EuroGO-SHIP
Enhancing ocean observations

SADCP data format converter and viewer

From oceanSITES to SeaDataNet TrajectoryProfile

May 31, 2024 /V2.0

EuroGO-SHIP has received funding from the European Union HORIZON EUROPE Programme under Grant Agreement No. 101094690

About this document

Title	M2.2: SADCP data format converter and viewer
Work Package	Concept Developlent through Co-Design
Lead Partner	Ifremer
Lead Author (Org)	Michèle Fichaut (Ifremer), Julie Gatti (Ifremer), Pascale Lherminier (Ifremer)
Contributing Author(s)	Sébastien Crouzille (CapGemini engineering), Tina Odaka (Ifremer), Léa Franc (Ifremer)
Reviewers	WMO, NORCE
Due Date	31 May 2024
Submission Date	27 May 2024
Version	**2.0**

Dissemination Level

X	PU: Public
	PP: Restricted to other programme participants (including the Commission)
	RE: Restricted to a group specified by the consortium (including the Commission)
	CO: Confidential, only for members of the consortium (including the Commission)

EuroGO-SHIP: Developing a Research Infrastructure Concept to Support European Hydrography is a Research and Innovation action (RIA) funded by the Horizon Europe Work programme topics addressed: HORIZON-INFRA-2022-DEV-01-01– Research infrastructure concept development. Start date: 01 December 2022. End date: 30 November 2025.

Disclaimer: This material reflects only the author's view, and the European Commission is not responsible for any use that may be made of the information it contains.

The EuroGO-SHIP project has received funding from the European Union's Horizon Europe Research and Innovation Programme under Grant Agreement No. 101094690.

TABLE of CONTENT

1. Introduction

This document describes the specifications of the converter that has to be developed to convert ADCP data at OceanSITES netCDF format (as the output of the CASCADE software used at IFREMER for the processing of ADCP data) to SeaDataNet netCDF trajectoryProfile format

The OceanSITES netCDF format is described in [1], the specificities of the files generated with the CASCADE software are explained in [2], and the SeaDataNet trajectory Profile data is specified in [3].

Ship-mounted acoustic Doppler current profilers (ADCP) have been used for over 25 years, and have been available on most research ships for much of that time. They are easy to operate on a routine basis, permitting nearly continuous monitoring of the upper ocean current structure beneath each ship.

The ADCP OceanSITES files generated by the CASCADE software (2] include current velocities (absolute velocities and velocities relative to the vessel) as well as the ship's speeds. Files are also enriched with auxiliary data (tide, bathymetry, etc.) and quality indicators on the data.

The particularity of ADCP data is that the measurements are done along the trajectory of the ship on a water layer (from depth *x* to depth *y*). In netCDF it means that the geophysical variables have another dimension (Depth) in addition to the Time dimension: that is the reason for the name of trajectoryProfile of the SeaDataNet netCDF format ([3], §4.5.5).

To increase the end user uptake, we have developed a web application SADCP Viewer [6]. SADCP Viewer is a reproducible web application based on pangeo environment [7, 8] and help users to better discover SADCP data on an web interface without manually downloading SADCP data stored in SeaDataNet.

2. Integration into the Octopus software

The new format converter will be integrated into the SeaDataNet Octopus software [4] which is already specialised in conversions to SeaDataNet format from different input formats like EGO for glider data, MGD77 for Magnetism, Gravimetry, and Depth data sets.

The plans are to convert all French ADCP data issued from CASCADE software to SeaDataNet format. We are not sure that the software could apply to other data sets, because CASCADE is an internal IFREMER tool.

Then, in the second step, we will have to write a converter from CODAS format (https://currents.soest.hawaii.edu/docs/adcp_doc/index.html#) to SeaDataNet netCDF trajectoryProfile format.

3. Specifications of the ADCP OceanSITES – netCDF SeaDataNet converter

3.1. File type detection

NetCDF OceanSITES format files, issued from the Cascade software, can be identified through the following Global Attributes:

- :SOFTWARE = "Cascade Exploitation [version of Cascade]".
- :CONVENTIONS =”OceanSite dictionary”

If both global attributes are present:

Octopus should display the following message when opening the file. Message: Detected input format: NetCDF OceanSITES ADCP generated by :SOFTWARE (global attribute of the OceanSITES file).

Once an OceanSITES Cascade file is detected, only netCDF is available for conversion, ODV and MedAtlas conversions are not available if Octopus is Used.

The OCTOPUS checker must not be available either.

3.2. Dimensions of the SeaDataNet trajectoryProfile file

INSTANCE = 1 (=CONST1 in in oceanSITES format)
MAXZ = Nbmax of Depth levels (= N_LEVEL in in oceanSITES format)
MAXT = Nb max of TIME (= N_DATE_TIME in oceanSITES format)
REFMAX = Nb max of REF
STRINGxx = Length for short strings (maximum length of STRINGS of METADATA)
STRINGyyy = Length for long strings (length of SDN references XLINK)
STRINGzz = length for DATE string (maximum length of STRINGS of DATA)

3.3. Mapping of the variables

The table below describes the mapping between oceanSITES variables and SeaDataNet variables.

Additional SeaDataNet variables are also described without mapping.

GA stands for Global Attribute. The **bold characters** (and the Rule column set to M for Mandatory and O for Optional) indicate **mandatory parameters**.
Some of the oceanSITES variables are not kept on purpose and should be ignored.
All geophysical variables have a *:coordinates* attribute which follows the following rules:

If the geophysical have as dimensions :

(INSTANCE,MAXT,MAXZ) then :coordinates = “TIME PROFZ LATITUDE LONGITUDE”;

(INSTANCE,MAXT) then :coordinates = “TIME LATITUDE LONGITUDE”;

Mapping Variable/Attribute Name			Content
SDN	**OceanSITES**	**Rule: M:mandatory O:optional**	**Description**
int SDN_EDMO_CODE (INSTANCE)	-	**M**	EDMO_Code as input in OCTOPUS settings SDN_EDMO_CODE:long_name = "European Directory of Marine Organisations code for the CDI partner"
char SDN_CRUISE(INSTANCE,STRINGxx)		**M**	
SDN_CRUISE :long_name	ADCP file name		The LOCAL_CSR_ID of the cruise is the first part of the file name (before the first underscore) Search on [EDMO_code]_[LOCAL_CSR_ID] to find the CSR id in the *csrCodeList.xml* file in *\externalResources\CSR* of Octopus *: <gml:name codeSpace="EDMO">486_18001235</gml:name>* The Cruise name is stored in *<gml:cruisename> TR_LHAMIN </gml:cruisename>*
SDN_CRUISE :ship_code	-		From the *csrCodeList.xml* file in *externalResources* of Octopus *<gml:platformcode>35HT</gml:platformcode>*
SDN_CRUISE:shipname	-		From the *csrCodeList.xml* file in *externalResources* of Octopus *<gml:platformname>Thalassa</gml:platform>*
char SDN_STATION(INSTANCE,STRINGxx)	-	**M**	SDN_STATION:long_name = "List of station numbers" ; *00001*
char SDN_LOCAL_CDI_ID(INSTANCE, STRINGxx)	ADCP file name	**M**	SDN_LOCAL_CDI_ID:cf_role = "trajectoryProfile_id" ; SDN_LOCAL_CDI_ID:long_name = "SeaDataNet CDI identifier" Input the literal string 'SADCP_'[the 2 first parts of the ADCP file name]. Example : *SADCP_18001235_1 for the file 18001235_1_TR_LHAMIN_2019_PP_38K_WT_1E_fhv1.nc*
char SDN_XLINK (INSTANCE,REFMAX,STRINGyyy)	-	O	SDN_XLINK:long_name = "External resource linkages" ; Add the sdn_reference for the CSR, the C17 ship code, the LOCAL_CDI_ID and the projects (EDMERP) • **For the CSR** *<gml:metaDataProperty about="https://csr.seadatanet.org/report/20185720/xml"></gml:metaDataProperty> gives <sdn_reference xlink:href="https://csr.seadatanet.org/report/20195582/xml" xlink:role="isObservedBy" xlink:type="SDN:L23::CSR"/>* • **For the ship** *<gml:platformcode>35HT</gml:platformcode> gives* <sdn_reference *xlink:href="https://www.seadatanet.org/urnurl/SDN:C17::35PK" xlink:role="isObservedBy" xlink:type="SDN:L23::NVS2CON"/>* • **For the CDI** *<sdn_reference xlink:href="https://cdi.seadatanet.org/report/edmo/486/SDN_LOCAL_CDI_ID/xml" xlink:role="isDescribedBy" xlink:type="SDN:L23::CDI"/>*

			• **For EDMERP, information to be taken csrCodeList.xml file** *<sdn_reference xlink:href="https://edmerp.seadatanet.org/report/12297" xlink:role="isObservedBy" xlink:type="SDN:L23::EDMERP"/>*
int crs		**M**	crs:grid_mapping_name = "latitude_longitude" ; crs:epsg_code = "EPSG:4326" ; crs:semi_major_axis = 6378137. ; crs:inverse_flattening = 298.257223563 ;
double TIME(INSTANCE,MAXT)	**JULD (N_DATE_TIME) "Julian day relative to REFERENCE_DATE_TIME"**	**M**	Conversion to be done from the unit to chronological Julian date with REFERENCE_DATE_TIME and the value for TIME. **Attributes:** TIME:long_name = "Chronological Julian Date" ; TIME:sdn_parameter_urn = "SDN:P01::CJDY1101" ; TIME:sdn_parameter_name = "Julian Date (chronological)" ; TIME:sdn_uom_urn = "SDN:P06::UTAA" ; TIME:sdn_uom_name = "Days" ; TIME:units = "days since -4713-01-01T00:00:00Z" ; TIME:standard_name = "time" ; TIME:axis = "T" ; TIME:ancillary_variables = "TIME_SEADATANET_QC" ; TIME:calendar = "julian" ; TIME:_FillValue = -99999. ;
byte TIME_SEADATANET_QC (INSTANCE,MAXT)	**-**	**M**	**Values:** All values set to 1 **Attributes:** TIME_SEADATANET_QC:long_name = "SeaDataNet quality flag" ; TIME_SEADATANET_QC:Conventions = "SeaDataNet measurand qualifier flags" ; TIME_SEADATANET_QC:_FillValue = 57b ; TIME_SEADATANET_QC:sdn_conventions_urn = "SDN:L20::" ; TIME_SEADATANET_QC:flag_values = 48b, 49b, 50b, 51b, 52b, 53b, 54b, 55b, 56b, 57b, 65b, 81b ; TIME_SEADATANET_QC:flag_meanings = "no_quality_control good_value probably_good_value probably_bad_value bad_value changed_value value_below_detection value_in_excess interpolated_value missing_value value_phenomenon_uncertain value_below_limit_of_quantification" ;
float ENS_DUR(INSTANCE,MAXT)	JULD_J2 (N_DATE_TIME) JULD_J1(N_DATE_TIME)	O	equal to JULD_J2 – JULD_J1 converted in seconds **Attributes:** ENS_DUR:long_name = "Ensemble duration" ; ENS_DUR:sdn_parameter_urn = "SDN:P01::AZDRZZ01" ; ENS_DUR:sdn_parameter_name = "Sample duration" ; ENS_DUR:sdn_uom_urn = "SDN:P06::UTBB" ; ENS_DUR:sdn_uom_name = "Seconds" ; ENS_DUR:units = "seconds" ; ENS_DUR:standard_name = "" ; ENS_DUR:ancillary_variables = "ENS_DUR _SEADATANET_QC" ; ENS_DUR:_FillValue = -99999. ;

byte ENS_DUR_SEADATANNET_QC (INSTANCE,MAXT)	-	O	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC
char DATE_TIME_UTC(INSTANCE,MAXT,STRINGzz)	DATE_TIME_UTC (N_DATE_TIME,DATE_TIME)	O	TIME in ISO-format YYYYMMDDThhmmss. **Attributes:** DATE_TIME_UTC:long_name = "Date and time at ISO format" ; DATE_TIME_UTC:sdn_parameter_urn = "SDN:P01:: DTUT8601 " ; DATE_TIME_UTC:sdn_parameter_name = " Date and time (UT in ISO8601 format to known precision) " ; DATE_TIME_UTC:sdn_uom_urn = "SDN:P06::TISO" ; DATE_TIME_UTC:sdn_uom_name = " ISO8601 " ; DATE_TIME_UTC:units = "YYYYMMDDThhmmss" ; DATE_TIME_UTC:standard_name = "time" ; DATE_TIME_UTC:ancillary_variables = " DATE_TIME_UTC _SEADATANET_QC" ; DATE_TIME_UTC:_FillValue = " ";
byte DATE_TIME_UTC_SEADATANET_QC (INSTANCE,MAXT)	-	O	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC
	CAS_DATE_FLAG (N_DATE_TIME)		This field is not kept in the SeaDataNet output file but is used to delete data from the dataset: when CAS_DATE_FLAG = 2, all the corresponding data have to be deleted. In all parameters
double LONGITUDE(INSTANCE,MAXT)	**LONGITUDE (N_DATE_TIME)**	**M**	**Attributes:** LONGITUDE:long_name = "Longitude"; LONGITUDE:sdn_parameter_urn = "SDN:P01::ALONZZ01" ; LONGITUDE:sdn_parameter_name = "Longitude east" ; LONGITUDE:sdn_uom_urn = "SDN:P06::DEGE" ; LONGITUDE:sdn_uom_name = "Degrees east" ; LONGITUDE:units = "degrees_east" ; LONGITUDE:standard_name = "longitude" ; LONGITUDE:axis = "X" ; LONGITUDE:ancillary_variables = "POSITION_SEADATANET_QC" ; LONGITUDE:grid_mapping = "crs" ; LONGITUDE:_FillValue = "+999.999999" ;
double LATITUDE (INSTANCE,MAXT)	**LATITUDE (N_DATE_TIME)**	**M**	**Attributes:** LATITUDE:long_name = "Latitude" ; LATITUDE:sdn_parameter_urn = "SDN:P01::ALATZZ01" ; LATITUDE:sdn_parameter_name = "Latitude north" ; LATITUDE:sdn_uom_urn = "SDN:P06::DEGN" ; LATITUDE:sdn_uom_name = "Degrees north" ; LATITUDE:units = "degrees_north" ; LATITUDE:standard_name = "latitude" ; LATITUDE:axis = "Y" ; LATITUDE:ancillary_variables = "POSITION_SEADATANET_QC" ;

			LATITUDE:grid_mapping = "crs"; LATITUDE:_FillValue = "+99.999999" ;
byte POSITION_SEADATANET_QC (INSTANCE,MAXT)	**-**	**M**	**Values:** All values set to 1 **Attributes:** POSITION_SEADATANET_QC:long_name = "SeaDataNet quality flag" ; POSITION_SEADATANET_QC:Conventions = "SeaDataNet measurand qualifier flags" ; POSITION_SEADATANET_QC:_FillValue = 57b ; POSITION_SEADATANET_QC:sdn_conventions_urn = "SDN:L20::" ; POSITION_SEADATANET_QC:flag_values = 48b, 49b, 50b, 51b, 52b, 53b, 54b, 55b, 56b, 57b, 65b, 81b ; POSITION_SEADATANET_QC:flag_meanings = "no_quality_control good_value probably_good_value probably_bad_value bad_value changed_value value_below_detection value_in_excess interpolated_value missing_value value_phenomenon_uncertain value_below_limit_of_quantification" ;
float USHIP (INSTANCE,MAXT)	**UVEL_SHIP (N_DATE_TIME)**	**M**	USHIP:long_name = "Eastward ship velocity"; USHIP:sdn_parameter_urn = "SDN:P01::APEWZZ01" ; USHIP:sdn_parameter_name = "Eastward velocity of measurement platform relative to ground surface" ; USHIP:sdn_uom_urn = "SDN:P06::UVAA" ; USHIP:sdn_uom_name = "Metres per second" ; USHIP:units =’meters/second” ; USHIP:standard_name = “platform_speed_wrt_ground”; USHIP:ancillary_variables = "USHIP_SEADATANET_QC" ; USHIP:_FillValue = -99999.;
byte USHIP_SEADATANET_QC (INSTANCE,MAXT)	**-**	**M**	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC
float VSHIP (INSTANCE,MAXT)	**VVEL_SHIP (N_DATE_TIME)**	**M**	**Attributes:** VSHIP:long_name = "Northward ship velocity"; VSHIP:sdn_parameter_urn = "SDN:P01::APNSZZ01" ; VSHIP:sdn_parameter_name = "Northward velocity of measurement platform relative to ground surface" ; VSHIP:sdn_uom_urn = "SDN:P06::UVAA" ; VSHIP:sdn_uom_name = "Metres per second" ; VSHIP:units = “meters/second” ; VSHIP:standard_name = “platform_speed_wrt_ground”; VSHIP:ancillary_variables = "VSHIP_SEADATANET_QC" ; VSHIP:_FillValue = -99999.;
byte VSHIP_SEADATANET_QC (INSTANCE,MAXT)	**-**	**M**	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC

Float PROFZ (INSTANCE,MAXT,MAXZ)	**DEPH (N_LEVEL)**	**M**	**Values**: Duplicate the values in all MAXT indices. All values have to be multiplied by -1: because the measurements in the OceanSITES file are negative so the opposite with respect to the attribute :positive definition = Down. **Attributes:** PROFZ:long_name = "Bin depth below sea surface" ; PROFZ:sdn_parameter_urn = "SDN:P01::DBINAA01" ; PROFZ:sdn_parameter_name = "Depth (spatial coordinate) of ADCP bin relative to water surface {bin depth} in the water body " ; PROFZ:sdn_uom_urn = "SDN:P06::ULAA" ; PROFZ:sdn_uom_name = "Metres" ; PROFZ:units = "meters"; PROFZ:standard_name = "depth_below_geoid"; PROFZ:axis ="Z" ; PROFZ:positive = "down" ; PROFZ:ancillary_variables = "PROFZ_SEADATANET_QC" ; PROFZ:_FillValue = -99999.;
byte PROFZ_SEADATANET_QC (INSTANCE, MAXZ)		**M**	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC
Float TR_TEMP(INSTANCE, MAXT)	**TEMP_ADCP(N_DATE _TIME)**	**M**	TR_TEMP:long_name = ADCP transducer temperature" ; TR_TEMP:sdn_parameter_urn = "SDN:P01::TEMPTADC" ; TR_TEMP:sdn_parameter_name = "Temperature of the water body by in-situ thermistor on acoustic doppler current profiler (ADCP) transducer" ; TR_TEMP:sdn_uom_urn = "SDN:P06::UPAA" ; TR_TEMP:sdn_uom_name = "Degrees Celsius" ; TR_TEMP:units = "degrees celsius" ; TR_TEMP:standard_name = ""; TR_TEMP:ancillary_variables = "TR_TEMP_SEADATANET_QC" ; TR_TEMP:_FillValue = -99999.;
byte TR_TEMP_SEADATANET_QC (INSTANCE, MAXT)		**M**	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC
Float HEADING (INSTANCE,MAXT)	**HDG(N_DATE_TIME)**	**M**	HEADING:long_name = " Ship Heading" ; HEADING:sdn_parameter_urn = "SDN:P01::HEADCM01" ; HEADING:sdn_parameter_name = "Orientation (horizontal relative to true north) of measurement device {heading}" ; HEADING:sdn_uom_urn = "SDN:P06::UAAA" ; HEADING:sdn_uom_name = "Degrees" ; HEADING:units = "degrees" ; HEADING:standard_name = "platform_orientation"; HEADING:ancillary_variables = "HEADING_SEADATANET_QC" ;

			HEADING:_FillValue = -99999.;
byte HEADING_SEADATANET_QC (INSTANCE, MAXZ)		**M**	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC
Float PITCH (INSTANCE,MAXT)	PTCH(N_DATE_TIME)	O	PITCH:long_name = " Ship Pitch" ; PITCH:sdn_parameter_urn = "SDN:P01::PTCHEI01" ; PITCH:sdn_parameter_name = "Orientation (pitch) of measurement platform by inclinometer" ; PITCH:sdn_uom_urn = "SDN:P06::UAAA" ; PITCH:sdn_uom_name = "Degrees" ; PITCH:units = "degrees" ; PITCH:standard_name = "platform_pitch"; PITCH:ancillary_variables = "PITCH_SEADATANET_QC" ; PITCH:_FillValue = -99999.;
byte PITCH_SEADATANET_QC (INSTANCE, MAXT)		O	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC
Float ROLL (INSTANCE,MAXT)	ROLL(N_DATE_TIME)	O	ROLL:long_name = " Ship Roll" ; ROLL:sdn_parameter_urn = "SDN:P01::ROLLEI01" ; ROLL:sdn_parameter_name = "Orientation (roll angle) of measurement platform by inclinometer" ; ROLL:sdn_uom_urn = "SDN:P06::UAAA" ; ROLL:sdn_uom_name = "Degrees" ; ROLL:units = "degrees" ; ROLL:standard_name = "platform_roll"; ROLL:ancillary_variables = "ROLL_SEADATANET_QC" ; ROLL:_FillValue = -99999.;
byte ROLL_SEADATANET_QC (INSTANCE, MAXT)		O	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC
Float NB_PINGS_PER_ENS (INSTANCE,MAXT)	NB_ENS_AVE(N_DATE_TIME)	O	NB_PINGS_PER_ENS:long_name = Number of pings averaged per ensembles" ; NB_PINGS_PER_ENS:sdn_parameter_urn = "SDN:P01::PNGCNT01" ; NB_PINGS_PER_ENS:sdn_parameter_name = "Count of acoustic signal emissions (per ensemble) {Pings per ensemble} to the water body by acoustic doppler current profiler (ADCP)" ; NB_PINGS_PER_ENS:sdn_uom_urn = "SDN:P06::UUUU" ; NB_PINGS_PER_ENS:sdn_uom_name = "Dimensionless" ; NB_PINGS_PER_ENS:units = "dimensionless" NB_PINGS_PER_ENS:standard_name = ""; NB_PINGS_PER_ENS:ancillary_variables = "NB_PINGS_PER_ENS_SEADATANET_QC" ; NB_PINGS_PER_ENS:_FillValue = -99999.;

byte NB_PINGS_PER_ENS_SEADATANET_QC (INSTANCE, MAXT)		O	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC
float U_BOTTOM (INSTANCE,MAXT)	U_BOTTOM(N_DATE_TIME)	O	**All values have to be multiplied by -1**: because the measurements in the OceanSITES file are relative to the velocity of the ground surface relative to the measurement platform by ADCP bottom tracking so the opposite sign of the P01 definition. U_BOTTOM:long_name = "Bottom Track Eastward velocity" ; U_BOTTOM:sdn_parameter_urn = "SDN:P01::APEWBT01" ; U_BOTTOM:sdn_parameter_name = "Eastward velocity of measurement platform relative to ground surface by ADCP bottom tracking" ; U_BOTTOM:sdn_uom_urn = "SDN:P06::UVAA" ; U_BOTTOM:sdn_uom_name = "Metres per second" ; U_BOTTOM:units = “meters/second” U_BOTTOM:standard_name = “platform_speed_wrt_ground”; U_BOTTOM:ancillary_variables = "U_BOTTOM_SEADATANET_QC" ; U_BOTTOM:_FillValue = -99999.;
byte U_BOTTOM_SEADATANET_QC (INSTANCE,MAXT)	-	O	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC
float V_BOTTOM (INSTANCE,MAXT)	V_BOTTOM(N_DATE_TIME)	O	**All values have to be multiplied by -1**: because the measurements in the OceanSITES file are relative to the velocity of the ground surface relative to the measurement platform by ADCP bottom tracking so the opposite sign of the P01 definition. V_BOTTOM:long_name = " Bottom Track Northward velocity " ; V_BOTTOM:sdn_parameter_urn = "SDN:P01::APNSBT01" ; V_BOTTOM:sdn_parameter_name = "Northward velocity of measurement platform relative to ground surface by ADCP bottom tracking” ; V_BOTTOM:sdn_uom_urn = "SDN:P06::UVAA" ; V_BOTTOM:sdn_uom_name = "Metres per second" ; V_BOTTOM:units = “meters/second” V_BOTTOM:standard_name = “ platform_speed_wrt_ground”; V_BOTTOM:ancillary_variables = "V_BOTTOM_SEADATANET_QC" ; V_BOTTOM:_FillValue = -99999.;
byte V_BOTTOM_SEADATANET_QC (INSTANCE,MAXT)	-	O	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC
float W_BOTTOM (INSTANCE,MAXT)	W_BOTTOM(N_DATE_TIME)	O	W_BOTTOM:long_name = "Bottom Track vertical velocity" ; W_BOTTOM:sdn_parameter_urn = "SDN:P01::APZABT01" ; W_BOTTOM:sdn_parameter_name = "Upward velocity of measurement platform by ADCP bottom tracking"; W_BOTTOM:sdn_uom_urn = "SDN:P06::UVAA" ; W_BOTTOM:sdn_uom_name = "Metres per second" ;

			W_BOTTOM:units = “meters/second” W_BOTTOM:standard_name = “”; W_BOTTOM:ancillary_variables = "W_BOTTOM_SEADATANET_QC" ; W_BOTTOM:_FillValue = -99999.;
byte W_BOTTOM_SEADATANET_QC (INSTANCE,MAXT)	-	O	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC
float BOTTOM_DEPTH (INSTANCE,MAXT)	RNG_BOTTOM(N_DATE_TIME) + XOFF	O	BOTTOM_DEPTH:long_name = "Bottom depth” ; BOTTOM_DEPTH:sdn_parameter_urn = "SDN:P01::MBANZZZZ" ; BOTTOM_DEPTH:sdn_parameter_name = "Sea-floor depth (below instantaneous sea level) {bathymetric depth} in the water body" ; BOTTOM_DEPTH:sdn_uom_urn = "SDN:P06::ULAA" ; BOTTOM_DEPTH:sdn_uom_name = "Metres" ; BOTTOM_DEPTH:units = “meters” ; BOTTOM_DEPTH:standard_name = “Sea_floor_depth_below_sea_surface”; BOTTOM_DEPTH:ancillary_variables = "BOTTOM_DEPTH_SEADATANET_QC" ; BOTTOM_DEPTH:_FillValue = -99999.;
byte BOTTOM_DEPTH_SEADATANET_QC (INSTANCE,MAXT)	-	O	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC
float UCUR (INSTANCE,MAXT,MAXZ)	**UVEL_ADCP(N_DATE_TIME,N_LEVEL)**	**M**	UCUR:long_name = "Eastward absolute ADCP current velocity " ; UCUR:sdn_parameter_urn = "SDN:P01:: LCEWAS01" ; UCUR:sdn_parameter_name = " Eastward velocity of water current (Eulerian measurement) in the water body by shipborne acoustic doppler current profiler (ADCP)" ; UCUR:sdn_uom_urn = "SDN:P06::UVAA" ; UCUR:sdn_uom_name = "Metres per second" ; UCUR:units = “meters/second” ; UCUR:standard_name = “eastward_sea_water_velocity”; UCUR:ancillary_variables = "UCUR_SEADATANET_QC" ; UCUR:_FillValue = -99999.;
byte UCUR_SEADATANET_QC (INSTANCE,MAXT,MAXZ)	**-**	**M**	Values to be taken from CAS_CURRENT_FLAG with the following mapping: If CAS_CURRENT_FLAG = 1 => 1 Else if CAS_CURRENT_FLAG = 2 => 2 Else if CAS_CURRENT_FLAG = 3 => 2 Else if CAS_CURRENT_FLAG = 4 => 4 Else if CAS_CURRENT_FLAG = 5 => 4 Else if CAS_CURRENT_FLAG = 6 => 4 Else if CAS_CURRENT_FLAG = 7 => 9 Else if CAS_CURRENT_FLAG = 8 => 4 Else CAS_CURRENT_FLAG = 9 => all corresponding data deleted **Attributes:**

			… Idem than for TIME_SEADATANET_QC
float VCUR (INSTANCE,MAXT, MAXZ)	**VVEL_ADCP(N_DATE _TIME,N_LEVEL)**	**M**	**Attributes:** VCUR:long_name = "Northward absolute ADCP current velocity " ; VCUR:sdn_parameter_urn = "SDN:P01::LCNSAS01" ; VCUR:sdn_parameter_name = " Northward velocity of water current (Eulerian measurement) in the water body by shipborne Acoustic Doppler Current Profiler (ADCP)" ; VCUR:sdn_uom_urn = "SDN:P06::UVAA" ; VCUR:sdn_uom_name = "Metres per second" ; VCUR:units = “meters/second” ; VCUR:standard_name = “northward_sea_water_velocity”; VCUR:ancillary_variables = "VCUR_SEADATANET_QC" ; VCUR
byte VCUR_SEADATANET_QC (INSTANCE,MAXT,MAXZ)	**-**	**M**	**Values :** Cf parameter UCUR_SEADATANET_QC **Attributes:** … Idem than for TIME_SEADATANET_QC
float WCUR (INSTANCE,MAXT, MAXZ)	WVEL_ADCP(N_DATE _TIME,N_LEVEL)	O	**Attributes:** WCUR:long_name = "Vertical absolute ADCP current velocity" ; WCUR:sdn_parameter_urn = "SDN:P01::LRZAAS01" ; WCUR:sdn_parameter_name = "Upward velocity of water current in the water body by shipborne acoustic doppler current profiler (ADCP)" ; WCUR:sdn_uom_urn = "SDN:P06::UVAA" ; WCUR:sdn_uom_name = "Metres per second" ; WCUR:units = “meters/second” ; WCUR:standard_name = “upward_sea_water_velocity”; WCUR:ancillary_variables = "WCUR_SEADATANET_QC" ; WCUR:_FillValue = -99999.;
byte WCUR_SEADATANET_QC (INSTANCE,MAXT,MAXZ)	-	O	**Values :** Cf parameter UCUR_SEADATANET_QC **Attributes:** … Idem than for TIME_SEADATANET_QC
float ECUR(INSTANCE,MAXT, MAXZ)	EVEL_ADCP (N_DATE_TIME,N_LEV EL)	O	**Attributes:** ECUR:long_name = "Absolute ADCP current velocity error" ; ECUR:sdn_parameter_urn = "SDN:P01::LERRAS01" ; ECUR:sdn_parameter_name = "Error velocity of water current in the water body by shipborne acoustic doppler current profiler (ADCP)" ; ECUR:sdn_uom_urn = "SDN:P06::UVAA" ; ECUR:sdn_uom_name = "Metres per second" ; ECUR:units = “meters/second” ; ECUR:standard_name = “”; ECUR:ancillary_variables = "ECUR_SEADATANET_QC" ; ECUR:_FillValue = -99999.;
byte ECUR_SEADATANET_QC (INSTANCE,MAXT,MAXZ)	-	O	**Values :** Cf parameter UCUR_SEADATANET_QC

			Attributes: … Idem than for TIME_SEADATANET_QC
float PGOOD (INSTANCE,MAXT,MAXZ)	**PGOOD_ADCP (N_DATE_TIME,N_LEVEL)**	**M**	PGOOD:long_name = "Percentage of good data with 4 beams"; PGOOD:sdn_parameter_urn = "SDN:P01::PCGDSA01" ; PGOOD:sdn_parameter_name = "Acceptable proportion of acoustic signal returns {Percent Good} from the water body by shipborne acoustic doppler current profiler (ADCP)" ; PGOOD:sdn_uom_urn = "SDN:P06::UPCT" ; PGOOD:sdn_uom_name = "Percent" ; PGOOD:units = "percent" ; PGOOD:standard_name = "proportion_of_acceptable_signal_returns_from_acoustic_instrument_in_sea_water"; PGOOD:ancillary_variables = "PGOOD_SEADATANET_QC" ; PGOOD:_FillValue = -99999.;
byte PGOOD_SEADATANET_QC (INSTANCE,MAXT,MAXZ)	**-**	**M**	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC
float ECI (INSTANCE,MAXT,MAXZ)	**ECI(N_DATE_TIME,N_LEVEL)**	**M**	ECI:long_name = "Mean echo Intensity"; ECI:sdn_parameter_urn = "SDN:P01::TNIHSBAV" ; ECI:sdn_parameter_name = " Echo intensity of acoustic signal returns from the water body by shipborne acoustic doppler current profiler (ADCP) and averaging of all functional beams" ; ECI:sdn_uom_urn = "SDN:P06::UCNT" ; ECI:sdn_uom_name = "Counts" ; ECI:units = "count" ; ECI:standard_name = ""; ECI:ancillary_variables = "ECI_SEADATANET_QC" ; ECI:_FillValue = -99999.;
byte ECI_SEADATANET_QC (INSTANCE,MAXT,MAXZ)	**-**	**M**	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC
float ECI_B1 (INSTANCE,MAXT,MAXZ)	ECI_B1(N_DATE_TIME, N_LEVEL)	O	ECI_B1:long_name = "Mean echo Intensity for beam 1"; ECI_B1:sdn_parameter_urn = "SDN:P01::TNIHSB01" ; ECI_B1:sdn_parameter_name = "Echo intensity of acoustic signal returns from the water body by shipborne acoustic doppler current profiler (ADCP) beam 1" ; ECI_B1:sdn_uom_urn = "SDN:P06::UCNT" ; ECI_B1:sdn_uom_name = "Counts" ; ECI_B1:units = "count" ; ECI_B1:standard_name = ""; ECI_B1:ancillary_variables = "ECI_B1_SEADATANET_QC" ; ECI_B1:_FillValue = -99999.;
byte ECI_B1_SEADATANET_QC (INSTANCE,MAXT,MAXZ)	-	O	**Values:** All values set to 1 **Attributes:**

			… Idem than for TIME_SEADATANET_QC
float ECI_B2 (INSTANCE,MAXT,MAXZ)	ECI_B2(N_DATE_TIME, N_LEVEL)	O	ECI_B2:long_name = "Mean echo Intensity for beam 2”; ECI_B2:sdn_parameter_urn = "SDN:P01::TNIHSB02" ; ECI_B2:sdn_parameter_name = “Echo intensity of acoustic signal returns from the water body by shipborne acoustic doppler current profiler (ADCP) beam 2" ; ECI_B2:sdn_uom_urn = "SDN:P06::UCNT" ; ECI_B2:sdn_uom_name = "Counts" ; ECI_B2:units = “count” ; ECI_B2:standard_name = ““; ECI_B2:ancillary_variables = "ECI_B2_SEADATANET_QC" ; ECI_B2 :_FillValue = -99999.;
byte ECI_B2_SEADATANET_QC (INSTANCE,MAXT,MAXZ)	-	O	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC
float ECI_B3 (INSTANCE,MAXT,MAXZ)	ECI_B3(N_DATE_TIME, N_LEVEL)	O	ECI_B3:long_name = "Mean echo Intensity for beam 3”; ECI_B3:sdn_parameter_urn = "SDN:P01::TNIHSB03" ; ECI_B3:sdn_parameter_name = “Echo intensity of acoustic signal returns from the water body by shipborne acoustic doppler current profiler (ADCP) beam 3" ; ECI_B3:sdn_uom_urn = "SDN:P06::UCNT" ; ECI_B3:sdn_uom_name = "Counts" ; ECI_B3:units = “count” ; ECI_B3:standard_name = ““; ECI_B3:ancillary_variables = "ECI_B3_SEADATANET_QC" ; ECI_B3:_FillValue = -99999.;
byte ECI_B3_SEADATANET_QC (INSTANCE,MAXT,MAXZ)	-	O	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC
float ECI_B4 (INSTANCE,MAXT,MAXZ)	ECI_B4(N_DATE_TIME, N_LEVEL)	O	ECI_B4:long_name = "Mean echo Intensity for beam 4”; ECI_B4:sdn_parameter_urn = "SDN:P01:: TNIHSB04" ; ECI_B4:sdn_parameter_name = “Echo intensity of acoustic signal returns from the water body by shipborne acoustic doppler current profiler (ADCP) beam 4" ; ECI_B4:sdn_uom_urn = "SDN:P06::UCNT" ; ECI_B4:sdn_uom_name = "Counts" ; ECI_B4:units = “count” ; ECI_B4:standard_name = ““; ECI_B4:ancillary_variables = "ECI_B4_SEADATANET_QC" ; ECI_B4:_FillValue = -99999.;
byte ECI_B4_SEADATANET_QC (INSTANCE,MAXT,MAXZ)	-	O	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC

float CORR (INSTANCE,MAXT,MAXZ)	CORR(N_DATE_TIME, N_LEVEL)	O	CORR:long_name = "Correlation”; CORR:sdn_parameter_urn = "SDN:P01::CMAGZZ06" ; CORR:sdn_parameter_name = “ Correlation magnitude mean of acoustic signal returns from the water body by shipborne acoustic doppler current profiler (ADCP) and averaging of all functional beams" ; CORR:sdn_uom_urn = "SDN:P06::UCNT" ; CORR:sdn_uom_name = "Counts" ; CORR:units = “count” ; CORR:standard_name = ““; CORR:ancillary_variables = "CORR_SEADATANET_QC" ; CORR:_FillValue = -99999.;
byte CORR_SEADATANET_QC (INSTANCE,MAXT,MAXZ)	-	O	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC
float VERROR (INSTANCE,MAXT,MAXZ)	VERROR(N_DATE_TIME,N_LEVEL)	O	VERROR:long_name = “Horizontal Velocity precision”; VERROR:sdn_parameter_urn = "SDN:P01:VVERRIP1" ; VERROR:sdn_parameter_name = “Vector velocity error of water current in the water body by acoustic doppler current profiler (ADCP) and computation from manufacturer's stated ping standard deviation divided by the squared root of the number of good pings in an ensemble" ; VERROR:sdn_uom_urn = "SDN:P06::UVAA" ; VERROR:sdn_uom_name = "Metres per second" ; VERROR:units = “meters/second” ; VERROR:standard_name = ““; VERROR:ancillary_variables = "VERROR_SEADATANET_QC" ; VERROR:_FillValue = -99999.;;
byte VERROR_SEADATANET_QC (INSTANCE,MAXT,MAXZ)	-	O	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC
float BATHY (INSTANCE,MAXT)	BATHY(N_DATE_TIME)	O	BATHY:long_name = “Bottom depth given by a digital atlas”; BATHY:sdn_parameter_urn = "SDN:P01:MBANDVGR" ; BATHY:sdn_parameter_name = “Sea-floor depth (below mean sea level) {bathymetric depth} in the water body by derivation from a global grid" ; BATHY:sdn_uom_urn = "SDN:P06::ULAA" ; BATHY:sdn_uom_name = "Metres" ; BATHY:units = “meters” ; BATHY:standard_name = “Sea_floor_depth_below_sea_surface“; BATHY:ancillary_variables = "BATHY_SEADATANET_QC" ; BATHY:_FillValue = -99999.; BATHY:atlas=”[value to be taken from the long_name attribute field of BATHY in OceanSITES format]” ;
byte BATHY_SEADATANET_QC (INSTANCE,MAXT)	-	O	**Values:** All values set to 1 **Attributes:**

			… Idem than for TIME_SEADATANET_QC
float UTIDE (INSTANCE,MAXT)	U_TIDE(N_DATE_TIME)	O	UTIDE:long_name = “Eastward tide Velocity”; UTIDE:sdn_parameter_urn = "SDN:P01:LCEWTIM1" ; UTIDE:sdn_parameter_name = “ Eastward velocity of water current (tidal) in the water body by model prediction" ; UTIDE:sdn_uom_urn = "SDN:P06::UVAA" ; UTIDE:sdn_uom_name = "Metres per second" ; UTIDE:units = “meters/second” ; UTIDE:standard_name = ““; UTIDE:ancillary_variables = "UTIDE_SEADATANET_QC" ; UTIDE:_FillValue = -99999.; UTIDE:tidal_model=.=”[value to be taken from the :type_tide attribute field of U_TIDE in OceanSITES format]” ;
byte UTIDE_SEADATANET_QC (INSTANCE,MAXT,MAXZ)	-	O	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC
float VTIDE (INSTANCE,MAXT)	V_TIDE(N_DATE_TIME)	O	VTIDE:long_name = “Northward tide Velocity”; VTIDE:sdn_parameter_urn = "SDN:P01:LCNSTIM1" ; VTIDE:sdn_parameter_name = “ Northward velocity of water current (tidal) in the water body by model prediction" ; VTIDE:sdn_uom_urn = "SDN:P06::UVAA" ; VTIDE:sdn_uom_name = "Metres per second" ; VTIDE:units = “meters/second” ; VTIDE:standard_name = ““; VTIDE:ancillary_variables = "VTIDE_SEADATANET_QC" ; VTIDE:_FillValue = -99999.; VTIDE:tidal_model [value to be taken from the :type_tide attribute field of V_TIDE in OceanSITES format]” ;
byte VTIDE_SEADATANET_QC (INSTANCE,MAXT,MAXZ)	-	O	**Values:** All values set to 1 **Attributes:** … Idem than for TIME_SEADATANET_QC

Table 1 - Variable mapping between OceanSites and SeaDataNet formats

3.4. Access to CSR and EDMERP mapping

The web service developed by the SISCO team on the CSR catalogue which generates the *csrCodeList.xml* file when updating the vocabulary of Octopus will be adapted to add the EDMERP code(s) linked to the cruises (see with SISCO what has been implemented).

Octopus will also have to upload the edmerp.xml list of projects when updating the vocabularies (same web service as the one used by NEMO).

3.5. Global attributes

3.5.1. SeaDataNet mandatory global attributes

SeaDataNet has 4 mandatory global attributes which are:

:title = "SeaDataNet Shipborne ADCP data – converted from *[:SOFTWARE]* OceanSITES netCDF by OCTOPUS *[version; where [:SOFTWARE]* is a global attribute of the OceanSITES input file.

:Conventions = "SeaDataNet_1.0 CF-1.6" ;

:featureType = "trajectoryProfile" ;

:date_update = " YYYY-MM-DD hh:mm:ss+0200" (=system date);

3.5.2. Global attributes created from OceanSITES variables

The following additional global attributes listed in Table 2 are created from the variables of the input file with dimension set to CONST1= 1: or dimension set to N_DATE_TIME for some of them (MODE and VRMS_ADCP). Mandatory global attributes have the rule column set to Mandatory (M).

Variable name in OceanSITES	Global attribute name in SeaDataNet	Rule M:mandatory O:optional	Comment
TX_FREQUENCY	:ADCP_frequency	M	Use of TX_FREQUENCY [:longname] [:value] [:unit] *:ADCP_frequency = "ADCP transmitter frequency: 38 KiloHz";*
BEAM_ANGLE	:ADCP_beam_angle	O	Use of BEAM_ANGLE [:longname] [:value] [:unit] *:ADCP_beam_angle = "Beam Angle/vertical: 30 degrees";*
ADCP_ANGLE	:ADCP_ship_angle	O	Use of ADCP_ANGLE [:longname] [:value] [:unit] *:ADCP_angle_ship = "ADCP Angle/ship axis: 45 Degrees";*
BIN_LENGTH	:bin_length	M	Use of BIN_LENGTH [:longname] [:value] [:unit] *:bin_length = "Bin Length: 24 metres";*
MIDDLE_BIN1_DEPTH	:middle_bin1_depth	M	Use of MIDDLE_BIN1_DEPTH [:longname] [:value] [:unit] *:middle_bin1_depth = "Depth of first bin center : 47.92 metres";*
HEAD_MISLG	:heading_corr	O	Use of HEAD_MISLG [:longname] [:value] [:unit] *:heading_corr = "Heading Misalignment: 0 degrees";*

PITCH_MISLG	:pitch_corr	O	Use of PITCH_MISLG [:value] [:unit] :longname in the input OceanSITES files must not be taken, instead put the literal string « Pitch correction ». *:pitch_corr = "Pitch correction: 4.76 degrees";*
AMPLI_CORFAC	:ampli_corr	O	Use of AMPLI_CORFAC [:longname] [:value] For this parameter, there is no unit. *:ampli_corr = "Correction factor on velocity amplitude: 1.029";*
XOFF	:ADCP_depth	O	Use of XOFF [:longname] [:value] [:unit] **For this parameter there is no unit by mistake in the OceanSITES files "metres" has to be forced** *:ADCP_depth = "Transducer Depth: 6 meters";*
CORR_PR	:pitch_roll_used	O	Use of CORR_PR [:value] If value = 0 then no, else if value = 1, then yes *:pitch_roll_used = "yes";*
MODE	:acquisition_mode	O	Use of MODE [:value], ● If all values = 1 or fillvalue then "BroadBand", ● else if all values = 10 or fillValue then "NarrowBand" ● Else if values = 1 or 10 or fillValue then "BroadBand and NarrowBand alternatively", ● else if all values = fillValue "Not reported" *:acquisition_mode = "NarrowBand";*
VRMS_ADCP	:instr_error	O	Use of VRMS_ADCP [:longname] [:value] [:unit] The values are identical and the first value can be taken *:instr_error ="Ping Horizontal Velocity precision: 0.16 meter per second";*
FLAG3_HALF_WINDOW	:flag2_half_window	O	Use of the FLAG3_HALF_WINDOW [:value], for this parameter, there is no unit. Long-name is not provided in the input file, the literal value has to be set. *:flag2_half_window = "Flag 2 half width of the median filter window: 30";*
FLAG3_SCF_MED_DEV	:flag2_scf_med_dev	O	Use of the FLAG3_SCF_MED_DEV [:value], for this parameter, there is no unit. Long-name is not provided in the input file, the literal value has to be set *: flag2_scf_med_dev = " Flag 2 authorised number of standard deviations: 2.7";*
FLAG5_MAX_WVEL	:flag4_max_wvel	O	Use of the FLAG5_MAX_WVEL [:value] [:unit]. Long-name is not provided in the input file, the literal value has to be set *: flag4_max_wvel = "Flag 4 maximum horizontal velocity: 0.2 meter per second";*
FLAG5_PGOOD_MIN	:flag4_pgood_min	O	Use of the FLAG5_PGOOD_MIN [:value] [:unit]. Long-name is not provided in the input file, the literal value has to be set *:flag4_pgood_min = "Flag 4 minimum percent data with 4 beams: 10 percent";*
FLAG4_MAX_VSHEAR	:flag4_max_vshear	O	Use of the FLAG4_MAX_VSHEAR [:value] [:unit]. Long-name is not provided in the input file, the literal value has to be set *: flag4_max_vshear = "Flag 4 maximum vertical shear: 0.25 second-1";*
FLAG6_MAX_VVEL	:flag4_max_vvel	O	Use of the FLAG4_MAX_VVEL [:value] [:unit] Long-name is not provided in the input file, the literal value has to be set *:flag4_max_vvel = "Flag 4 maximum vertical velocity: 2 meter per second";*
FLAG6_MIN_CORR	:flag4_min_corr	O	Use of the FLAG4_MIN_CORR [:value], for this parameter, there is no unit. Long-name is not provided in the input file, the literal value has to be set

			:flag4_min_corr =”Flag 4 minimum correlation: 60” ;
FLAG8_BOTTOM	:flag4_bottom	O	Use of the FLAG8_BOTTOM [:value] For this parameter, there is no unit. ● If value = 1 then set the literal value “External bathymetry” ● else if value = 2 then set the literal value “ADCP bottom detection and external bathymetry when the bottom is out of reach of the ADCP “ ● else set the literal value “No detection” *:flag4_bottom= ”External bathymetry”;*

Table 2 - Global attributes created from OceanSITES variables - Examples in blue

3.5.3. Global attributes copied from OceanSITES

If present, the following global attributes should be copied from the global attributes of the input Cascade OceanSITES data file (Examples in blue): (use lowercase except for acronyms ADCP, CPU, PI):

- :date_creation = "25-Jul-2019" ;
- :CPU_firmware_version = 23 ;
- :CPU_firmware_revision = 17 ;
- :ADCP_form = "CONVEXE" ;
- :ADCP_facing = "DOWN" ;
- :ADCP_manufacturer = "RDI" ;
- :ADCP_type = "OceanSurveyor" ;
- :processing_chief= "GOURTAY" ;
- :data_type = "SADCP" ;
- :navigation_reference = "GPS Navigation" ;
- :heading_reference = "GPS Heading" ;
- :platform_number = "FHQB" ;
- :principal_investigator = "UNKNOWN" ;
- :PI_institut = "IFREMER" ;
- :PI_country = "FRANCE" ;
- :project = "UNKNOWN" ; If the project value is “UNKNOWN”, it should be replaced by the acronym(s)/name(s) of the project(s) linked to the cruise, separated by ‘;’. The information about the projects should be taken from the *edmerp.xml* (for the project acronym and name) and the *csrCodeList.xml* (for the mapping between EDMERP and CSR) files, note that Acronym is not mandatory and could be empty. If no EDMERP codes are linked to the cruise, then UNKNOWN can be kept.

4. OCTOPUS checker of the SDN trajectoryProfile netcdf format

A checker of the new format created for the ADCP has been implemented in Octopus.

It must be based on the netCDF trajectory checker with some special features:

When opening the file the data type can be detected by reading the *:featureType* and the :data_type global attributes and by that the global attribute *:Conventions* is equal to"SeaDataNet_1.0 CF-1.6" ;

- Then OCTOPUS writes the following message: "Input file is a CFPOINT [:data_type] [:featureType] (Example: Input file is a CFPOINT SADCP trajectoryProfile file.
- The check file for netCDF used by OCTOPUS checker has been updated ([5] SDC_WP9_OctopusNetCDFFormatCheckerV2.xlsx).

5. SADCP Viewer: Improved Currents Visualisation for End Users via an Interactive and Reproducible Web Interface

5.1. Rationale

End users can download EuroGoShip data from SeaDataNet. To generate more interest among end users, we need to increase visibility and lower the entry barriers for using SADCP data. To support this, we have developed a web application called SADCP Viewer, which finalises our data processing pathways for user adoption.

It is crucial for end users to interactively explore the data without the need to download it. Therefore, we utilise the Pangeo environment, which SADCP Viewer heavily relies on. The Pangeo ecosystem, created by a community of engineers and geoscientists, addresses the challenges of big data geoscience. Pangeo is interactive, scalable, and enables open, reproducible, and scalable big data geoscience for everyone [reference to the interactive Pangeo paper].

SADCP Viewer comprises various tools implemented in the software package xsadcp. Using Xarray [9], xsadcp can extract current data from SADCP NetCDF files stored in SeaDataNet in a cloud-optimised format, Zarr [10]. The software is developed in a reproducible environment using Jupyter, hvPlot, Matplotlib, and Docker. In addition to the web application, researchers and engineers can install xsadcp on their laptops, cloud, or HPC infrastructure to run Jupyter notebooks and explore different visualisations using Pangeo's capabilities. New developments can be easily updated through the GitHub environment and published as web applications in cloud environments, such as Hugging Face Spaces [https://huggingface.co/spaces/SADCPVIEW/SADCP_VIEWER].

Recommended Practices:

- **Reproducible Workflow:** Ensure workflows are reproducible for consistency and reliability.
- **Recommended Data Format:** Use cloud-native data formats, such as Zarr, to facilitate easier data access for end users.

5.2. Technical description

5.2.1. Technologies

micromamba

We used the Micromamba package manager, which allows installing Python packages in virtual environments but works much faster than the Anaconda distribution. We installed Python 3.11 and Jupyter Lab within an environment named 'xsadcp'.

jupyter lab

We used the Jupyter Lab user interface, which allows coding and testing programs and has the unique feature of generating notebooks where text and code execution results can be integrated.

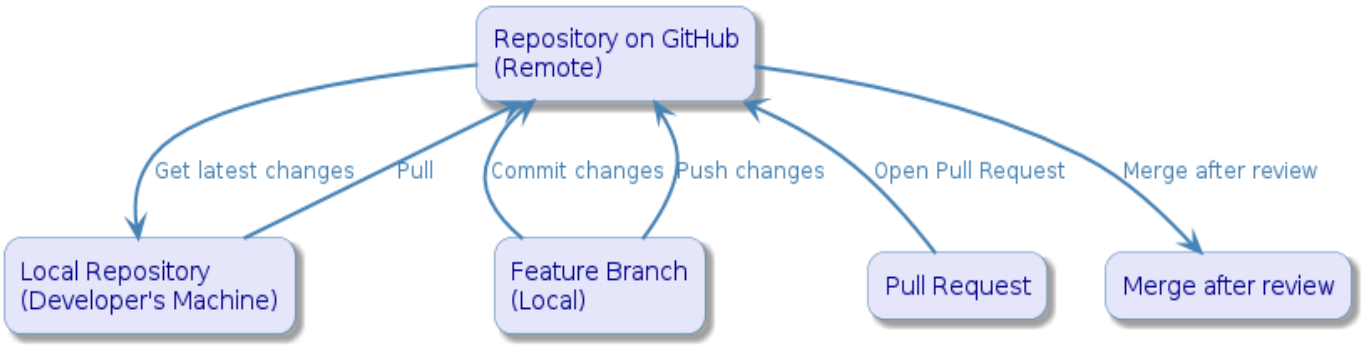
GitHub

All the programs, along with explanatory documents, have been uploaded to the GitHub platform. This platform allows storing code projects and managing them with the Git control system. Jupyter Lab integrates Git functionality.

The usual steps to collaborate on a software project using Git and GitHub are as follows:

1. Local Repository (Developer's Machine):
    - This is the developer's local repository where they work on their computer.
    - The developer retrieves the latest changes from the remote repository (Repository on GitHub) using `Pull` to ensure their local repository is up to date.
2. Feature Branch (Local):
    - The developer creates a new local branch to work on a new feature or a fix.
    - Changes made to this branch are committed (Commit changes) to create snapshots of the progress made.
3. Repository on GitHub (Remote):
    - Once the changes are complete, the developer sends these changes to the GitHub repository (`Push changes`).
    - This creates an updated version of the branch in the remote repository.
4. Pull Request:
    - On GitHub, the developer opens a Pull Request, meaning they ask for their changes to be reviewed and merged into the project's main branch.
5. Merge After Review:
    - After the changes have been reviewed and approved, they are merged (`Merge`) into the main branch of the remote GitHub repository.

Here is a diagram that illustrates these steps:



HuggingFace

To enable access to our application via the web, we used the Hugging Face platform. Hugging Face Hub is a centralized web platform that provides different functionalities:

- Git-based code repositories, including discussions and pull requests for projects.
- Models, also utilizing Git-based version control.
- Datasets, primarily consisting of text, images, and audio.
- Web applications ("spaces" and "widgets"), designed for small-scale demonstrations of machine learning applications.

Spaces are hosted web environments where web applications can be hosted.

We created the 'SADCP_View' space for our final application.

The space consists of an 'App' tab where the application runs and a 'Files' tab.
The latter contains the following elements :

1. Source Code: The source code of the application in .py format (not visible in our SADCP_View space).
2. Configuration Files: Configuration files needed for deploying and configuring the application, such as `requirements.txt` to specify Python dependencies or `Dockerfile` for Docker images.
3. Data Files: If the application requires data files to function, they can also be included in the "Files" tab.
4. Documentation Files: Documentation files such as `README.md` to provide an overview of the application, or `LICENSE` files to define licenses.

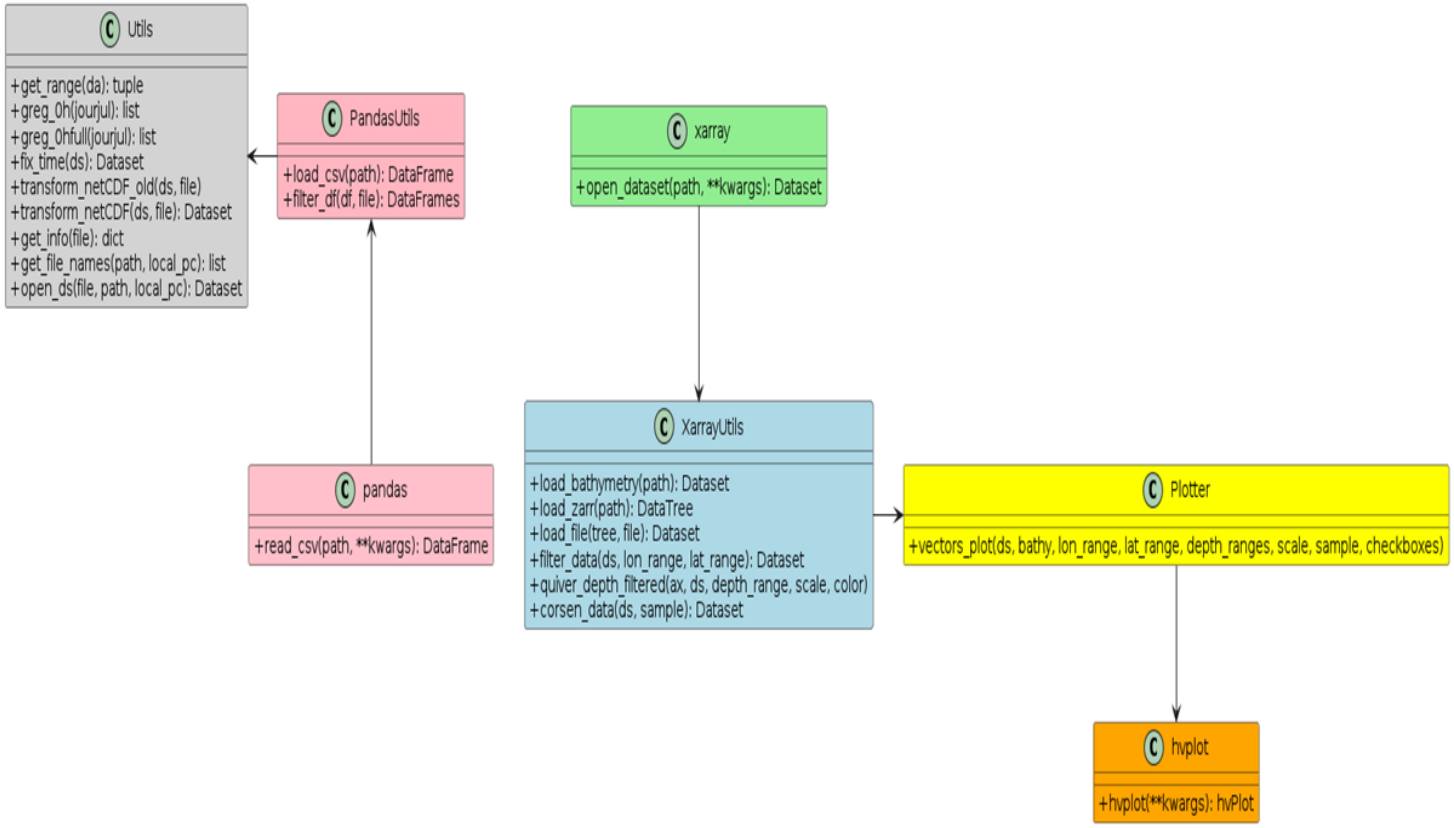
5.2.2. Application Architecture

The Python application is composed of three different modules:

- `util.py`, which includes all the used functions.
- `init.py`, which initialises the functions so they can be used in the main program.
- `app.py`, which is the main program.

Additionally, we also created the `create_csv_jarr` program to generate a CSV file containing metadata

module util.py



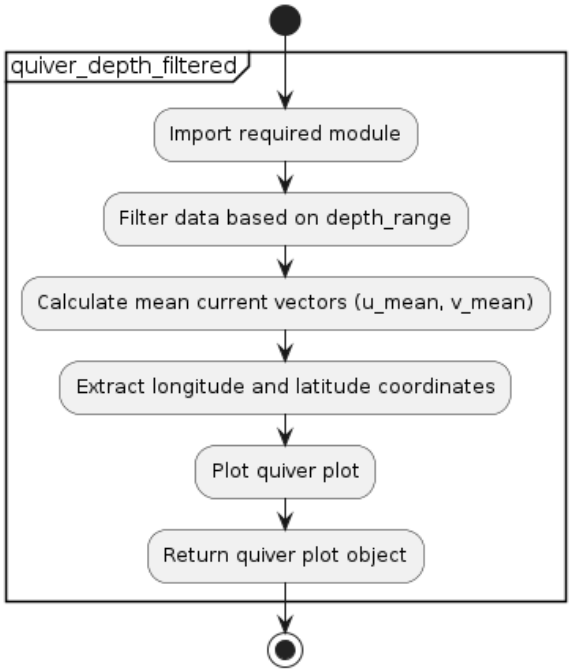
This diagram represents the class diagram of the util module.

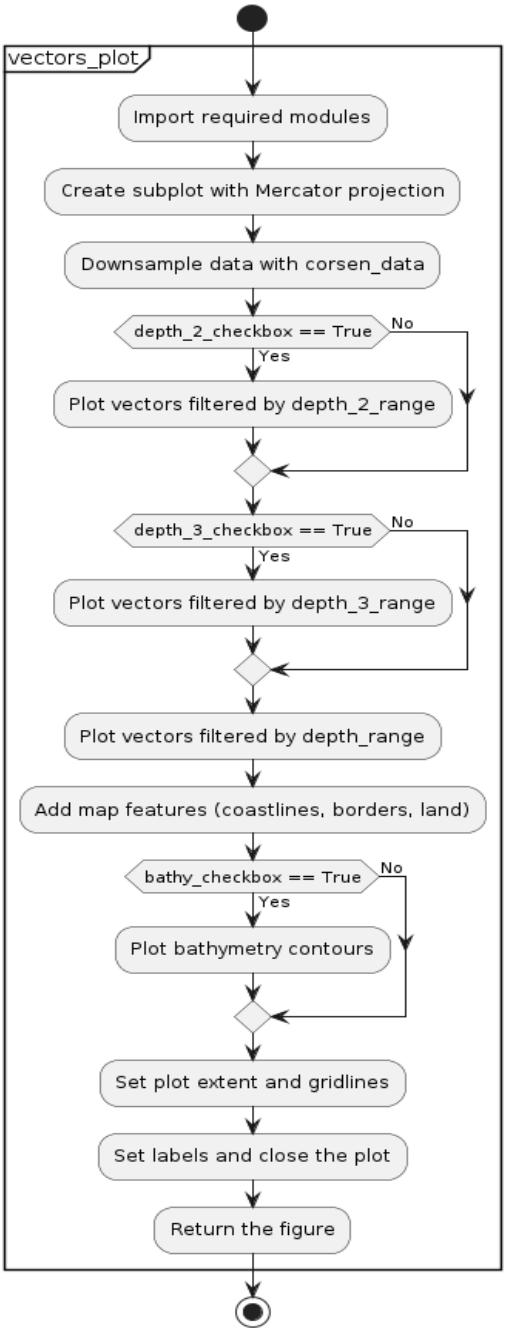
Here is how it works :

1. Data Loading and Preparation:
    - `get_range(da)`: Calculates the rounded interval (min, max) of a dataset.
    - `load_csv(path)`: Loads a CSV file containing metadata about data files.
    - `load_bathymetry(path)`: Loads a NetCDF file containing bathymetric data.
    - `load_zarr(path)`: Loads a dataset in Zarr format.
    - `load_file(tree, selected_file)`: Loads a specific file from a Zarr data tree.
    - `filter_df(sorted_df, selected_file)`: Filters a DataFrame based on the selected file name.
    - `filter_data(ds, longitude_range, latitude_range)`: Filters data based on latitude and longitude ranges.
2. Data Visualization and Analysis:
    - `quiver_depth_filtered(ax, ds, depth_range, scale_factor, color)`: Plots a vector graph filtered by depth.
    - `bathy_uship_vship_bottom_depth(ds)`: Plots graphs for different variables over time.
    - `corsen_data(ds, sample)`: Resamples the data to reduce the temporal resolution.
    - `vectors_plot(...)`: Creates a vector map based on different parameters.
3. Time Conversion:
    - `greg_0h(jourjul)` and `greg_0hfull(jourjul)`: Convert Julian days into Gregorian dates.
    - `fix_time(ds)`: Fixes and corrects the time in a dataset.
4. Data Transformation:
    - `transform_netCDF_old(ds, selected_file)`: Transforms the NetCDF file according to the old format.
    - `transform_netCDF(ds, selected_file)`: Transforms the NetCDF file according to the new format.
5. Information Extraction:

- `get_info(file_name)`: Extracts detailed metadata from a data file.
- `get_file_names(path, local_pc)`: Retrieves file names from a given path.
- `open_ds(file_name, base_path, local_pc)`: Opens a data file with specific configurations.

The diagrams below respectively represent the functioning of the `quiver_depth_filtered` and `vectors_plots` functions, which are used for data visualisation :

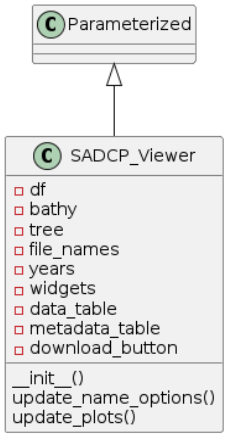




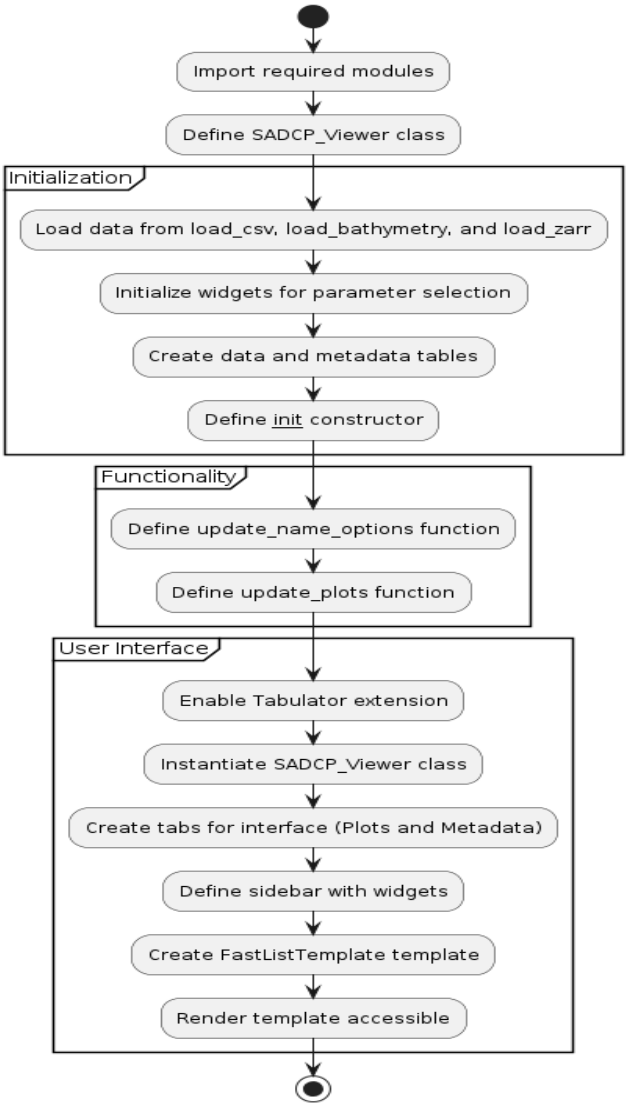
module app.py

This Python code uses the Panel framework to create an interactive web application for visualising and analysing ADCP data by using widgets. We created a `SADCP_Viewer` class to more easily initialise and manipulate the widgets.

This diagram represents the class diagram of the `app.py` module. The `SADCP_Viewer` class inherits parameters from the `Parameterized` class



Here is another diagram representing the overall functioning

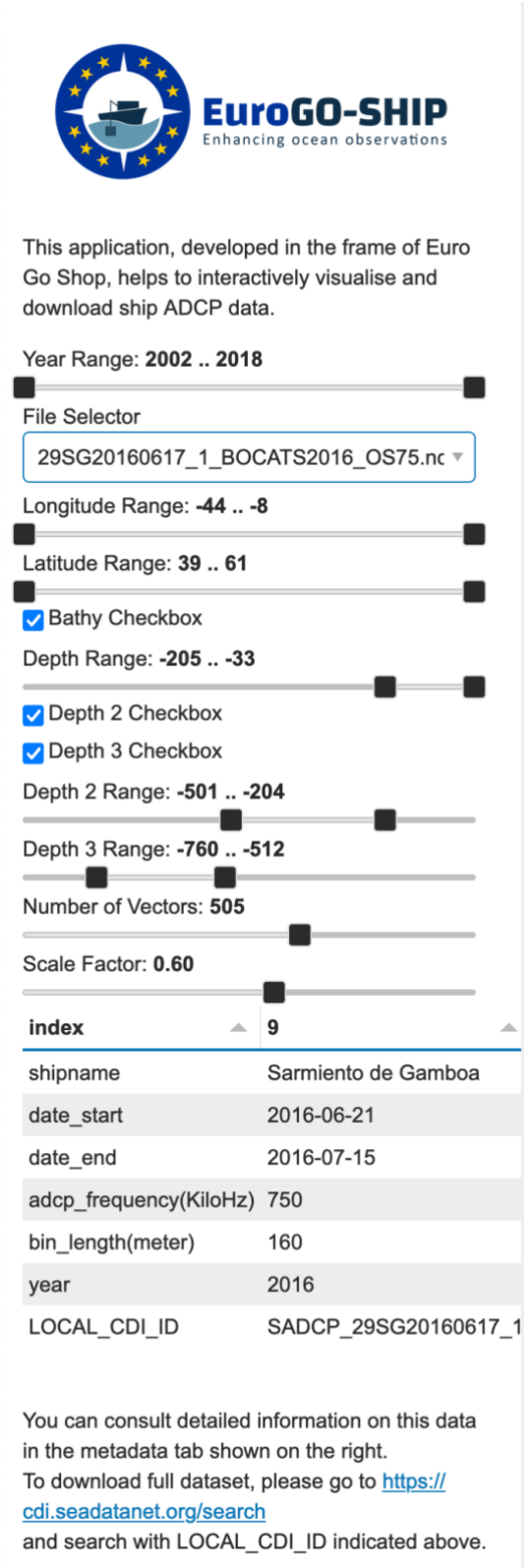


Here is an explanation of how it works:

1. Imports and Data Loading:
    - `import panel as pn`: Importing the Panel framework to create interactive dashboards.
    - `import param`: Importing the Param module to manage the widget parameters.
    - `from xsadcp import...`: Importing specific data processing functions from the xsadcp module.
    - Data Loading: The `load_csv`, `load_bathymetry`, and `load_zarr` functions load data from various sources and cache them for faster access.
2. Definition of the SADCP_Viewer Class:
    - Parameters and Widgets: The class defines widgets to allow users to select years, files, longitude/latitude ranges, and other parameters related to ADCP data.
    - `__init__`: Initialises the class and updates the file selector options.
    - `update_name_options`: Updates the selection options and filters the data based on the selected years and file.
    - `update_plots`: Updates the graphs based on the selected parameters, filters the data, and then displays the graphs.
3. Creating the User Interface:
    - Control Widgets: The widgets defined in the class (like sliders and selectors) are used to control the graph parameters.
    - Dashboard Creation: The graphs and tables are organized into columns for display together. The dashboard is structured with tabs (plots, metadata) and a sidebar with information about the application.
4. Displaying the Application:
    - The application is displayed using `template.servable()`, which allows it to be hosted as a web application.

5.3. How to use the SADCP viewer

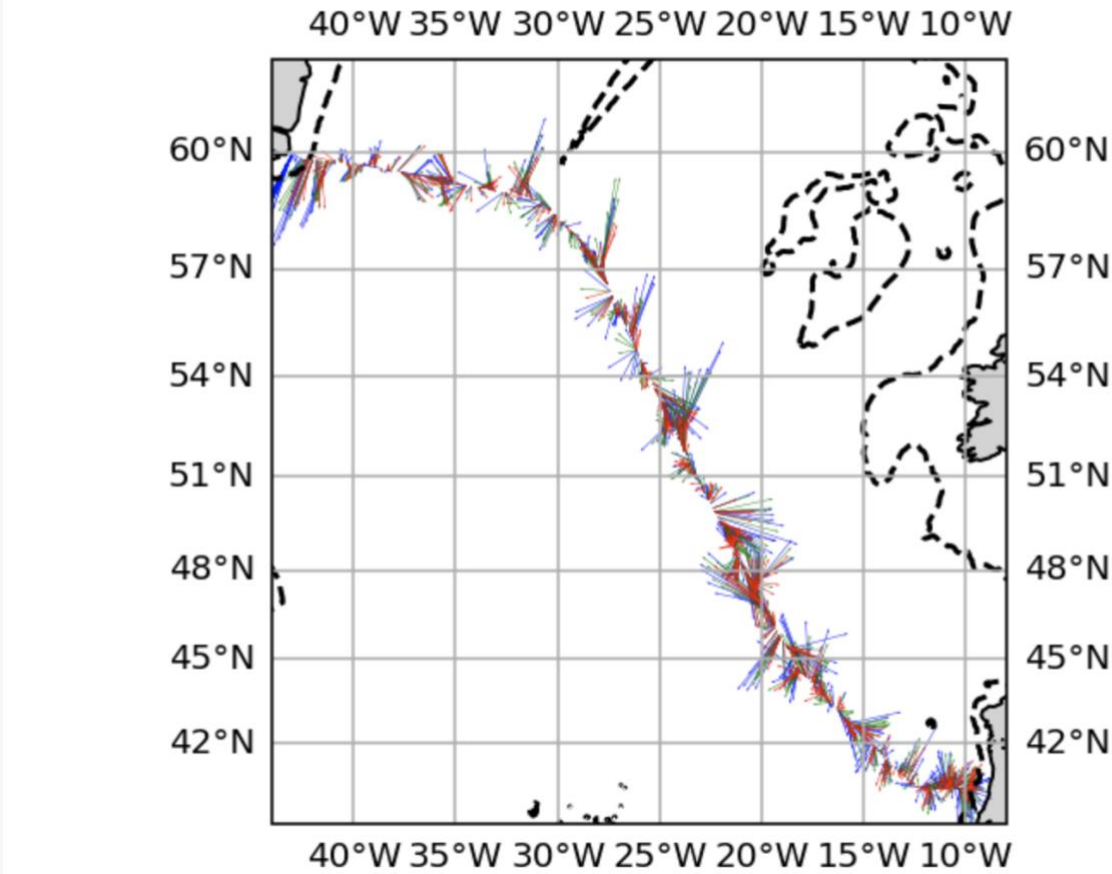
First go to https://huggingface.co/spaces/SADCPVIEW/SADCP_VIEWER

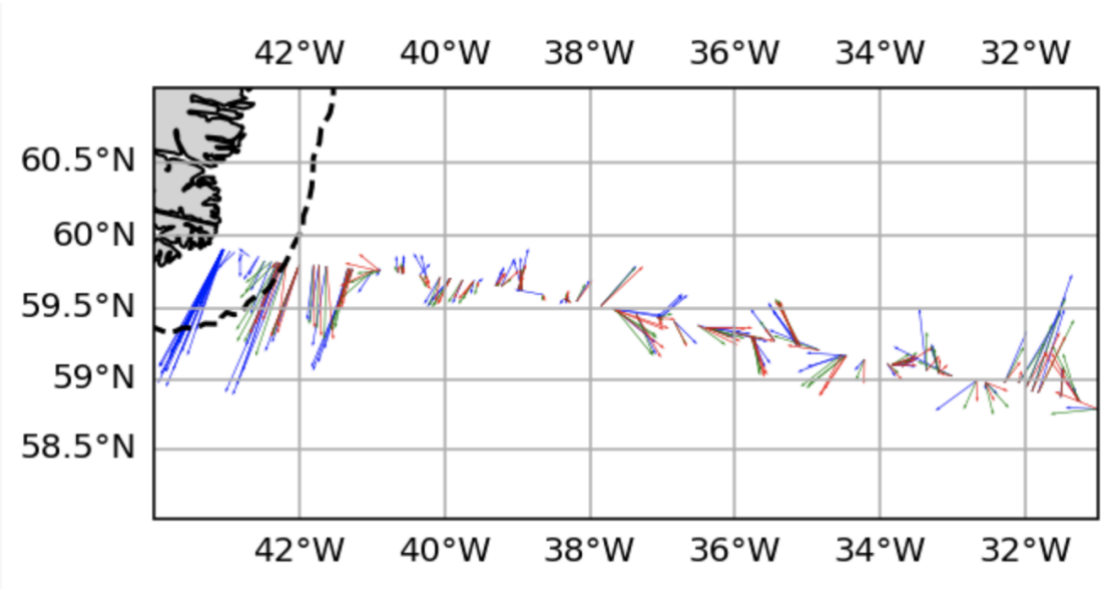


The left panel helps you to control what you want to see:

- The Year Range selects the files displayed in the File Selector according to the year of the cruises
- The File Selector displays all the files available in the chosen year range
- Longitude Range and Latitude Range zoom on the chosen region (be patient)
- Bathy toggle displays the 1000-meter isobath (more in a future version)
- By default, the shown current vectors are currents averaged over the full range of the ADCP. You can change by sliding the Depth Range to the desired values (here 33 to 205m). This layer will be displayed in blue.
- If you want other layers, toggle Depth 2 (green vectors) and Depth 3 (red vectors) and adapt Depth2 Range and Depth 3 Range to the desired values (here 204 to 501m depth and 512 to 760m depth)
- Too see more data, adapt the Number of Vectors. Note that data were subsampled at a 1-hour rate to increase the speed of the app.
- To change the length of the arrows, adapt the Scale Factor
- Most relevant metadata are summarised at the bottom of the left panel (still a little correction on frequency and bin_length that need to be divided by 10)
- To download the data, follow the link and search for the LOCAL_CDI_ID in SeaDataNet.

Plots Metadata





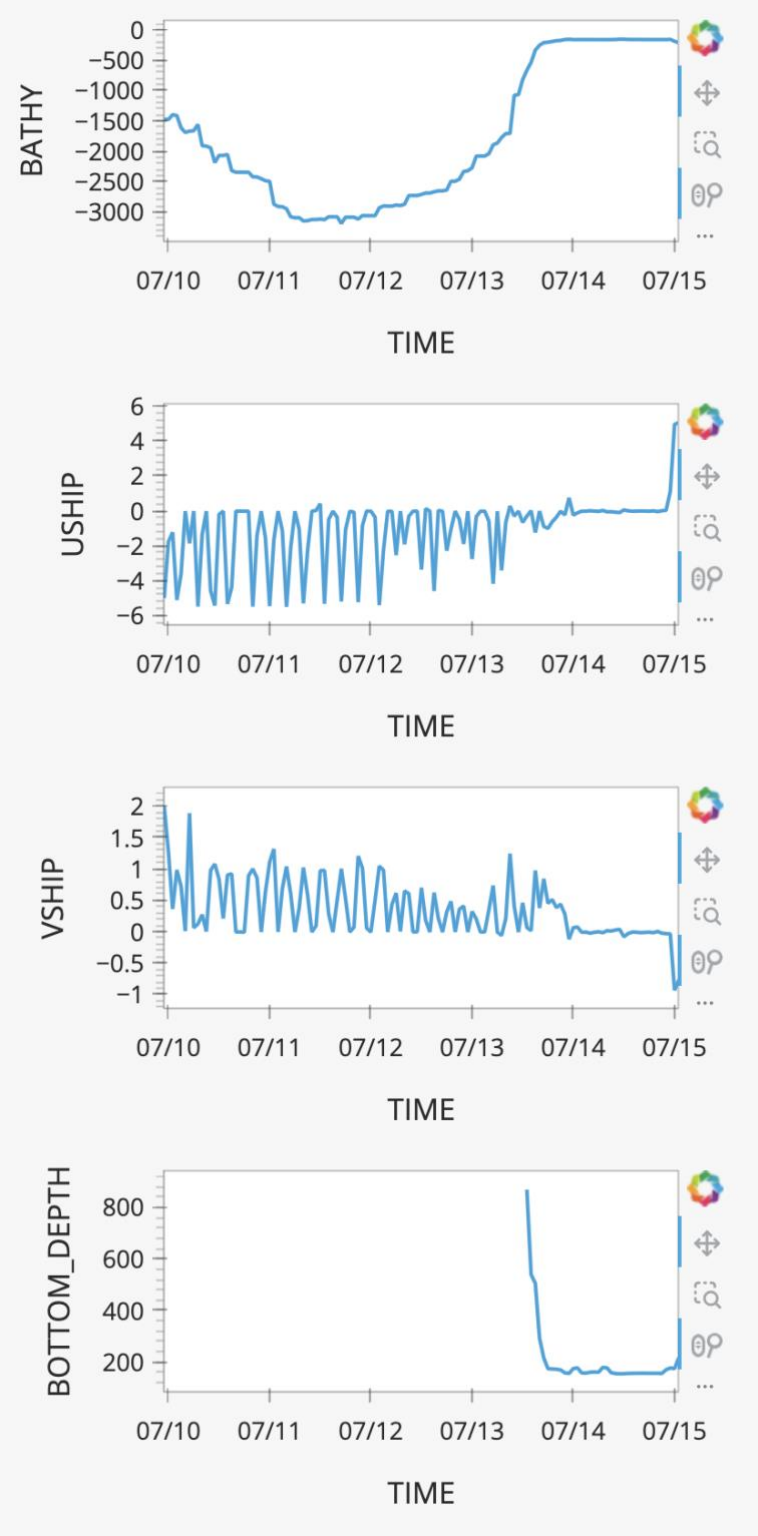
The central panel is a vector plot on the map according to your selection.

index	9
file_name	29SG20160617_1_BOCATS2016_OS75.nc
filename	29SG20160617_1_BOCATS2016_OS75.nc
shipname	Sarmiento de Gamboa
date_start	2016-06-21
date_end	2016-07-15
adcp_frequency(KiloHz)	750
bin_length(meter)	160
year	2016
title	SeaDataNet Shipborne ADCP data - Converted from Cascade Exploita
Conventions	SeaDataNet_1.0 CF-1.6
featureType	trajectoryProfile
date_update	2024-04-19T14:55:34+02:00
ADCP_frequency	ADCP transmitter frequency: 75.0 KiloHz
ADCP_beam_angle	Beam Angle/vertical: 30.0 degrees
ADCP_ship_angle	ADCP Angle/ship axis: 45.13 degrees
bin_length	Bin Length: 16.0 meter
middle_bin1_depth	Depth of first bin center: 23.98 meters
heading_corr	Heading Misalignment: 0.0 degrees
pitch_corr	Pitch correction: 2.5 degrees
ampli_corr	Correction factor on velocity amplitude: 1.0
pitch_roll_used	yes
date_creation	15-Jul-2016
ADCP_type	Ocean Surveyor
data_type	SADCP
user_interface_url	https://cdi.seadatanet.org/report/edmo/2489/SADCP_29SG20160617_
LOCAL_CDI_ID	SADCP_29SG20160617_1

It offers the possibility to see more information on the data in the “Metadata” tab:

Here again, the LOCAL_CDI_ID is recalled to find the data in SeaDataNet.

Finally, the right panel gives you more information of the ship velocity and the bathymetry below the ship (in the selected region), as a function of TIME (without the year)



BATHY is the bottom depth as stored in the SADCP data file (interpolated from an external bathymetry file)

USHIP is the eastward velocity of the ship

VSHIP if the northward velocity of the ship

BOTTOM_DEPTH is the bottom depth as detected by the SADCP when the bottom track mode is on (here on the Greenland shelf).

This is a demonstrator based on 12 datasets that were successfully converted in the SeaDataNet format using the OCTOPUS toolbox. We will progressively enrich the database that will be automatically detected by the SADCP viewer. In particular, as said before, we are working on the conversion of files that are in standard CODAS short format.

We took this opportunity to correctly connect the GO-SHIP datasets with the GO-SHIP project.

6. References

[1] OceanSITES data management team (2014). OceanSITES Data Format Reference Manual. NetCDF Conventions and Reference Tables. Ifremer. https://doi.org/10.13155/36149

[2] Kermabon Catherine, Lherminier Pascale, Le Bot Philippe (2023). **CASCADE V7.2: Software for processing, qualifying and visualizing SADCP data. User's guide**. https://doi.org/10.13155/100082

[3] Lowry Roy, Fichaut Michele, Schlitzer Reiner, Maudire Gilbert, Bregent Sophie, Gatti Julie (2024) - **SeaDataNet Datafile formats: ODV, MEDATLAS, netCDF**. Deliverable D8.5, 69 p. https://doi.org/10.13155/56547

[4] S. Brégent, M. Fichaut, S. Crouzille (2016) – **Octopus user manual**, 15 p. https://www.seadatanet.org/content/download/698/file/SDN_OCTOPUS_UserManual.pdf

[5] S. Brégent, M. Fichaut Excel file describing the checks to be done on netCDF file by OCTOPUS:https://www.seadatanet.org/content/download/2109/file/SDC_WP9_OctopusNetCDFFormatChecker.xlsx

[6] Tina Odaka, Pascale Lherminier, Léa Franc (2024): SADCP Viewer. https://doi.org/10.5281/zenodo.11071218

[7] Odaka Tina, Banihirwe Anderson, Eynard-Bontemps Guillaume, Ponte Aurelien, Maze Guillaume, Paul Kevin, Baker Jared, Abernathey Ryan (2019). The Pangeo Ecosystem: Interactive Computing Tools for the Geosciences: Benchmarking on HPC . Juckeland G., Chandrasekaran S. (eds) Tools and Techniques for High Performance Computing. HUST 2019, SE-HER 2019, WIHPC 2019. Communications in Computer and Information Science, vol 1190. Springer, Cham. Print ISBN 978-3-030-44727-4 Online ISBN 978-3-030-44728-1. https://doi.org/10.1007/978-3-030-44728-1_12. pp.190-204 . https://archimer.ifremer.fr/doc/00597/70946/

[8] Fouilloux Anne, Odaka Tina, Coca-Castro Alejandro, Luna-Valero Sebastian, Marasco Pier Lorenzo, Eynard-Bontemps Guillaume (2023). Pangeo@EOSC: Enabling Open, Reproducible, and Scalable Big Data Geoscience for Everyone . European Commission, Joint Research Centre, Soille, P., Lumnitz, S., Albani, S., Proceedings of the 2023 conference on Big Data from Space (BiDS'23) – From foresight to impact – 6-9 November 2023, Austrian Center, Vienna. Soille, P., Lumnitz, S., Albani, S. (Eds), Publications Office of the European Union, 2023. https://doi.org/10.2760/46796 pp.81-84 .

[9] Hoyer, S. & Hamman, J., (2017). xarray: N-D labeled Arrays and Datasets in Python. Journal of Open Research Software. 5(1), p.10. DOI: https://doi.org/10.5334/jors.148

Hoyer, S., Roos, M., Joseph, H., Magin, J., Cherian, D., Fitzgerald, C., Hauser, M., Fujii, K., Maussion, F., Imperiale, G., Clark, S., Kleeman, A., Nicholas, T., Kluyver, T., Westling, J., Munroe, J., Amici, A., Barghini, A., Banihirwe, A., … Wolfram, P. J. (2024). xarray (v2024.05.0). Zenodo. https://doi.org/10.5281/zenodo.11183201

[10] Alistair Miles, John Kirkham, Martin Durant, James Bourbeau, Tarik Onalan, Joe Hamman, Zain Patel, shikharsg, Matthew Rocklin, raphael dussin, Vincent Schut, Elliott Sales de Andrade, Ryan Abernathey, Charles Noyes, sbalmer, pyup.io bot, Tommy Tran, Stephan Saalfeld, Justin Swaney, … Anderson Banihirwe. (2020). zarr-developers/zarr-python: v2.4.0 (v2.4.0). Zenodo. https://doi.org/10.5281/zenodo.3773450