTORY_SOFTWARE_RELEASE:long_name = "Version/release of software which performed action"; HISTORY_SOFTWARE_RELEASE:conventions = "Institution dependent"; HISTORY_SOFTWARE_RELEASE:_FillValue = " ";	Version of the software. This name is institution dependent. Example : «1.0»

HISTORY_REFERENCE	char HISTORY_REFERENCE(N_HISTORY, STRING64); HISTORY_REFERENCE:long_name = "Reference of database"; HISTORY_REFERENCE:conventions = "Institution dependent"; HISTORY_REFERENCE:_FillValue = " ";	Code of the reference database used for quality control in conjunction with the software. This code is institution dependent. Example : WOD2001
HISTORY_DATE	char HISTORY_DATE(N_HISTORY, DATE_TIME); HISTORY_DATE:long_name = "Date the history record was created"; HISTORY_DATE:conventions = "YYYYMMDDHHMISS"; HISTORY_DATE:_FillValue = " ";	Date of the action. Example : 20011217160057
HISTORY_ACTION	char HISTORY_ACTION(N_HISTORY, STRING64); HISTORY_ACTION:long_name = "Action performed on data"; HISTORY_ACTION:conventions = "EGO reference table 7"; HISTORY_ACTION:_FillValue = " ";	Name of the action. The action codes are described in reference table 7. Example : QCF$ for QC failed
HISTORY_PARAMETER	char HISTORY_PARAMETER(N_HISTORY, STRING16); HISTORY_PARAMETER:long_name = "Parameter action is performed on"; HISTORY_PARAMETER:conventions = "EGO reference table 3"; HISTORY_PARAMETER:_FillValue = " ";	Name of the parameter on which the action is performed. Example : PSAL
HISTORY_PREVIOUS_VALUE	float HISTORY_PREVIOUS_VALUE(N_HISTORY); HISTORY_PREVIOUS_VALUE:long_name = "Parameter/Flag previous value before action"; HISTORY_PREVIOUS_VALUE:_FillValue = 99999.f;	Parameter or flag of the previous value before action. Example : 2 (probably good) for a flag that was changed to 1 (good)
HISTORY_START_TIME_INDEX	int HISTORY_START_TIME_INDEX(N_HISTORY); HISTORY_START_TIME_INDEX:long_name = "Start time index action applied on"; HISTORY_START_TIME_INDEX:_FillValue = 99999;	Start time index the action is applied to. Example : 100
HISTORY_STOP_TIME_INDEX	int HISTORY_STOP_TIME_INDEX(N_HISTORY); HISTORY_STOP_TIME_INDEX:long_name = "Stop time index action applied on"; HISTORY_STOP_TIME_INDEX:_FillValue = 99999;	Stop time index the action is applied to. Example : 150
HISTORY_QCTEST	char HISTORY_QCTEST(N_HISTORY, STRING16); HISTORY_QCTEST:long_name = "Documentation of tests performed, tests failed (in hex form)"; HISTORY_QCTEST:conventions = "Write tests performed when ACTION=QCP$; tests failed when ACTION=QCF$"; HISTORY_QCTEST:_FillValue = " ";	This field records the tests performed when ACTION is set to QCP$ (qc performed), the test failed when ACTION is set to QCF$ (qc failed). The QCTEST codes are describe in reference table 11. Example : 0A (in hexadecimal form)

The usage of history section is described in §4 "Using the History section of the EGO netCDF Structure".

2.6 Gliders metadata: platform – deployment – instrument - sensor

The glider metadata are recorded in dimensionless integer (containers) whose attributes describe the metadata features.

In a deployment file, there is a metadata container for the platform, the deployment, the instrument and the sensors.

2.6.1 Platform metadata container

This section contains the main characteristics of the glider, stored in the attributes of « platform » meta-data container.

- int platform

Attribute name	comment
TRANS_SYSTEM_ID	Program identifier of the telecommunication subscription. Use N/A when not applicable (eg : Iridium or Orbcomm). Example: 38511 is a program number for all the beacons of an ARGOS customer.
TRANS_FREQUENCY	Frequency of transmission from the glider. Unit : hertz Example : 1/44
POSITIONING_SYSTEM	Position system from reference table 9. ARGOS and GPS are 2 positioning systems. Example : GPS
PLATFORM_FAMILY	Category of instrument. See reference table 22 (§3.15). Example: coastal glider, open ocean glider
PLATFORM_TYPE	Type of glider. See reference table 8. Example: Slocum, Seaglider
PLATFORM_MAKER	Name of the manufacturer. Example : Webb Research Corporation
FIRMWARE_VERSION_NAVIGATION	The firmware version of the navigation controller board.
FIRMWARE_VERSION_SCIENCE	The firmware version of the scientific sensors controller board.
MANUAL_VERSION	The version date or number for the manual for each glider. Example: 110610 or 004
GLIDER_SERIAL_NO	This field should contain only the serial number of the glider. Example: 1679
STANDARD_FORMAT_ID	Standardised format number as described in the reference table online (host site yet to be determined), this table cross references to individual DAC format numbers Example: 1

DAC_FORMAT_ID	Format numbers used by individual DACs to describe each glider type. This is cross-referenced to a standard format id by a reference table online (host site yet to be determined).
WMO_INST_TYPE	Instrument type from WMO code table 1770. A subset of WMO table 1770 is documented in the reference table 8. Example : 846 : Webb Research glider, Seabird sensor
PROJECT_NAME	Name of the project which operates the profiling glider that performed the profile. Example : GYROSCOPE (EU project for EGO program)
DATA_CENTRE	Code of the data centre in charge of the glider data management. The data centre codes are described in the reference table 4. Example: ME for MEDS
PI_NAME	Name of the principal investigator in charge of the glider. Example: Yves Desaubies
ANOMALY	This field describes any anomaly or problem the glider may have had. Example: "the immersion drift is not stable."
BATTERY_TYPE	Describes the type of battery packs in the glider. Example: Alkaline, Lithium or Alkaline and Lithium
BATTERY_PACKS	Describes the configuration of battery packs in the glider, number and type. Example: "4DD Li + 1C Alk"
SPECIAL_FEATURES	Additional glider features can be specified here such as algorithms used by the glider (Ice Sensing Algorithm, Interim Storage Algorithm, grounding avoidance) or additional hardware such as a compressee (buoyancy compensator). Example : "Ice Sensing Algorithm"
GLIDER_OWNER	The owner of the glider (may be different from the data centre and operating institution). Example: "SCRIPPS"
OPERATING_INSTITUTION	The operating institution of the glider (may be different from the glider owner and data centre). Example: "INSU"
CUSTOMIZATION	Free form field to record changes made to the glider after manufacture and before deployment, i.e. this could be the customization institution plus a list of modifications.

2.6.2 Glider deployment information

This section contains the main characteristics of the glider deployment, stored in the attributes of « deployment » meta-data container.

- int deployment

Name	Comment
DEPLOYMENT_START_DATE	Date and time (UTC) of deployment of the glider. Format : YYYYMMDDHHMISS Example : 20011230090500 : December 30th 2001 09:05:00
DEPLOYMENT_START_LATITUDE	Latitude of the deployment. Unit : degree north. Example : 44.4991 : 44° 29' 56.76" N

DEPLOYMENT_START_LONGITUDE	Longitude of the deployment. Unit : degree east Example : 16.7222 : 16° 43' 19.92" E
DEPLOYMENT_START_QC	Quality flag on deployment date, time and location. The flag scale is described in the reference table 2. Example: 1: deployment location seems correct.
DEPLOYMENT_PLATFORM	Identifier of the deployment platform. Example : L'ATALANTE
DEPLOYMENT_CRUISE_ID	Identifier of the cruise used to deploy the platform. Example : POMME2
DEPLOYMENT_REFERENCE_STATION_ID	Identifier of CTD or XBT stations used to verify the first profile. Example : 58776, 58777
DEPLOYMENT_END_DATE	Date (UTC) of the recoveryof the glider. Format : YYYYMMDDHHMISS Example : 20011230090500 : December 30th 2001 09:05:00
DEPLOYMENT_END_LATITUDE	Latitude of the glider recovery. Unit: degree north. Example : 44.4991 : 44° 29' 56.76" N
DEPLOYMENT_END_LONGITUDE	Longitude of the glider recovery. Unit : degree east Example : 16.7222 : 16° 43' 19.92" E
DEPLOYMENT_END_QC	Quality flag on the glider recovery date, time and location. The flag scale is described in the reference table 2. Example: 1: deployment location seems correct.
DEPLOYMENT_END_STATUS	Status of the end of mission of the glider. "R": retrieved "L": lost
DEPLOYMENT_OPERATOR	Name of the person in charge of the glider deployment

2.6.3 Glider sensor information

This section contains the main characteristics of the glider sensors, stored in the attributes of « sensor » meta-data container.

- n_sensor = X (the number of sensors)
- int sensor(n_sensor)

Name	Comment
SENSOR	Names of the sensors mounted on the glider Example: CTD, OPTODE.
SENSOR_MAKER	Name of the manufacturer of the sensor. Example : SEABIRD The draft list of sensor makers is available at : https://docs.google.com/spreadsheet/ccc?key=0AitL8e3zpeffdEtyVmN3a0hvUC1NMDJMcHlLN2FMSlE&pli=1#gid=3
SENSOR_MODEL	Model of sensor. Example : SBE41 The draft list of sensor models is available at : https://docs.google.com/spreadsheet/ccc?key=0AitL8e3zpeffdEtyVmN3a0hvUC1NMDJMcHlLN2FMSlE&pli=1#gid=6
SENSOR_SERIAL_NO	Serial number of the sensor. Example : 2646 036 073

SENSOR_PARAMETERS	Comma separated list of parameters measured by the sensor. The parameter names are listed in reference table 3. Examples : TEMP, PSAL, CNDC TEMP : temperature PSAL : practical salinity CNDC : conductivity

2.4.7 Glider parameter information

This section contains the main characteristics of the glider parameters, stored in the attributes of « parameter » meta-data container.

- n_parameter = X (the number of parameters)
- int parameter(n_parameter)

Name	Comment
PARAMETER	Names of the parameters measured by the glider's sensors or derived from glider's measurements. The parameter names are listed in reference table 3. Examples : TEMP, PSAL, CNDC TEMP : temperature in celsius PSAL : practical salinity in psu CNDC : conductvity in mhos/m
PARAMETER_UNITS	Units of accuracy and resolution of the parameter. Example : psu
PARAMETER_ACCURACY	Accuracy of the parameter. Example: "8 micromole/l or 5%"
PARAMETER_RESOLUTION	Resolution of the parameter. Example : 0.001 micromole/l

In the calibration information arrays, PARAMETER variable is also defined. Delete it and use the PARAMETER definition given in glider parameter information.

Calibration information is provided for all the N_PARAM parameters produced (or derived) by the glider even if there is no calibration information for some of them.

For derived parameters, calibration information arrays are used to explain the derivation process.

2.6.4 Glider parameter derivation and calibration information

This section contains information about the parameter derivation and the parameter calibration.

A derived parameter is calculated from one or several parameters.
Example: salinity is derived from conductivity, temperature and pressure.

Calibrations are applied to parameters to create adjusted parameters. Different calibration methods will be used by groups processing glider data. When a method is applied, its description is stored in the following fields.

If no derivation or calibration is available, N_DERIVATION is set to 1, all values of the derivation section are set to fill values.

Name	Definition	Comment
DERIVATION_PARAMETER	char DERIVATION_PARAMETER(N_DERIVATION, STRING16); DERIVATION_PARAMETER:long_name = "List of parameters with derivation or calibration information"; DERIVATION_PARAMETER:conventions = "EGO reference table 3"; DERIVATION_PARAMETER:_FillValue = " ";	Name of the derived or calibrated parameter. The list of parameters is in reference table 3. Example : "PSAL"
DERIVATION_EQUATION	char DERIVATION_EQUATION(N_DERIVATION, STRING256); DERIVATION_EQUATION:long_name = "Derivation or calibration equation for this parameter"; DERIVATION_EQUATION:_FillValue = " ";	Derivation or calibration equation applied to the parameter. Example : "Tc = a1 * T + a0"
DERIVATION_COEFFICIENT	char DERIVATION_COEFFICIENT(N_DERIVATION,STRING256); DERIVATION_COEFFICIENT:long_name = "Derivation or calibration coefficients for this equation"; DERIVATION_COEFFICIENT:_FillValue = " ";	Derivation or calibration coefficients for this equation. Example : "a1=0.99997 , a0=0.0021"
DERIVATION_COMMENT	char DERIVATION_COMMENT(N_DERIVATION,STRING256); DERIVATION_COMMENT:long_name = "Comment applying to this parameter derivation or calibration"; DERIVATION_COMMENT:_FillValue = " ";	Comment about this derivation or calibration Example : "The sensor is not stable"
DERIVATION_DATE	char DERIVATION_DATE(N_DERIVATION, DATE_TIME) DERIVATION_DATE:long_name = "Date (UTC) of derivation or calibration"; DERIVATION_DATE:conventions = "YYYYMMDDHHMISS"; DERIVATION_DATE:_FillValue = " ";	Date of the derivation or calibration. Format : YYYYMMDDHHMISS Example : "20011230090500" : December 30th 2001 09:05:00

Note on derived parameters, such as oxygen and salinity

Some parameters are calculated (derived) from observed parameters.

Example: salinity is calculated from temperature, conductivity and pressure.

The equation used to calculate derived parameters is recorded as the first calibration of the parameter.

Its calibration date is set to the deployment date.

Note on pre-deployment calibrations

Pre-deployment calibrations are recorded in this section. Their calibration date is set to the deployment date.

2.7 Gliders technical data

The glider technical data are managed as variables (see “variables” chapter 2.5).

Generally, a technical data has no CF standard name.

3 Reference tables

3.1 Reference tables 1: data type

The data_type global attribute should have one of the valid values listed here.

Data type
EGO glider time-series data

3.2 Reference table 2: Variable quality control flag scale

The quality control flags indicate the data quality of the data values in a file, and are normally assigned after quality control procedures have been performed. These codes are used in the <PARAM>_QC, TIME_QC, POSITION_QC variables to describe the quality of each measurement.

Code	Meaning	Comment
0	No QC was performed	-
1	Good data	All QC tests passed.
2	Probably good data	-
3	Bad data that are potentially correctable	These data are not to be used without scientific correction or re-calibration.
4	Bad data	Data have failed one or more tests.
5	Value changed	Data may be recovered after transmission error.
6	-	Not used.
8	Interpolated value	Missing data may be interpolated from neighbouring data in space or time.
9	Missing value	-

3.2.1 Reference table 2.2: cell methods

From NetCDF Climate and Forecast (CF) Metadata Conventions, Version 1.2, 4 May, 2008. In the Units column, *u* indicates the units of the physical quantity before the method is applied.

Cell Method	Units	Description
point	u	The data values are representative of points in space or time (instantaneous)
sum	u	The data values are representative of a sum or accumulation over the cell
maximum	u	Maximum
median	u	Median

mid_range	u	Average of maximum and minimum
minimum	u	Minimum
mean	u	Mean (average value)
mode	u	Mode (most common value)
standard_deviation	u	Standard deviation
variance	u2	Variance

3.3 Reference table 3: EGO parameter dictionary

3.3.1 Convention for parameter names, standard names and units

- Parameter names should start with a code based on SeaDataNet-BODC parameter discovery vocabulary.

  They are not strictly standardized, however.

  When necessary, a parameter name has a suffix that designates secondary parameters. The suffix starts with the character "_".
- The NetCDF "standard_name" attribute contains the standardized parameter name from CF conventions.
- The NetCDF "units" attribute are compliant with Udunits as implemented in the CF/COARDS standards.

As the parameter names are not strictly standardized, one should use the standard_name attribute to query a particular measurement from different data files.
Relevant information on the parameter is recorded in the attributes of the parameter; _xxx in the parameter name is just a guide.
Clarify the description of calculated parameters (ex : PSAL).

Example

On a glider, sea temperature measured by a series of Microcat CTD is reported as TEMP, with a standard name of sea_water_temperature.

Secondary temperature measurement performed by an oxygen sensor is reported as DOXY_TEMP with a standard name of temperature_of_sensor_for_oxygen_in_sea_water.

For both measurements, the unit attribute is "degree_Celsius".

3.3.2 EGO parameter list

The EGO parameter list is regularly updated with new parameters. It is available from:

- http://xxxx/ego-parameter-list

On December 2012, the parameter list was as below (not available yet…).

3.3.3 References

The EGO standard names are taken from the CF standard names, available at:

- http://cf-pcmdi.llnl.gov/documents/cf-standard-names/

The parameter names are shared with Argo floats parameter names.

- http://www.argodatamgt.org/Documentation

The units are compliant with Udunits, as implemented by the CF standard; definitions are available at:

- http://www.unidata.ucar.edu/software/udunits

The valid parameter names, standard names are available on GDACs ftp servers.

3.4 Reference table 4: Data Assembly Center Codes

Data Assembly Centers and institutions	
IF	Ifremer for Coriolis (French joint project for operational oceanography)
BO	British Oceanographic Data Centre

3.5 Reference table 5: Location classes

ARGOS location classes	
Value	Estimated accuracy in latitude and longitude
0	Argos accuracy estimation over 1500m radius
1	Argos accuracy estimation better than 1500m radius
2	Argos accuracy estimation better than 500 m radius
3	Argos accuracy estimation better than 250 m radius
G	GPS positioning accuracy
I	Iridium accuracy

3.6 Reference table 7: History action codes

Code	Meaning
CF	Change a quality flag
CR	Create record
CV	Change value
DC	Station was checked by duplicate checking software
ED	Edit a parameter value
IP	This history group operates on the complete input record
NG	No good trace
PE	Position error. Profile position has been erroneously encoded. Corrected if possible.
QC	Quality Control
QCF$	Tests failed
QCP$	Test performed
SV	Set a value
TE	Time error. Profile date/time has been erroneously encoded. Corrected if possible.
UP	Station passed through the update program

3.7 Reference table 8: Instrument types

The instrument type codes come from WMO table 1770.

Glider instrument codes should be requested to WMO.

~~As a default value, EGO uses the instrument type 830: CTD.~~

Code number	instrument
	Seaglider
	Slocum
	Spray

3.8 Reference table 9: Phases of the glider trajectory

A glider regularly performs surface, descent, inflexion, subsurface drift and ascent phases.

During ascent or descent phase, the glider performs vertical profiles.

Code	Meaning	Comment
0	surface drift	the glider is drifting on the surface layer
1	descent	the glider is descending
2	subsurface drift	the glider is drifting in subsurface
3	inflexion	the glider is changing its trajectory
4	ascent	the glider is ascending
5	grounded	the glider touched the ground or seafloor or onshore
6	inconsistent	the glider pressure is not consistent with the surrounding pressures

3.9 Reference table 10: Positioning method

The positions reported in variables latitude and longitude are reported from various sources such as: GPS, Argos or interpolation.

Code	Meaning	Comment
0	gps	gps positioning method.
1	argos	argos positioning method.
2	interpolated	position derived from other positions
3	iridium	Iridium positioning method

3.10 Reference table 11: QC test binary IDs

This table is used to record the result of the quality control tests in the history section.

The binary IDs of the QC tests are used to define the history variable HISTORY_QCTEST, whose value is computed by adding the binary ID together, then translating to a hexadecimal number. An example is given on §4.

The test numbers and the test names are listed in the Argo Quality Control Manual:

- §2.1 "Argo Real-Time Quality Control Test Procedures on Vertical Profiles", and
- §2.2 "Argo Real-Time Quality Control Test Procedures on Trajectories"

See http://www.argodatamgt.org/Documentation .

Test number	QC test binary ID	Test name
1	2	Platform Identification test
2	4	Impossible Date test
3	8	Impossible Location test
4	16	Position on Land test
5	32	Impossible Speed test
6	64	Global Range test
7	128	Regional Global Parameter test
8	256	Pressure Increasing test
9	512	Spike test
10	*1024*	*Top and Bottom Spike test (obsolete)*
11	2048	Gradient test
12	4096	Digit Rollover test
13	8192	Stuck Value test
14	16384	Density Inversion test
15	32768	Grey List test
16	65536	Gross Salinity or Temperature Sensor Drift test
17	131072	Visual QC test
18	261144	Frozen profile test
19	524288	Deepest pressure test
20	1044576	Questionable Argos position test

3.11 Reference table 12: History steps codes

Code	Meaning
ARFM	Convert raw data from telecommunications system to a processing format

ARGQ	Automatic QC of data reported in real-time has been performed
IGO3	Checking for duplicates has been performed
ARSQ	Delayed mode QC has been performed
ARCA	Calibration has been performed
ARUP	Real-time data have been archived locally and sent to GDACs
ARDU	Delayed data have been archived locally and sent to GDACs
RFMT	Reformat software to convert hexadecimal format reported by the buoy to our standard format
COOA	Coriolis objective analysis performed

If individual centers wish to record other codes, they may add to this list as they feel is appropriate.

3.12 Reference table 19: Data mode

The values for the global attribute “data_mode” is defined as follows:

Value	Meaning
R	Real-time data. Data coming from the (typically remote) platform through a communication channel without physical access to the instruments, disassembly or recovery of the platform. Example: for a glider with a radio communication, this would be data obtained through the radio.
P	Provisional data. Data obtained after the instruments or the platform have been recovered or serviced. Example: for instruments on a glider, this would be data downloaded directly from the instruments after the glider has been recovered on a ship.
D	Delayed-mode data. Data published after all calibrations and quality control procedures have been applied on the internally recorded or best available original data. This is the best possible version of processed data.
M	Mixed. This value is only allowed in the global attribute “data_mode” or in attributes to variables in the form “<PARAM>:DM_indicator”. It indicates that the file contains data in more than one of the above states.

3.13 Reference table 20: Sensor mount characteristics

The <PARAM>:”sensor_mount” attribute indicates the way a sensor is mounted on a glider.

The following table lists the valid sensor_mount attribute values.

Sensor mount
mounted_on_fixed_structure
mounted_on_surface_buoy
mounted_on_glider_line
mounted_on_bottom_lander
mounted_on_moored_glider
mounted_on_glider
mounted_on_shipborne_fixed
mounted_on_shipborne_glider

mounted_on_seafloor_structure
mounted_on_benthic_node
mounted_on_benthic_crawler
mounted_on_surface_buoy_tether
mounted_on_seafloor_structure_riser
mounted_on_fixed_subsurface_vertical_glider

3.14 Reference table 21: Sensor orientation characteristics

When appropriate, the <PARAM>:"sensor_orientation" attribute indicates the way a sensor is oriented on a glider.

The following table lists the valid sensor_orientation attribute values.

Sensor orientation	comment
downward	Example : ADCP measuring from surface to bottom currents.
upward	Example : In-line ADCP measuring currents towards the surface
vertical	-
horizontal	-

3.15 Reference table 22: type of glider

Sensor orientation	Comment
coastal glider	-
open ocean glider	-
-	-
-	-

4 Using the History section of the EGO NetCDF structure

Within the NetCDF format are a number of fields that are used to track the progression of the data through the data system. This section records the processing stages, results of actions that may have altered the original values and information about QC tests performed and failed. The purpose of this document is to describe how to use this section of the format.

The next sections provide examples of what is expected. The information shown in the column labeled "Sample" is what would be written into the associated "Field" name in the NetCDF format.

4.1 Recording information about the Delayed Mode QC process

The process of carrying out delayed mode QC may result in adjustments being made to observed variables. The table below shows how to record that the delayed mode QC has been done. Note that the fields HISTORY_SOFTWARE, HISTORY_SOFTWARE_RELEASE and HISTORY_REFERENCE are used together to document the name and version of software used to carry out the delayed QC, and the reference database used in the process. The contents of these three fields are defined locally by the person carrying out the QC.

Example: History entry to record that delayed mode QC has been carried out

Field	Sample	Explanation
HISTORY_INSTITUTION	CI	Selected from the list in reference table 4
HISTORY_STEP	ARSQ	Selected from the list in reference table 12.
HISTORY_SOFTWARE	WJO	This is a locally defined name for the delayed mode QC process employed.
HISTORY_SOFTWARE_RELEASE	1	This is a locally defined indicator that identifies what version of the QC software is being used.
HISTORY_REFERENCE	WOD2001	This is a locally defined name for the reference database used for the delayed mode QC process.
HISTORY_DATE	20030805000000	The year, month, day, hour, minute, second that the process ran
HISTORY_ACTION	IP	Selected from the list in reference table 7
HISTORY_PARAMETER	FillValue	This field does not apply (1)
HISTORY_START_TIME	FillValue	This field does not apply
HISTORY_STOP_TIME	FillValue	This field does not apply
HISTORY_PREVIOUS_VALUE	FillValue	This field does not apply
HISTORY_QCTEST	FillValue	This field does not apply

Note

(1) The present version of delayed mode QC only tests salinity and as such it is tempting to place “PSAL” in the _PARAMETER field. In future, delayed mode QC tests may include tests for temperature, pressure and perhaps other parameters. For this reason, simply addressing the software and version number will tell users what parameters have been tested.

4.2 Recording processing stages

Each entry to record the processing stages has a similar form. An example is provided to show how this is done. Note that reference table 12 contains the present list of processing stages and there should be at least one entry for each of these through which the data have passed. If data pass through one of these steps more than once, an entry for each passage should be written and the variable N_HISTORY updated appropriately.

Some institutions may wish to record more details of what they do. In this case, adding additional "local" entries to table 12 is permissible as long as the meaning is documented and is readily available. These individual additions can be recommended to the wider community for international adoption.

Example: History entry to record decoding of the data.

Field	Sample	Explanation
HISTORY_INSTITUTION	ME	Selected from the list in reference table 4
HISTORY_STEP	ARFM	Selected from the list in reference table 12.
HISTORY_SOFTWARE	FillValue	This field does not apply
HISTORY_SOFTWARE_RELEASE	FillValue	This field does not apply
HISTORY_REFERENCE	FillValue	This field does not apply
HISTORY_DATE	20030805000000	The year, month, day, hour, minute, second that the process ran
HISTORY_ACTION	IP	Selected from the list in reference table 7
HISTORY_PARAMETER	FillValue	This field does not apply
HISTORY_START_PRES	FillValue	This field does not apply
HISTORY_STOP_PRES	FillValue	This field does not apply
HISTORY_PREVIOUS_VALUE	FillValue	This field does not apply
HISTORY_QCTEST	FillValue	This field does not apply

4.3 Recording QC Tests Performed and Failed

The delayed mode QC process is recorded separately from the other QC tests that are performed because of the unique nature of the process and the requirement to record other information about the reference database used. When other tests are performed, such as the automated real-time QC, a group of tests are applied all at once. In this case, instead of recording that each individual test was performed and whether or not the test was failed, it is possible to document all of this in two history records.

The first documents what suite of tests was performed, and the second documents which tests in the suite were failed. A test is failed if the value is considered to be something other than good (i.e. the resulting QC flag is set to anything other than "1"). An example of each is provided. If data pass through QC more than once, an entry for each passage should be written and the variable N_HISTORY updated appropriately.

Example: QC tests performed and failed.

The example shown here records that the data have passed through real-time QC and that two tests failed. The encoding of tests performed is done by adding the ID numbers provided in reference table 11 for all tests performed, then translating this to a hexadecimal number and recording this result.

Record 1: Documenting the tests performed

Field	Sample	Explanation
HISTORY_INSTITUTION	ME	Selected from the list in reference table 4
HISTORY_STEP	ARGQ	Selected from the list in reference table 12.
HISTORY_SOFTWARE	FillValue	This field does not apply
HISTORY_SOFTWARE_RELEASE	FillValue	This field does not apply
HISTORY_REFERENCE	FillValue	This field does not apply
HISTORY_DATE	20030805000000	The year, month, day, hour, minute, second that the process ran
HISTORY_ACTION	QCP$	Selected from the list in reference table 7
HISTORY_PARAMETER	FillValue	This field does not apply
HISTORY_START_TIME	FillValue	This field does not apply
HISTORY_STOP_TIME	FillValue	This field does not apply
HISTORY_PREVIOUS_VALUE	FillValue	This field does not apply
HISTORY_QCTEST	1BE	This is the result of all tests with IDs from 2 to 256 having been applied (see reference table 11)

Record 2: Documenting the tests that failed

Field	Sample	Explanation
HISTORY_INSTITUTION	ME	Selected from the list in reference table 4
HISTORY_STEP	ARGQ	Selected from the list in reference table 12.
HISTORY_SOFTWARE	FillValue	This field does not apply
HISTORY_SOFTWARE_RELEASE	FillValue	This field does not apply
HISTORY_REFERENCE	FillValue	This field does not apply
HISTORY_DATE	20030805000000	The year, month, day, hour, minute, second that the process ran
HISTORY_ACTION	QCF$	Selected from the list in reference table 7
HISTORY_PARAMETER	FillValue	This field does not apply
HISTORY_START_TIME	FillValue	This field does not apply
HISTORY_STOP_TIME	FillValue	This field does not apply
HISTORY_PREVIOUS_VALUE	FillValue	This field does not apply
HISTORY_QCTEST	A0	This is the result when data fail tests with IDs of 32 and 128 (see reference table 11)

4.4 Recording changes in values

The PIs have the final word on the content of the data files in the Argo data system. In comparing their data to others there may arise occasions when changes may be required in the data.

We will use the example of recomputation of where the glider first surfaced as an example. This computation process can be carried out once all of the messages from a glider have been received. Not all real-time processing centers make this computation, but it can be made later on and added to the delayed mode data. If this is the case, we would insert the new position into the latitude and longitude fields and we would record the previous values in two history entries. Recording these allows us to return to the original value if we have made an error in the newly computed position. The two history entries would look as follows.

Example: Changed latitude

Field	Sample	Explanation
HISTORY_INSTITUTION	CI	Selected from the list in reference table 4
HISTORY_STEP	ARGQ	Selected from the list in reference table 12.
HISTORY_SOFTWARE	FillValue	This field does not apply
HISTORY_SOFTWARE_RELEASE	FillValue	This field does not apply
HISTORY_REFERENCE	FillValue	This field does not apply
HISTORY_DATE	2003080500000 0	The year, month, day, hour, minute, second that the process ran
HISTORY_ACTION	CV	Selected from the list in reference table 7
HISTORY_PARAMETER	LAT$	A new entry for reference table 3 created by institution CI to indicate changes have been made in the latitude.
HISTORY_START_TIME	FillValue	This field does not apply
HISTORY_STOP_TIME	FillValue	This field does not apply
HISTORY_PREVIOUS_VALUE	23.456	This is the value of the latitude before the change was made.
HISTORY_QCTEST	FillValue	This field does not apply

Notes

1. Be sure that the new value is recorded in the latitude and longitude of the trajectory.
2. Be sure that the POSITION_QC flag is set to “5” to indicate to a user that the value now in the position has been changed from the original one that was there.
3. Be sure to record the previous value in history entries.

It is also sometimes desirable to record changes in quality flags that may arise from reprocessing data through some QC procedures. In this example, assume that whereas prior to the analysis, all temperature values from 75 to 105 dbars were considered correct, after the analysis, they are considered wrong. The history entry to record this would look as follows.

Example: Changed flags

Field	Sample	Explanation
HISTORY_INSTITUTION	CI	Selected from the list in reference table 4
HISTORY_STEP	ARGQ	Selected from the list in reference table 12.
HISTORY_SOFTWARE	FillValue	This field does not apply
HISTORY_SOFTWARE_RELEASE	FillValue	This field does not apply
HISTORY_REFERENCE	FillValue	This field does not apply
HISTORY_DATE	20030805000000	The year, month, day, hour, minute, second that the process ran
HISTORY_ACTION	CF	Selected from the list in reference table 7
HISTORY_PARAMETER	TEMP	Selected from the list in reference table 3
HISTORY_START_PRES	75	Shallowest pressure of action.
HISTORY_STOP_PRES	105	Deepest pressure of action.
HISTORY_PREVIOUS_VALUE	1	This is the value of the quality flag on temperature readings before the change was made.
HISTORY_QCTEST	FillValue	This field does not apply

Notes

1. The new QC flag of “4” (to indicate wrong values) would appear in the <param>_QC field.

5 GDAC organization

There are two GDACs (global data assembly centers) for redundancy, which are the users' access points for EGO data. One GDAC is located in France (Coriolis, http://www.coriolis.eu.org), the other one in the USA (NDBC, National Data Buoy Center, http://www.ndbc.noaa.gov). The GDACs handle EGO data, metadata, and index files on ftp servers. The servers at both GDACs are synchronized at least daily to provide the same EGO data.

The user can access the data at either GDAC's ftp site:

- ftp://ftp.ifremer.fr/ifremer/ego-gliders
- ftp://xxx.noaa.gov/

From these root directories of the GDACs downward, the organization of the directories and files is:

- DATA/site/FileName.nc
  site: EGO site code

The site codes will be listed in the "EGO catalogue" document at either GDAC's root directory.

5.1 File naming convention

The EGO file names use the following naming convention for data and metadata files.

YYY/GL_XXX_YYY _ZZZ_T.nc

- GL: EGO gliders prefix
- XXX: platform code from the EGO catalogue
- YYY: deployment day YYYYMMDD
- ZZZ: deployment code
- T: data Mode
  - R: real-time data
  - P : provisional data
  - D: delayed mode
  - M: mixed delayed mode and real-time.
- .nc : NetCDF file suffix

Example

- pytheas/gl_pytheas_20100612_MooseT0009_R.nc

This file contains observations and metadata from the Pytheas glider, from the MooseT0009 deployment performed in June 2010.

5.2 Index of glider deployments files

To allow for data discovery without downloading the data files themselves, an index file is created at the GDAC level, which lists all available data files and the location and time ranges of their data contents:

- The data index file is located at the root directory of the GDAC.
- The index file contains the list and a description of all data files available on the GDAC.
- There is a header section, lines of which start with # characters.
- The information sections are comma-separated values.
- Each line contains the following information:
  - file: the file name, beginning from the GDAC root directory
  - date_update: the update date of the file, YYYY-MM-DDTHH:MI:SSZ
  - start_date: first date for observations, YYYY-MM-DDTHH:MI:SSZ
  - end_date: last date for observations, YYYY-MM-DDTHH:MI:SSZ
  - southern_most_latitude, decimal degrees
  - northern_most_latitude, decimal degrees
  - western_most_longitude, decimal degrees
  - eastern_most_longitude, decimal degrees
  - geospatial_vertical_min, decibar
  - geospatial_vertical_max, decibar
  - update_interval: M monthly, D daily, Y yearly, V void
  - size: the size of the file in megabytes
  - gdac_creation_date: date of creation of the file on the GDAC, YYYY-MM-DDTHH:MI:SSZ
  - gdac_update_date: date of update of the file on the GDAC, YYYY-MM-DDTHH:MI:SSZ
  - data_mode: R, P, D, M (real-time, provisional, delayed mode, mixed; see reference table 19)
  - parameters: list of parameters (standard_name) available in the file separated with blank

The fill value is empty: "".

GDAC data files index: EGO_files_index.txt
EGO FTP GLOBAL INDEX # FTP://FTP.IFREMER.FR/IFREMER/EGO # Contact: HTTP://WWW.EGO.ORG # Index update date YYYY-MM-DDTHH:MI:SSZ: 2008-03-30T18:37:46Z # #file,date_update,start_date,end_date, southern_most_latitude,northern_most_latitude,western_most_longitude,eastern_most_longitude, geospatial_vertical_min,geospatial_vertical_max,update_interval,size,gdac_creation_date,gdac_update_date, data_mode,parameters PYTHEAS/GL_PYTHEAS_201006_R_LATEX.nc,2008-04-12T08:05:00Z,2007-03-17T18:07:00Z,2008-04- 12T08:05:00Z,0,0,-170,-170,16.7,0,550,M,2008-04-12T08:05:00Z,2008-04- 12T08:05:00Z,R,sea_water_pressure sea_water_temperature sea_water_salinity

6 Data distribution

6.1 DAC to GDAC data distribution

The Data Assembly Centers (DAC) collect data from glider operators (real-time) or from scientists (delayed mode data).

In real-time, each DAC converts glider data into EGO-NetCDF files. It applies the real-time quality controls on the NetCDF files.

The DAC pushes these quality control data files to the Global Data Assembly Centers (GDACs).

The role of the GDAC is to distribute the best versions of EGO NetCDF files.

6.2 DAC to GTS data distribution

The EGO glider data received in real-time are quality-controlled. The real-time quality control procedures are described in the EGO glider quality control manual. They are automatically applied, without human intervention to minimize the delay between data observation and data distribution.

For each active glider, the data that passed the real-time QC tests are distributed on GTS (the WMO data transmission system). Data distributed on GTS should be less than 30 days old. The target for distribution is within 48 hours of the observation time.

TESAC format distribution

The vertical profiles extracted from the glider time-series are distributed as TESAC messages.

Each vertical profile should have a vertical length greater or equal to 10 decibars.

Buoy format distribution

The glider time-series are distributed as BUOY format messages.

BUFR format distribution

In a near future, the glider time-series will be distributed in BUFR format. The glider BUFR template is under construction.

7 Glossary, definitions

This chapter gives a definition for the EGO items described in this manual.

7.1 Observatory

An observatory is a facility that manages a series of gliders.

7.2 Deployment

The deployment is the period between the launch and recovery or loss of a glider.

7.3 Glider

A steered and autonomous platform deployed in the sea that performs environmental monitoring.

7.4 Sensor

A device that measures environmental parameter but does not digitize data for transmission, it needs to be connected to an instrument to produce a data stream that a computer can read. Examples: Transmissiometer, Fluorometer, Oxygen sensor.

7.4.1 Parameter measured by the sensor

What was measured.

7.4.2 Calibration of the parameter measured by the sensor

Verification of any operation measurement against independent measurements to derive a corrected value or a new parameter.

7.5 Principal Investigator (PI)

The **Principal Investigator (PI)**, typically a scientist at a research institution, maintains the observing platform and the sensors that deliver the data. He or she is responsible for providing the data and all auxiliary information to a **Data Assembly Center (DAC)**.

7.6 Global Data Assembly Centre (GDAC)

The **GDAC** distributes the best copy of the data files. When a higher quality data file (e.g.

calibrated data) is available, it replaces the previous version of the data file.
The user can access the data at either GDAC, cf. section “GDAC organization”.

7.7 Data Assembly Centre (DAC)

The **DAC** assembles EGO-compliant files from this information and delivers these to the two **Global Data Assembly Centers (GDACs)**, where they are made publicly available.