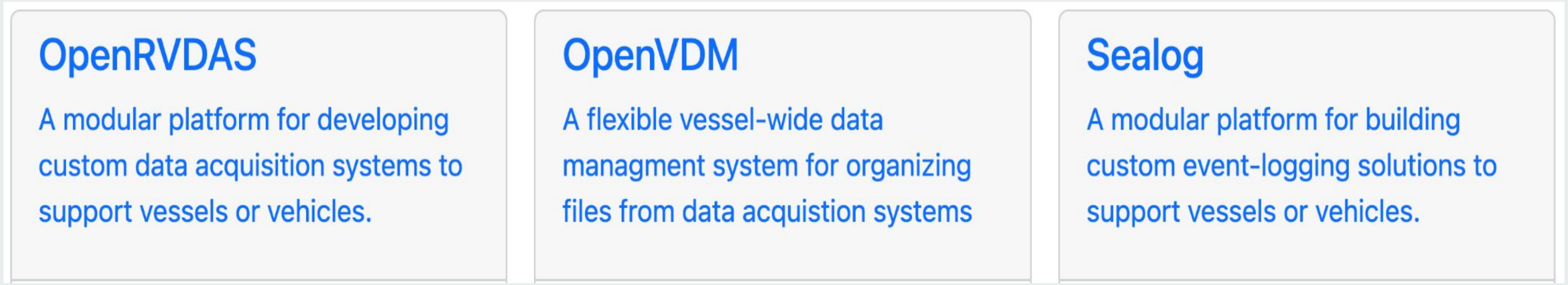
Ocean Data Tools

https://www.oceandatatools.org

RVTEC 2024 Tutorial



OpenRVDAS

A modular platform for developing custom data acquisition systems to support vessels or vehicles.

OpenVDM

A flexible vessel-wide data managment system for organizing files from data acquistion systems

Sealog

A modular platform for building custom event-logging solutions to support vessels or vehicles.

OpenVDM

- **Introduction** - what/why/where
- **Lingo 101**
- **Whole system overview** - installation, Web-UI tour, defining/controlling transfers
- **Hooks** - attaching processes to key points in processes
- **Displaying data** - plugins and parsers
- **Leveraging OpenVDM data elsewhere**
- **Best practices**
- **Contributing**
- **Where to from here?**

OpenVDM

- **Introduction - what/why/where**
- Lingo 101
- Whole system overview - installation, Web-UI tour, defining/controlling transfers
- Hooks - attaching processes to key points in processes
- Displaying data - plugins and parsers
- Leveraging OpenVDM data elsewhere
- Best practices
- Contributing
- Where to from here?

OpenVDM

- **What is it?**
- What's special about it?
- What can it do?

Tool for retrieving files from across a vessel, organizing those files into a single directory structure and ***providing*** crew and clients with ***safe and immediate access to datasets***.

Identifies file naming issues to ***prevent problems from propagating***.

OpenVDM

- **What is it?**
- What's special about it?
- What can it do?

Provides ability to ***setup bespoke data processing workflows*** based on the arrival of files.

Updates data deliverables throughout cruise to ***reduce end-of-cruise workload***.

Allows vessel operators to better ***adhere to*** their data management plan and ***best practices of*** archival facilities such as ***R2R***.

OpenVDM

- What is it?
- **What's special about it?**
- What can it do?

Flexible - the tool doesn't tell the user how to organize their data or the mechanism by which the files are retrieved

Centralized - Provides a single interface for defining and controlling the flow of files

Simple - Nothing to install on the systems creating the data

OpenVDM

- What is it?
- What's special about it?
- **What can it do?**

Retrieve data files from remote systems via:

- SMB (Windows) Share
- Rsync Server
- SSH Server
- Mounted NFS volumes
- Local directories

OpenVDM

- What is it?
- What's special about it?
- **What can it do?**

Replicate data files to remote systems via:

- SMB (Windows) Share
- Rsync Server
- SSH Server
- Mounted NFS volumes
- Mounted external devices (USB, fibre)

OpenVDM

- What is it?
- What's special about it?
- **What can it do?**

- Organize files based on the vessel operator defined schema
- Enforce vessel operator defined file naming conventions
- Assist in data file depending processing workflows (MBES Processing)

OpenVDM

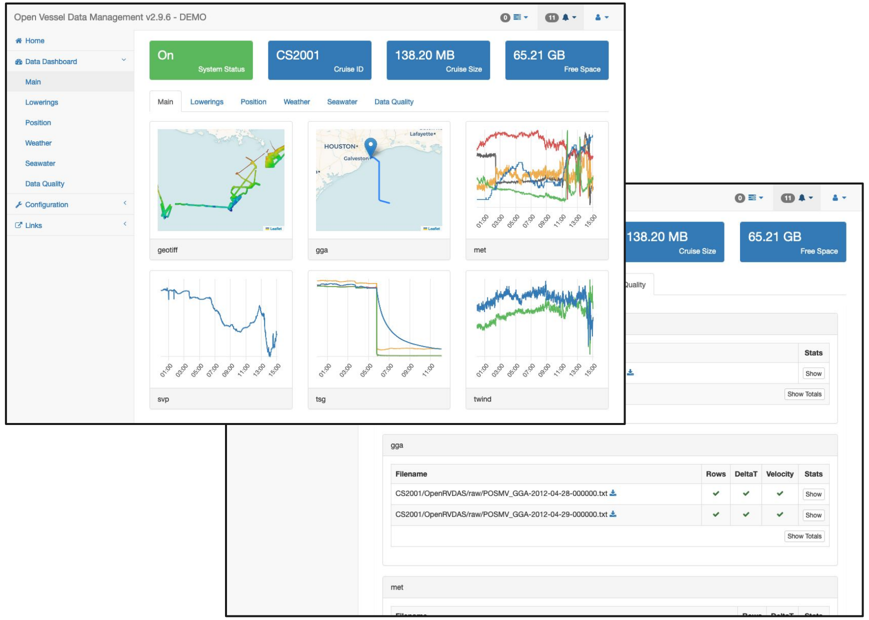
- What is it?
- What's special about it?
- **What can it do?**

- Talk to other systems (OpenRVDAS, Sealog, etc) at key cruise milestones such as Start/End of cruises
- Kick off automated processes when specific files arrive
- API for allowing independent processes to leverage OpenVDM configuration data

OpenVDM

- What is it?
- What's special about it?
- **What can it do?**

- Plugins and parsers for visualizing data files and running QA tests



gga

Filename	Rows	DeltaT	Velocity	Stats
CS2001/OpenRVDAS/raw/POSMV_GGA-2012-04-28-000000.txt	✓	✓	✓	Show
CS2001/OpenRVDAS/raw/POSMV_GGA-2012-04-29-000000.txt	✓	✓	✓	Show

Show Totals

met

OpenVDM Installations

OpenVDM

- Introduction - what/why/where
- **Lingo 101**
- Whole system overview - installation, Web-UI tour, defining/controlling transfers
- Hooks - attaching processes to key points in processes
- Displaying data - plugins and parsers
- Leveraging OpenVDM data elsewhere
- Best practices
- Contributing
- Where to from here?

Lingo 101

Collection System
System that creates files to be managed by OpenVDM

Collection System Transfer
Configuration data needed for OpenVDM to retrieve files from a local directory, locally mounted volume or remote data collection system

Lingo 101

Cruise Data Directory
Directory structure containing the data files collected during a given cruise

Cruise Data Transfer
Configuration data needed for OpenVDM to copy files from the cruise data directory to a local directory, locally mounted volume or remote data collection system

Lingo 101

Extra Directories
Directories to be created within the cruise data directory but not associated with a collection data transfer

Data Dashboard
Section of the Web UI that visualizes the output from the parsers including QA test results

Lingo 101

Hooks
Mechanism for adding additional processes at key milestones during a given cruise

Milestones

- Start of Cruise
- End of Cruise
- Post Data Transfer
- Post Data Dashboard

Lingo 101

Plugins
Optional script to process files from a specific collection system transfer

Parser
Used in conjunction with plugins to create web appropriate representations of raw data files, and define the QA tests to be run against the raw data files.

Lingo 101

Gearman
Job broker used schedule OpenVDM jobs

Worker
Instance of a script that can run one or more specific types of tasks

Task
A defined process

Job
A unit of work to complete a specific task

OpenVDM

- Introduction - what/why/where
- Lingo 101
- **Whole system overview - installation, Web-UI tour, defining/controlling transfers**
- Hooks - attaching processes to key points in processes
- Displaying data - plugins and parsers
- Leveraging OpenVDM data elsewhere
- Best practices
- Contributing
- Where to from here?

Installation - automated script

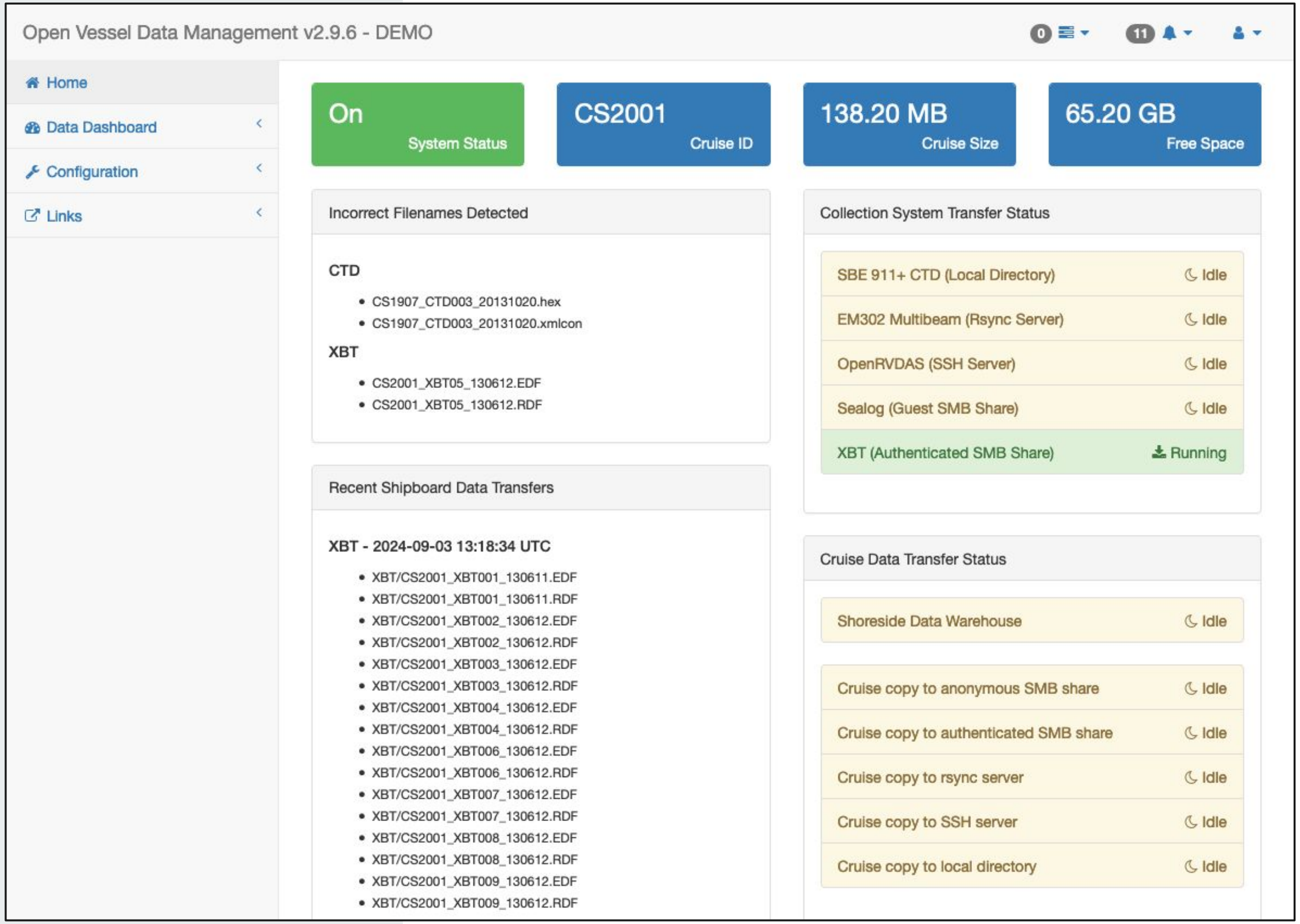
- Download the install script from OpenVDM GitHub repo
- Run the script

Built for Ubuntu but can be run on Rocky

Code Orientation

```
openvdm/
├── bin
├── database
├── docs
├── server
|_ ├── etc
|_ ├── lib
|_ ├── plugins
|_ |_ ├── parsers
|_ ├── workers
├── utils
├── www
```

The Web UI



Open Vessel Data Management v2.9.6 - DEMO

On
System Status

CS2001
Cruise ID

138.20 MB
Cruise Size

65.20 GB
Free Space

Incorrect Filenames Detected

CTD

- CS1907_CTD003_20131020.hex
- CS1907_CTD003_20131020.xmlcon

XBT

- CS2001_XBT05_130612.EDF
- CS2001_XBT05_130612.RDF

Recent Shipboard Data Transfers

XBT - 2024-09-03 13:18:34 UTC

- XBT/CS2001_XBT001_130611.EDF
- XBT/CS2001_XBT001_130611.RDF
- XBT/CS2001_XBT002_130612.EDF
- XBT/CS2001_XBT002_130612.RDF
- XBT/CS2001_XBT003_130612.EDF
- XBT/CS2001_XBT003_130612.RDF
- XBT/CS2001_XBT004_130612.EDF
- XBT/CS2001_XBT004_130612.RDF
- XBT/CS2001_XBT006_130612.EDF
- XBT/CS2001_XBT006_130612.RDF
- XBT/CS2001_XBT007_130612.EDF
- XBT/CS2001_XBT007_130612.RDF
- XBT/CS2001_XBT008_130612.EDF
- XBT/CS2001_XBT008_130612.RDF
- XBT/CS2001_XBT009_130612.EDF
- XBT/CS2001_XBT009_130612.RDF

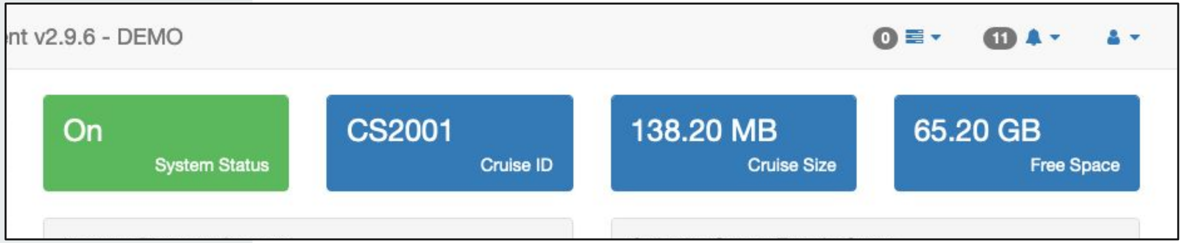
Collection System Transfer Status

SBE 911+ CTD (Local Directory)	Idle
EM302 Multibeam (Rsync Server)	Idle
OpenRVDAS (SSH Server)	Idle
Sealog (Guest SMB Share)	Idle
XBT (Authenticated SMB Share)	Running

Cruise Data Transfer Status

Shoreside Data Warehouse	Idle
Cruise copy to anonymous SMB share	Idle
Cruise copy to authenticated SMB share	Idle
Cruise copy to rsync server	Idle
Cruise copy to SSH server	Idle
Cruise copy to local directory	Idle

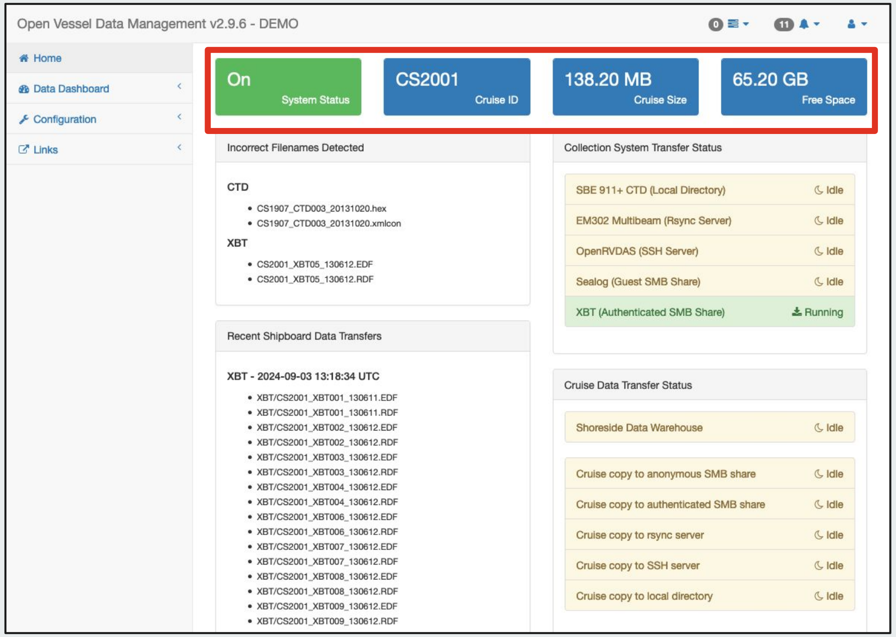
The Web UI



nt v2.9.6 - DEMO

0 · 11

On System Status	CS2001 Cruise ID	138.20 MB Cruise Size	65.20 GB Free Space



Open Vessel Data Management v2.9.6 - DEMO

0 · 11

On System Status	CS2001 Cruise ID	138.20 MB Cruise Size	65.20 GB Free Space

Incorrect Filenames Detected

CTD

- CS1907_CTD003_20131020.hex
- CS1907_CTD003_20131020.xmlcon

XBT

- CS2001_XBT05_130612.EDF
- CS2001_XBT05_130612.RDF

Recent Shipboard Data Transfers

XBT - 2024-09-03 13:18:34 UTC

- XBT/CS2001_XBT001_130611.EDF
- XBT/CS2001_XBT001_130611.RDF
- XBT/CS2001_XBT002_130612.EDF
- XBT/CS2001_XBT002_130612.RDF
- XBT/CS2001_XBT003_130612.EDF
- XBT/CS2001_XBT003_130612.RDF
- XBT/CS2001_XBT004_130612.EDF
- XBT/CS2001_XBT004_130612.RDF
- XBT/CS2001_XBT006_130612.EDF
- XBT/CS2001_XBT006_130612.RDF
- XBT/CS2001_XBT007_130612.EDF
- XBT/CS2001_XBT007_130612.RDF
- XBT/CS2001_XBT008_130612.EDF
- XBT/CS2001_XBT008_130612.RDF
- XBT/CS2001_XBT009_130612.EDF
- XBT/CS2001_XBT009_130612.RDF

Collection System Transfer Status

SBE 911+ CTD (Local Directory)	Idle
EM302 Multibeam (Rsync Server)	Idle
OpenRVDAS (SSH Server)	Idle
Sealog (Guest SMB Share)	Idle
XBT (Authenticated SMB Share)	Running

Cruise Data Transfer Status

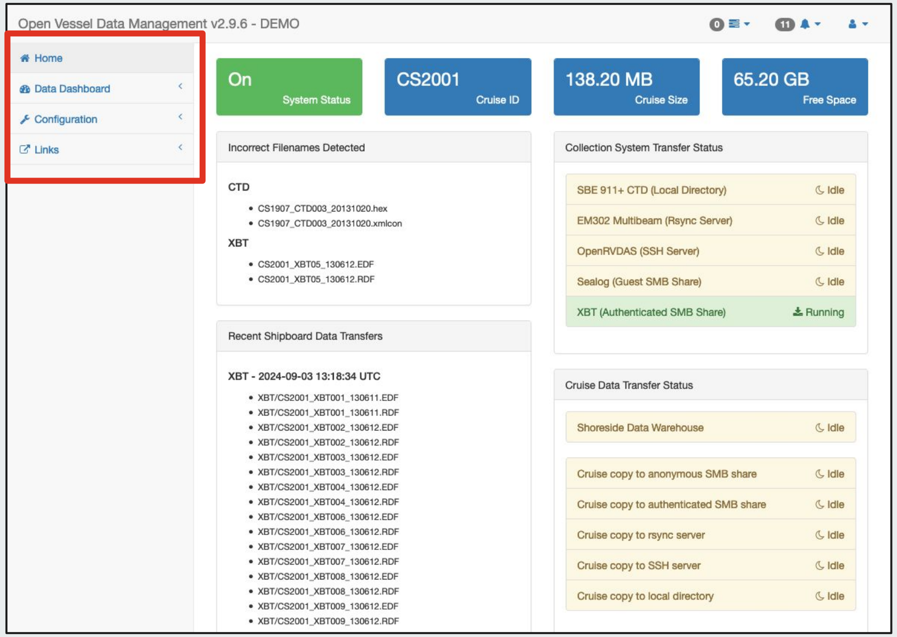
Shoreside Data Warehouse	Idle
Cruise copy to anonymous SMB share	Idle
Cruise copy to authenticated SMB share	Idle
Cruise copy to rsync server	Idle
Cruise copy to SSH server	Idle
Cruise copy to local directory	Idle

Topbar Navigation

Status Panels

- System Status
- Cruise ID
- Cruise SIze
- Free Space

The Web UI



Open Vessel Data Management v2.9.6 - DEMO

0 11

- Home
- Data Dashboard
- Configuration
- Links

On
System Status

CS2001
Cruise ID

138.20 MB
Cruise Size

65.20 GB
Free Space

Incorrect Filenames Detected

CTD

- CS1907_CTD003_20131020.hex
- CS1907_CTD003_20131020.xmlcon

XBT

- CS2001_XBT05_130612.EDF
- CS2001_XBT05_130612.RDF

Recent Shipboard Data Transfers

XBT - 2024-09-03 13:18:34 UTC

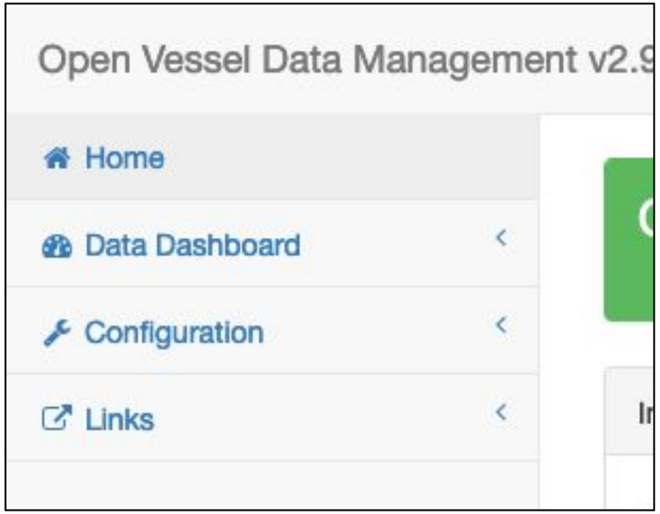
- XBT/CS2001_XBT001_130611.EDF
- XBT/CS2001_XBT001_130611.RDF
- XBT/CS2001_XBT002_130612.EDF
- XBT/CS2001_XBT002_130612.RDF
- XBT/CS2001_XBT003_130612.EDF
- XBT/CS2001_XBT003_130612.RDF
- XBT/CS2001_XBT004_130612.EDF
- XBT/CS2001_XBT004_130612.RDF
- XBT/CS2001_XBT006_130612.EDF
- XBT/CS2001_XBT006_130612.RDF
- XBT/CS2001_XBT007_130612.EDF
- XBT/CS2001_XBT007_130612.RDF
- XBT/CS2001_XBT008_130612.EDF
- XBT/CS2001_XBT008_130612.RDF
- XBT/CS2001_XBT009_130612.EDF
- XBT/CS2001_XBT009_130612.RDF

Collection System Transfer Status

SBE 911+ CTD (Local Directory)	Idle
EM302 Multibeam (Rsync Server)	Idle
OpenRVDAS (SSH Server)	Idle
Sealog (Guest SMB Share)	Idle
XBT (Authenticated SMB Share)	Running

Cruise Data Transfer Status

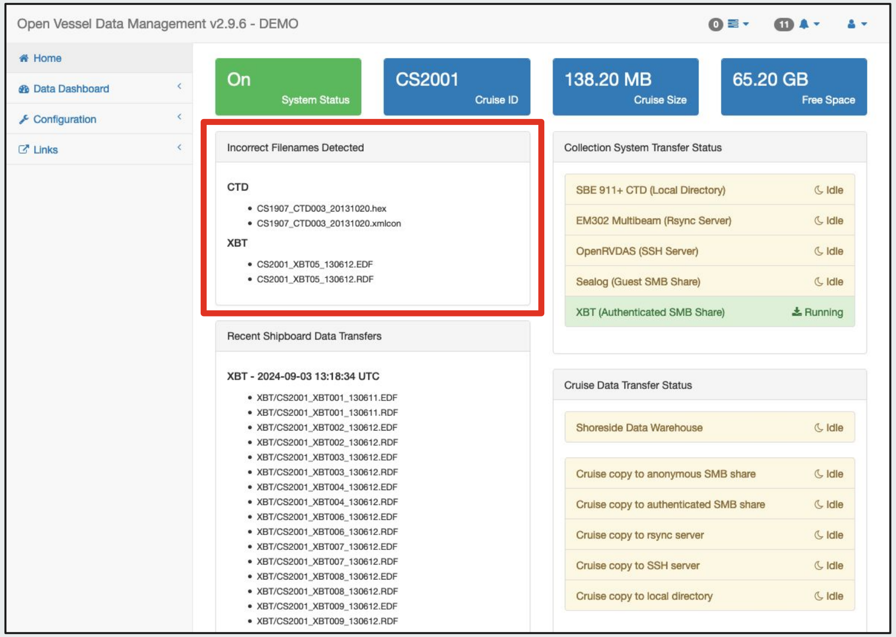
Shoreside Data Warehouse	Idle
Cruise copy to anonymous SMB share	Idle
Cruise copy to authenticated SMB share	Idle
Cruise copy to rsync server	Idle
Cruise copy to SSH server	Idle
Cruise copy to local directory	Idle



Sidebar Navigation

- Links to all areas of the Web UI

The Web UI


Open Vessel Data Management v2.9.6 - DEMO

On — System Status
CS2001 — Cruise ID
138.20 MB — Cruise Size
65.20 GB — Free Space

Incorrect Filenames Detected

CTD
- CS1907_CTD003_20131020.hex
- CS1907_CTD003_20131020.xmlcon

XBT
- CS2001_XBT05_130612.EDF
- CS2001_XBT05_130612.RDF

Collection System Transfer Status

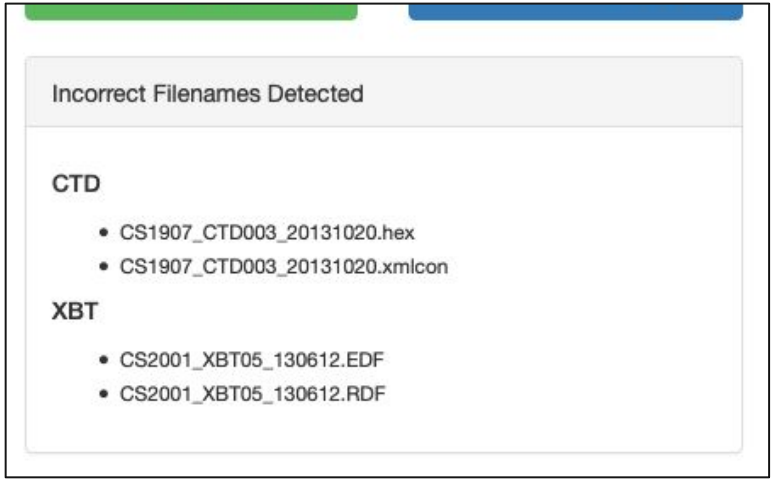
SBE 911+ CTD (Local Directory)	Idle
EM302 Multibeam (Rsync Server)	Idle
OpenRVDAS (SSH Server)	Idle
Sealog (Guest SMB Share)	Idle
XBT (Authenticated SMB Share)	Running

Recent Shipboard Data Transfers

XBT - 2024-09-03 13:18:34 UTC
- XBT/CS2001_XBT001_130611.EDF
- XBT/CS2001_XBT001_130611.RDF
- XBT/CS2001_XBT002_130612.EDF
- XBT/CS2001_XBT002_130612.RDF
- XBT/CS2001_XBT003_130612.EDF
- XBT/CS2001_XBT003_130612.RDF
- XBT/CS2001_XBT004_130612.EDF
- XBT/CS2001_XBT004_130612.RDF
- XBT/CS2001_XBT006_130612.EDF
- XBT/CS2001_XBT006_130612.RDF
- XBT/CS2001_XBT007_130612.EDF
- XBT/CS2001_XBT007_130612.RDF
- XBT/CS2001_XBT008_130612.EDF
- XBT/CS2001_XBT008_130612.RDF
- XBT/CS2001_XBT009_130612.EDF
- XBT/CS2001_XBT009_130612.RDF

Cruise Data Transfer Status

Shoreside Data Warehouse	Idle
Cruise copy to anonymous SMB share	Idle
Cruise copy to authenticated SMB share	Idle
Cruise copy to rsync server	Idle
Cruise copy to SSH server	Idle
Cruise copy to local directory	Idle

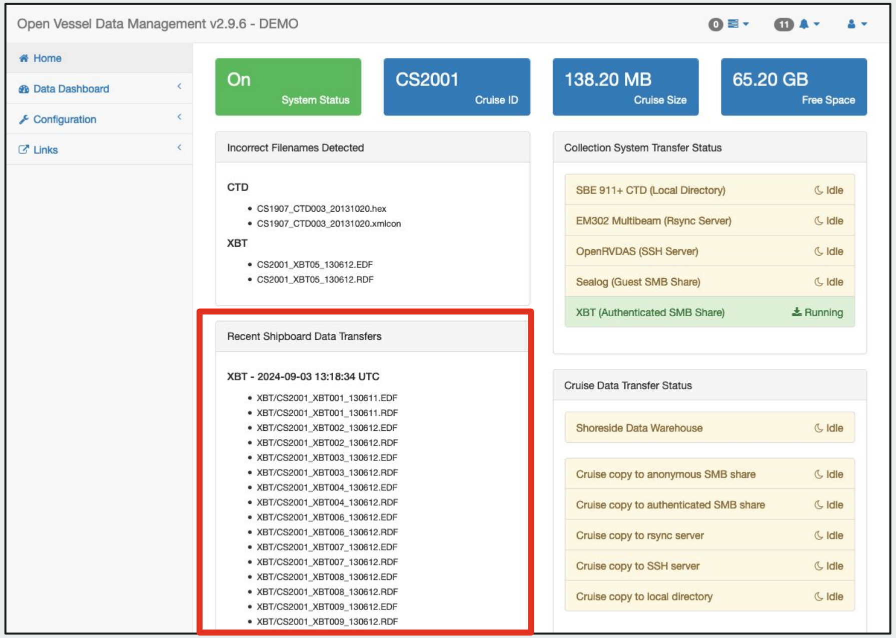

Incorrect Filenames Detected

CTD
- CS1907_CTD003_20131020.hex
- CS1907_CTD003_20131020.xmlcon

XBT
- CS2001_XBT05_130612.EDF
- CS2001_XBT05_130612.RDF

Incorrect Filenames
Card shows list of files whose names do not match the vessel’s file naming conventions for the given collection system transfer

The Web UI



Open Vessel Data Management v2.9.6 - DEMO

On System Status

CS2001 Cruise ID

138.20 MB Cruise Size

65.20 GB Free Space

Incorrect Filenames Detected

CTD

- CS1907_CTD003_20131020.hex
- CS1907_CTD003_20131020.xmlcon

XBT

- CS2001_XBT05_130612.EDF
- CS2001_XBT05_130612.RDF

Recent Shipboard Data Transfers

XBT - 2024-09-03 13:18:34 UTC

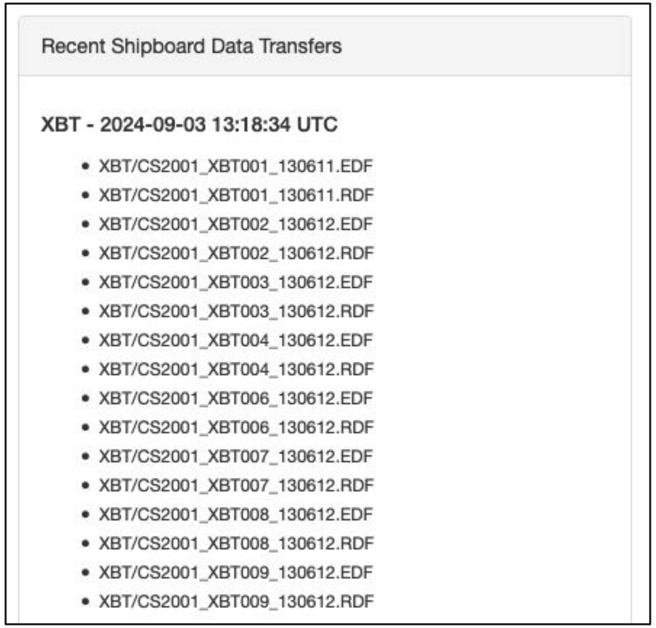
- XBT/CS2001_XBT001_130611.EDF
- XBT/CS2001_XBT001_130611.RDF
- XBT/CS2001_XBT002_130612.EDF
- XBT/CS2001_XBT002_130612.RDF
- XBT/CS2001_XBT003_130612.EDF
- XBT/CS2001_XBT003_130612.RDF
- XBT/CS2001_XBT004_130612.EDF
- XBT/CS2001_XBT004_130612.RDF
- XBT/CS2001_XBT006_130612.EDF
- XBT/CS2001_XBT006_130612.RDF
- XBT/CS2001_XBT007_130612.EDF
- XBT/CS2001_XBT007_130612.RDF
- XBT/CS2001_XBT008_130612.EDF
- XBT/CS2001_XBT008_130612.RDF
- XBT/CS2001_XBT009_130612.EDF
- XBT/CS2001_XBT009_130612.RDF

Collection System Transfer Status

SBE 911+ CTD (Local Directory)	Idle
EM302 Multibeam (Rsync Server)	Idle
OpenRVDAS (SSH Server)	Idle
Sealog (Guest SMB Share)	Idle
XBT (Authenticated SMB Share)	Running

Cruise Data Transfer Status

Shoreside Data Warehouse	Idle
Cruise copy to anonymous SMB share	Idle
Cruise copy to authenticated SMB share	Idle
Cruise copy to rsync server	Idle
Cruise copy to SSH server	Idle
Cruise copy to local directory	Idle



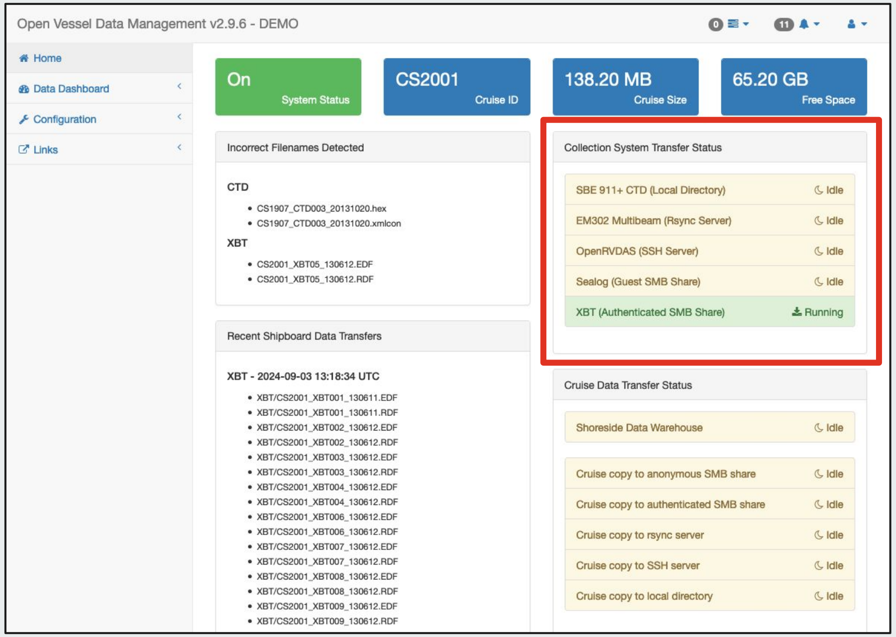
Recent Shipboard Data Transfers

XBT - 2024-09-03 13:18:34 UTC

- XBT/CS2001_XBT001_130611.EDF
- XBT/CS2001_XBT001_130611.RDF
- XBT/CS2001_XBT002_130612.EDF
- XBT/CS2001_XBT002_130612.RDF
- XBT/CS2001_XBT003_130612.EDF
- XBT/CS2001_XBT003_130612.RDF
- XBT/CS2001_XBT004_130612.EDF
- XBT/CS2001_XBT004_130612.RDF
- XBT/CS2001_XBT006_130612.EDF
- XBT/CS2001_XBT006_130612.RDF
- XBT/CS2001_XBT007_130612.EDF
- XBT/CS2001_XBT007_130612.RDF
- XBT/CS2001_XBT008_130612.EDF
- XBT/CS2001_XBT008_130612.RDF
- XBT/CS2001_XBT009_130612.EDF
- XBT/CS2001_XBT009_130612.RDF

Recent Shipboard Data Transfers
Card shows list of files recently transferred for the given collection system transfer

The Web UI



Open Vessel Data Management v2.9.6 - DEMO

On System Status

CS2001 Cruise ID

138.20 MB Cruise Size

65.20 GB Free Space

Incorrect Filenames Detected

CTD

- CS1907_CTD003_20131020.hex
- CS1907_CTD003_20131020.xmlcon

XBT

- CS2001_XBT05_130612.EDF
- CS2001_XBT05_130612.RDF

Recent Shipboard Data Transfers

XBT - 2024-09-03 13:18:34 UTC

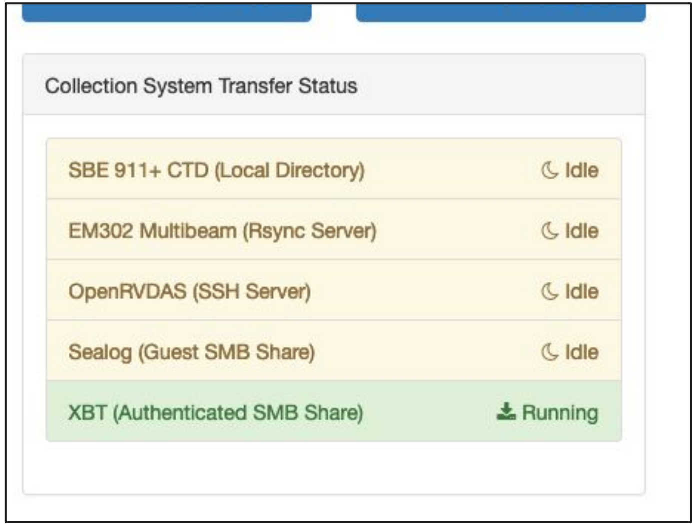
- XBT/CS2001_XBT001_130611.EDF
- XBT/CS2001_XBT001_130611.RDF
- XBT/CS2001_XBT002_130612.EDF
- XBT/CS2001_XBT002_130612.RDF
- XBT/CS2001_XBT003_130612.EDF
- XBT/CS2001_XBT003_130612.RDF
- XBT/CS2001_XBT004_130612.EDF
- XBT/CS2001_XBT004_130612.RDF
- XBT/CS2001_XBT006_130612.EDF
- XBT/CS2001_XBT006_130612.RDF
- XBT/CS2001_XBT007_130612.EDF
- XBT/CS2001_XBT007_130612.RDF
- XBT/CS2001_XBT008_130612.EDF
- XBT/CS2001_XBT008_130612.RDF
- XBT/CS2001_XBT009_130612.EDF
- XBT/CS2001_XBT009_130612.RDF

Collection System Transfer Status

SBE 911+ CTD (Local Directory)	Idle
EM302 Multibeam (Rsync Server)	Idle
OpenRVDAS (SSH Server)	Idle
Sealog (Guest SMB Share)	Idle
XBT (Authenticated SMB Share)	Running

Cruise Data Transfer Status

Shoreside Data Warehouse	Idle
Cruise copy to anonymous SMB share	Idle
Cruise copy to authenticated SMB share	Idle
Cruise copy to rsync server	Idle
Cruise copy to SSH server	Idle
Cruise copy to local directory	Idle



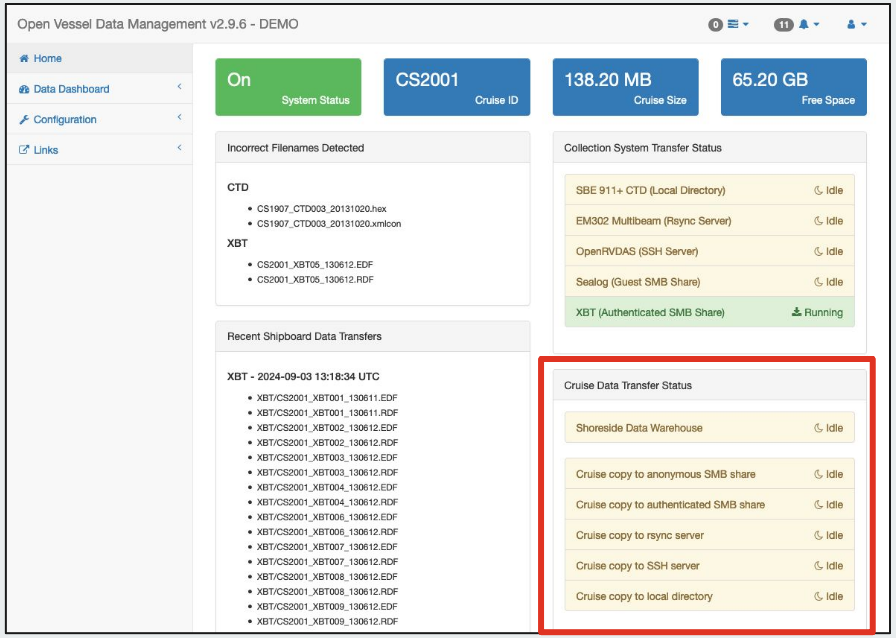
Collection System Transfer Status

SBE 911+ CTD (Local Directory)	Idle
EM302 Multibeam (Rsync Server)	Idle
OpenRVDAS (SSH Server)	Idle
Sealog (Guest SMB Share)	Idle
XBT (Authenticated SMB Share)	Running

Collection System Transfer Status

Card shows the status of the currently enabled collection system transfers

The Web UI



Open Vessel Data Management v2.9.6 - DEMO

Home
Data Dashboard
Configuration
Links

On System Status | CS2001 Cruise ID | 138.20 MB Cruise Size | 65.20 GB Free Space

Incorrect Filenames Detected

CTD
- CS1907_CTD003_20131020.hex
- CS1907_CTD003_20131020.xmlcon

XBT
- CS2001_XBT05_130612.EDF
- CS2001_XBT05_130612.RDF

Collection System Transfer Status

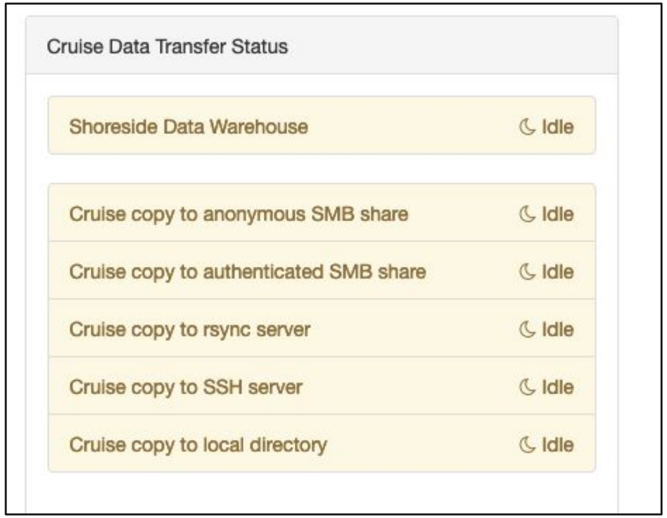
SBE 911+ CTD (Local Directory)	Idle
EM302 Multibeam (Rsync Server)	Idle
OpenRVDAS (SSH Server)	Idle
Sealog (Guest SMB Share)	Idle
XBT (Authenticated SMB Share)	Running

Recent Shipboard Data Transfers

XBT - 2024-09-03 13:18:34 UTC
- XBT/CS2001_XBT001_130611.EDF
- XBT/CS2001_XBT001_130611.RDF
- XBT/CS2001_XBT002_130612.EDF
- XBT/CS2001_XBT002_130612.RDF
- XBT/CS2001_XBT003_130612.EDF
- XBT/CS2001_XBT003_130612.RDF
- XBT/CS2001_XBT004_130612.EDF
- XBT/CS2001_XBT004_130612.RDF
- XBT/CS2001_XBT006_130612.EDF
- XBT/CS2001_XBT006_130612.RDF
- XBT/CS2001_XBT007_130612.EDF
- XBT/CS2001_XBT007_130612.RDF
- XBT/CS2001_XBT008_130612.EDF
- XBT/CS2001_XBT008_130612.RDF
- XBT/CS2001_XBT009_130612.EDF
- XBT/CS2001_XBT009_130612.RDF

Cruise Data Transfer Status

Shoreside Data Warehouse	Idle
Cruise copy to anonymous SMB share	Idle
Cruise copy to authenticated SMB share	Idle
Cruise copy to rsync server	Idle
Cruise copy to SSH server	Idle
Cruise copy to local directory	Idle



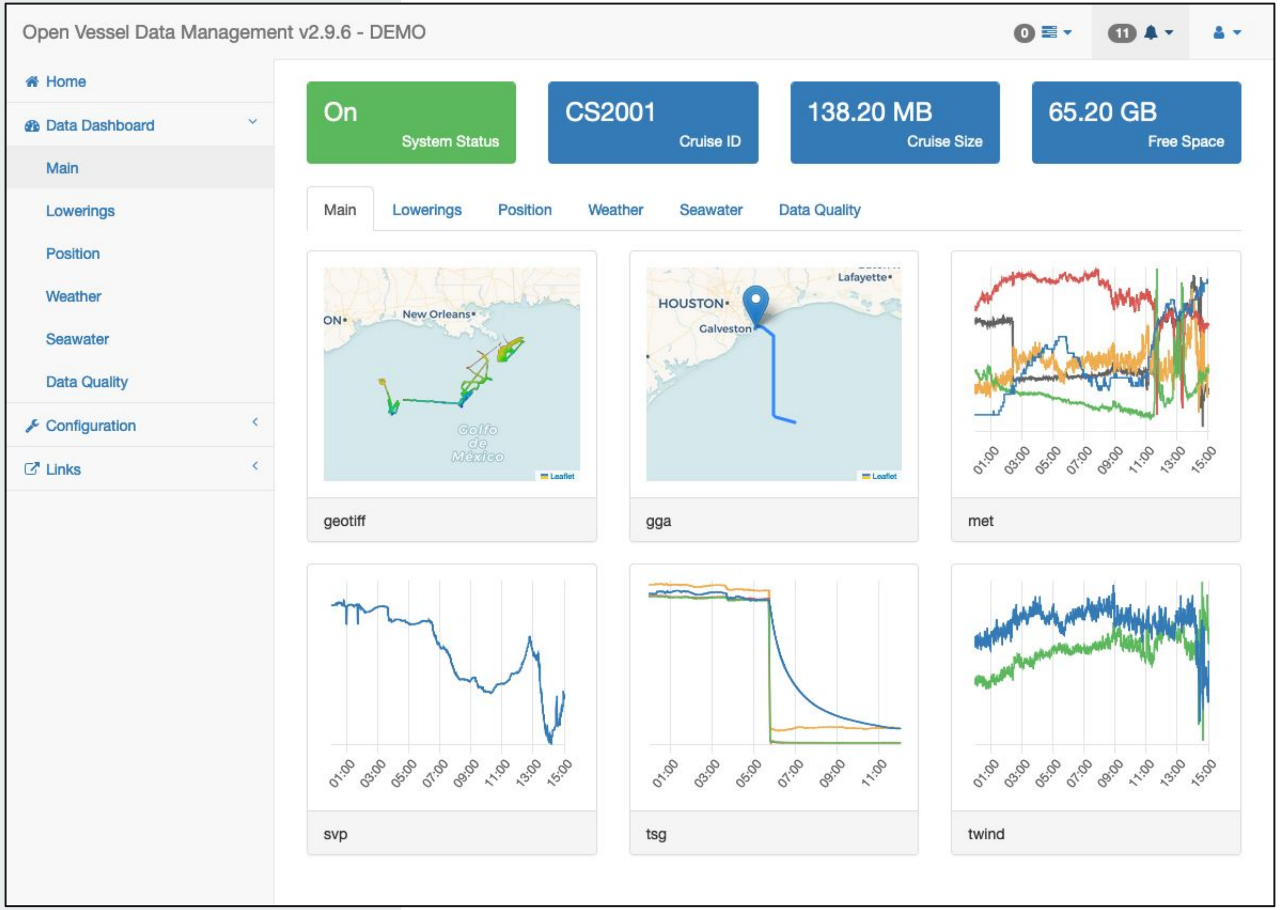
Cruise Data Transfer Status

Shoreside Data Warehouse	Idle
Cruise copy to anonymous SMB share	Idle
Cruise copy to authenticated SMB share	Idle
Cruise copy to rsync server	Idle
Cruise copy to SSH server	Idle
Cruise copy to local directory	Idle

Cruise Data Transfer Status
Card shows the status of the currently enabled cruise data transfers

The Web UI

Data Dashboard


Open Vessel Data Management v2.9.6 - DEMO

On System Status

CS2001 Cruise ID

138.20 MB Cruise Size

65.20 GB Free Space

geotiff

gga

met

svp

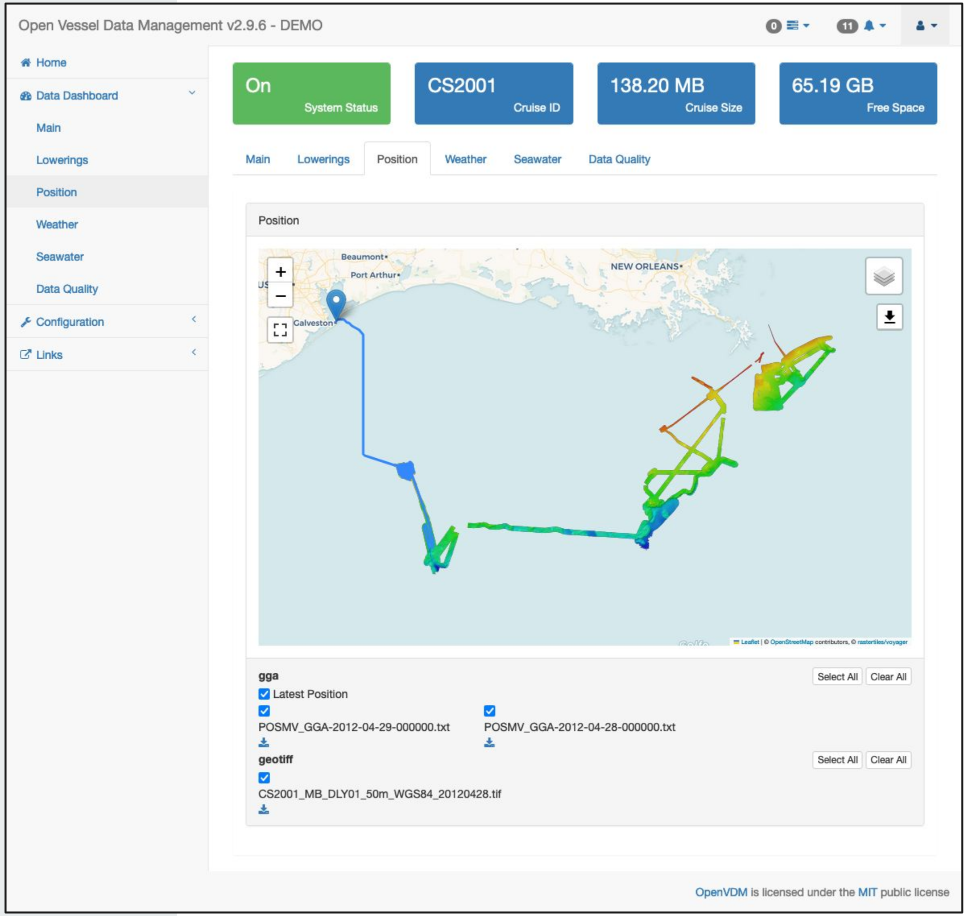
tsg

twind

The Web UI

Data Dashboard

Geographic Datasets via Leaflet


Open Vessel Data Management v2.9.6 - DEMO

On System Status | CS2001 Cruise ID | 138.20 MB Cruise Size | 65.19 GB Free Space

gga

Latest Position: ✓

✓ POSMV_GGA-2012-04-29-000000.txt

✓ POSMV_GGA-2012-04-28-000000.txt

geotiff

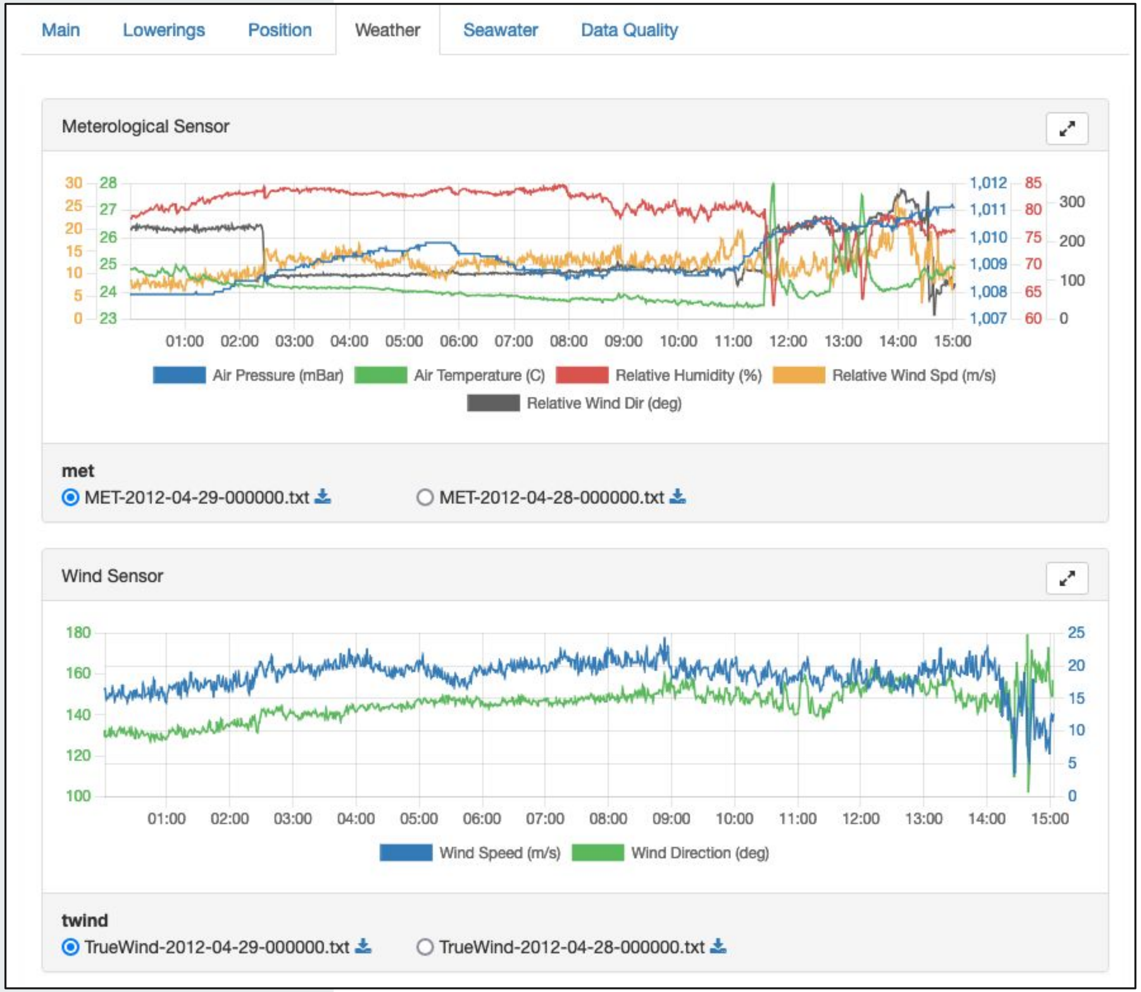
✓ CS2001_MB_DLY01_50m_WGS84_20120428.tif

OpenVDM is licensed under the MIT public license

The Web UI

Data Dashboard

Time series datasets via chart.js



met

[✓ MET-2012-04-29-000000.txt] MET-2012-04-28-000000.txt

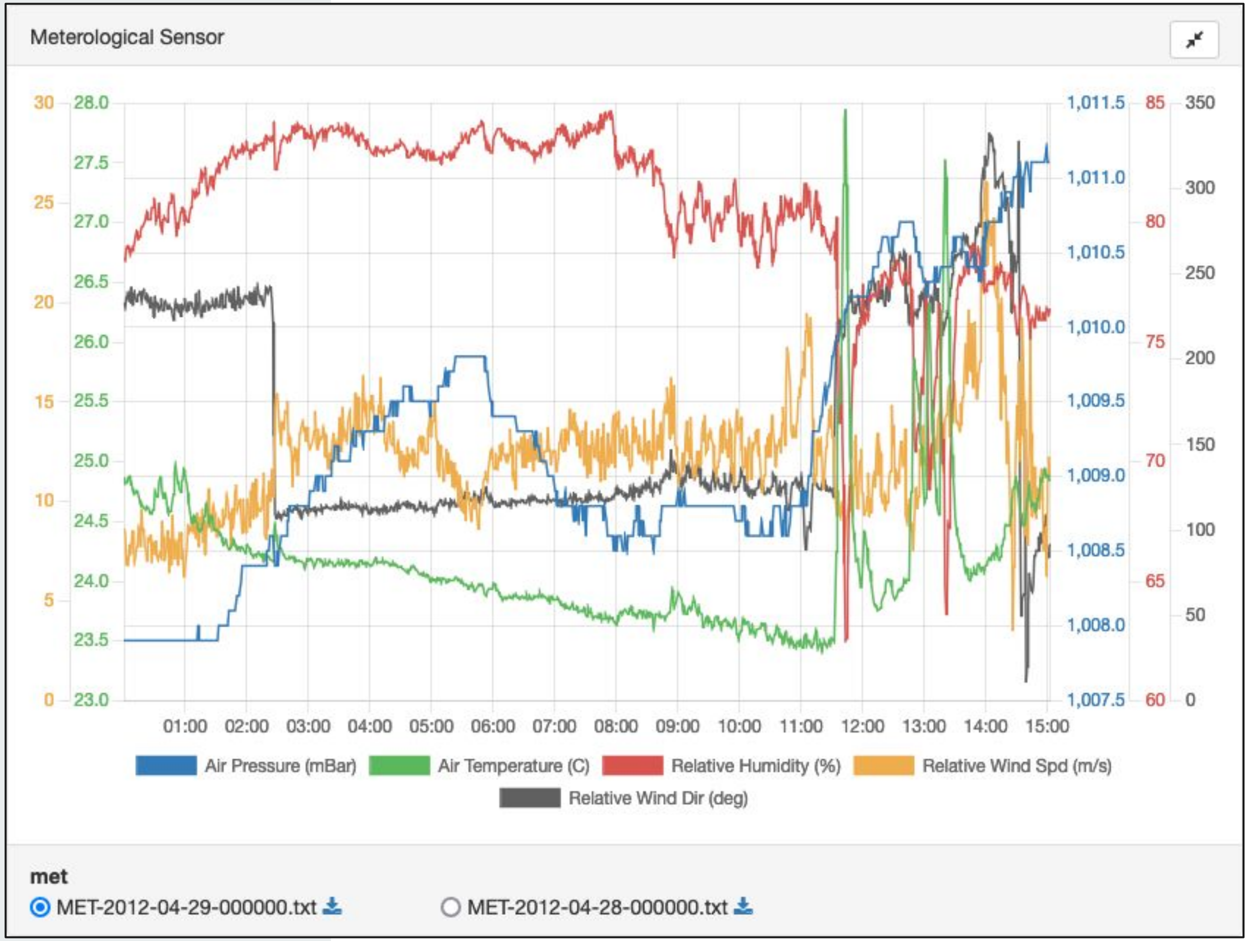
twind

[✓ TrueWind-2012-04-29-000000.txt] TrueWind-2012-04-28-000000.txt

The Web UI

Data Dashboard

Time series datasets via chart.js



Meterological Sensor

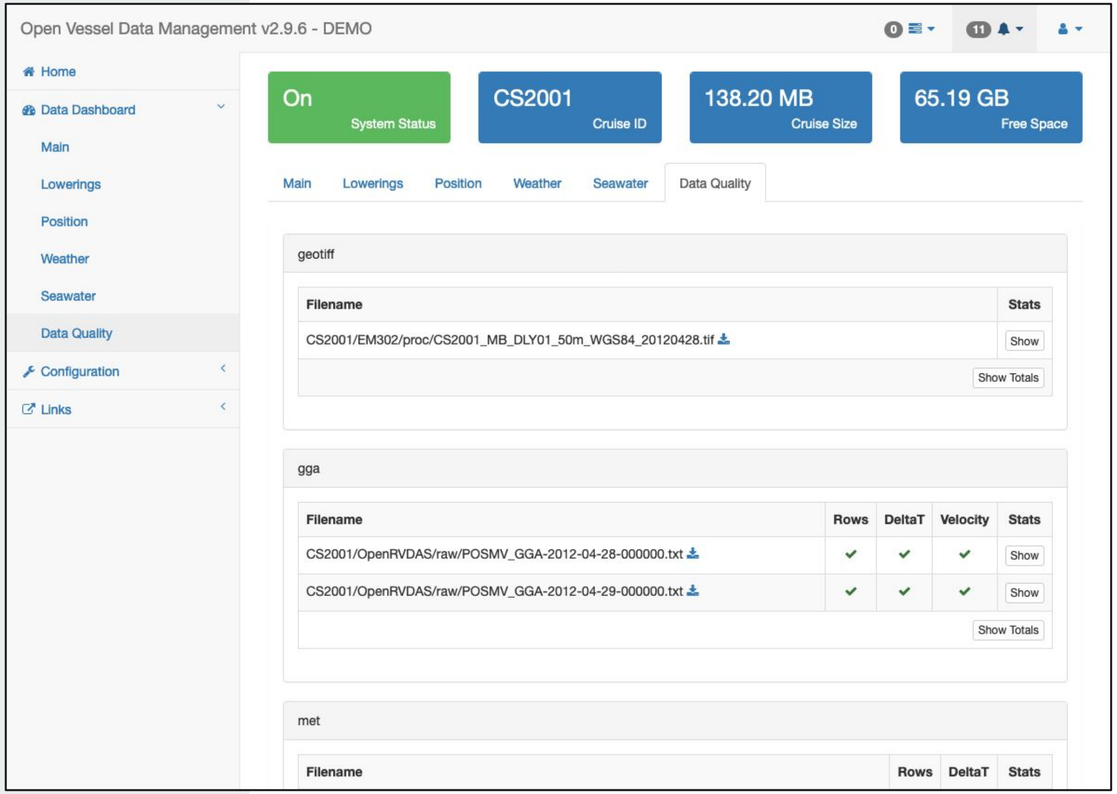
met

[✓ MET-2012-04-29-000000.txt] MET-2012-04-28-000000.txt

The Web UI

Data Dashboard

Data Quality


Open Vessel Data Management v2.9.6 - DEMO

On System Status | CS2001 Cruise ID | 138.20 MB Cruise Size | 65.19 GB Free Space

geotiff

Filename	Stats
CS2001/EM302/proc/CS2001_MB_DLY01_50m_WGS84_20120428.tif	Show
	Show Totals

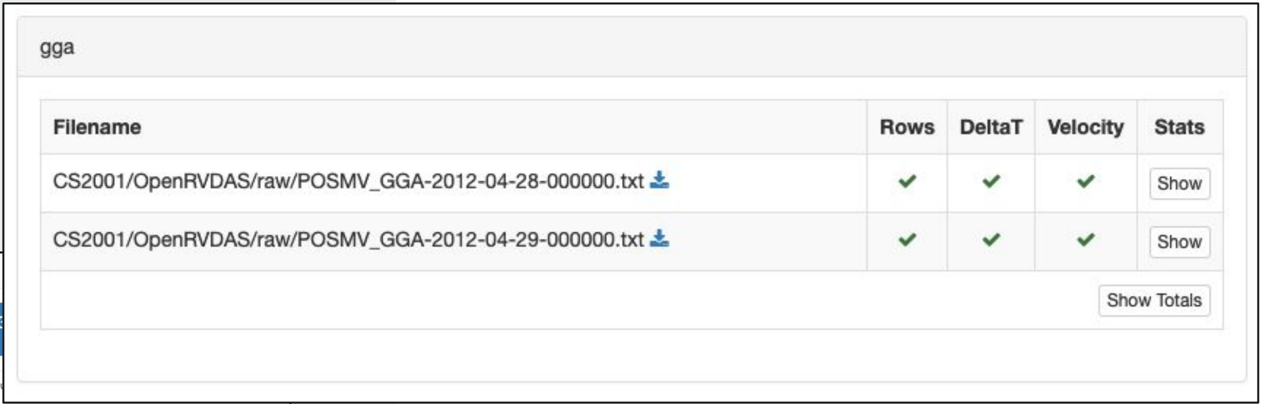
gga

Filename	Rows	DeltaT	Velocity	Stats
CS2001/OpenRVDAS/raw/POSMV_GGA-2012-04-28-000000.txt	✓	✓	✓	Show
CS2001/OpenRVDAS/raw/POSMV_GGA-2012-04-29-000000.txt	✓	✓	✓	Show
				Show Totals

met

Filename	Rows	DeltaT	Stats

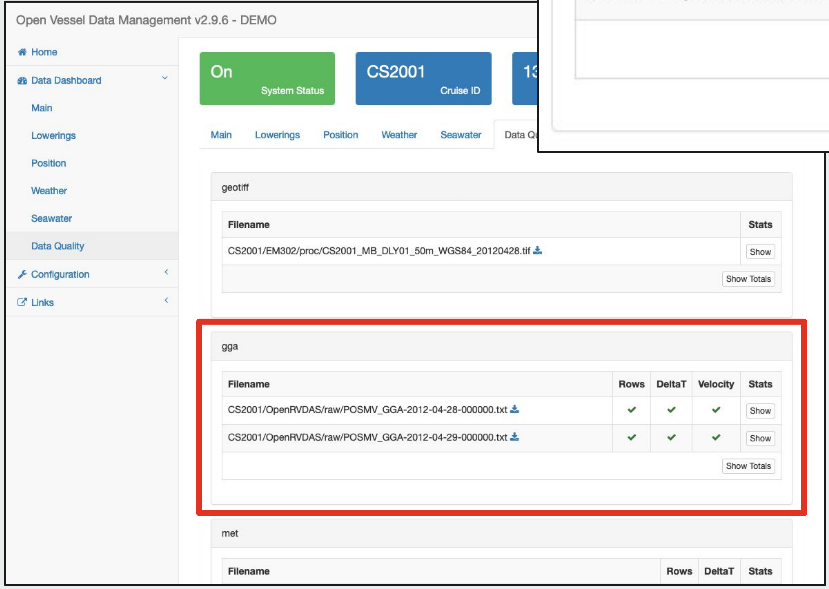
The Web UI



gga

Filename	Rows	DeltaT	Velocity	Stats
CS2001/OpenRVDAS/raw/POSMV_GGA-2012-04-28-000000.txt	✓	✓	✓	Show
CS2001/OpenRVDAS/raw/POSMV_GGA-2012-04-29-000000.txt	✓	✓	✓	Show

Show Totals



Open Vessel Data Management v2.9.6 - DEMO

geotiff

Filename	Stats
CS2001/EM302/proc/CS2001_MB_DLY01_50m_WGS84_20120428.tif	Show

Show Totals

gga

Filename	Rows	DeltaT	Velocity	Stats
CS2001/OpenRVDAS/raw/POSMV_GGA-2012-04-28-000000.txt	✓	✓	✓	Show
CS2001/OpenRVDAS/raw/POSMV_GGA-2012-04-29-000000.txt	✓	✓	✓	Show

Show Totals

met

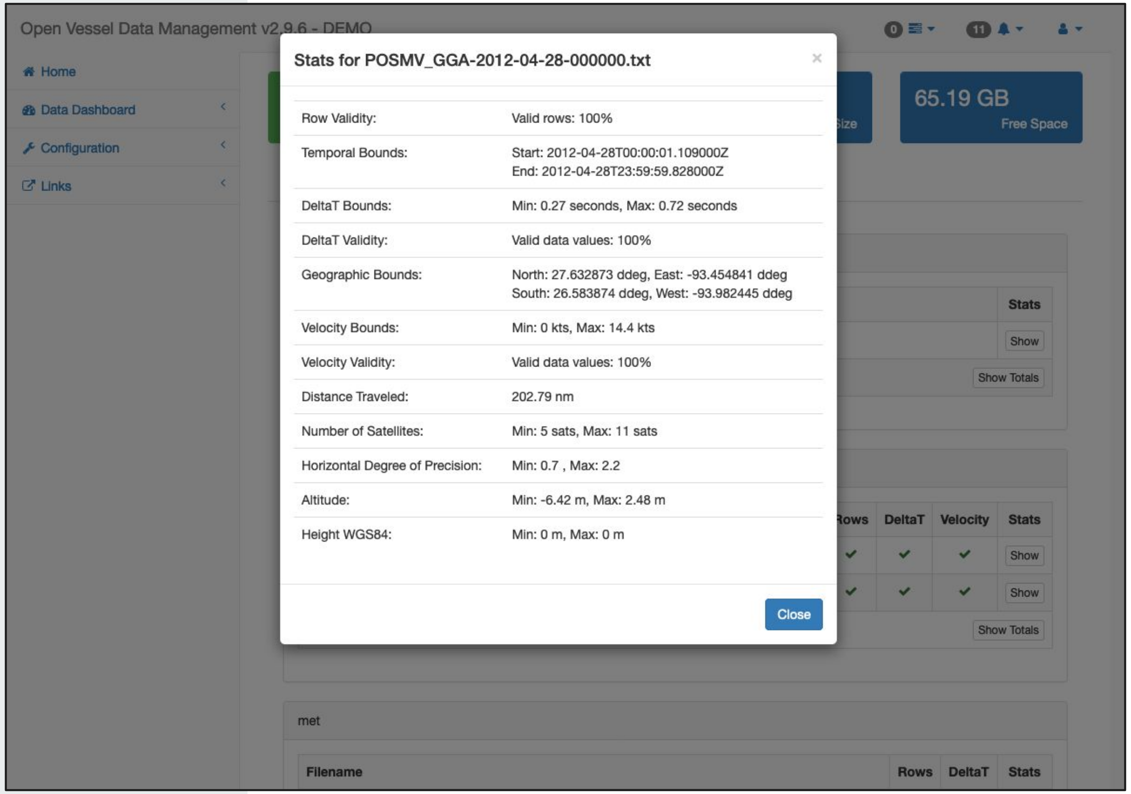
Filename	Rows	DeltaT	Stats

Data Dashboard

Show Filename and QA test results

The Web UI

Data Dashboard
Clicking "Show" displays details file stats as requested in the parser

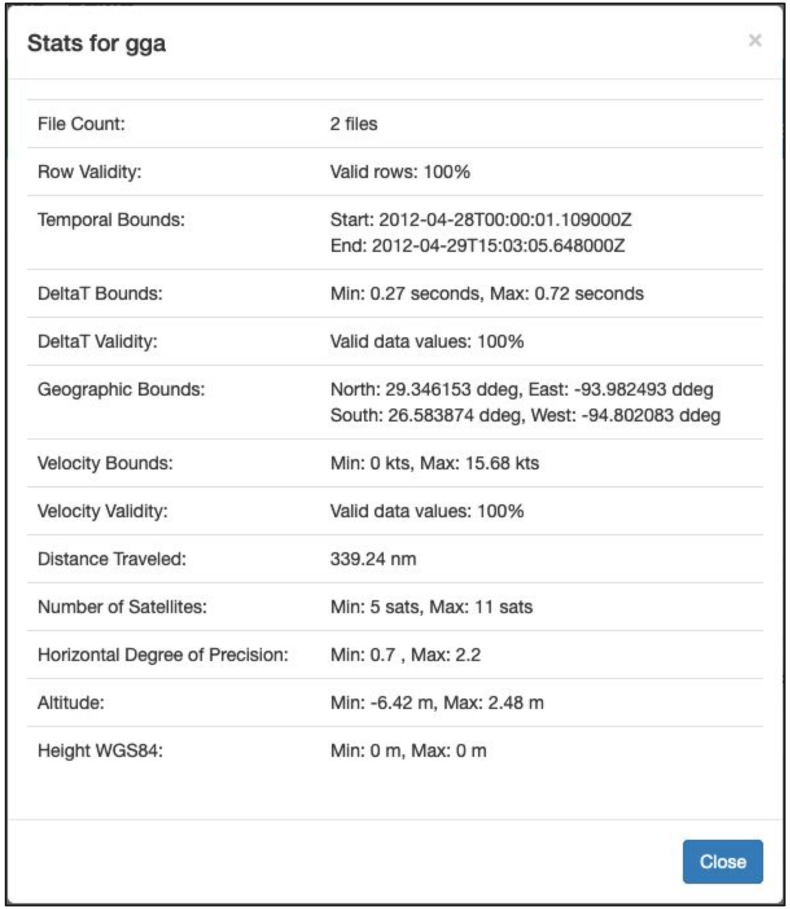


Stats for POSMV_GGA-2012-04-28-000000.txt

Row Validity:	Valid rows: 100%
Temporal Bounds:	Start: 2012-04-28T00:00:01.109000Z End: 2012-04-28T23:59:59.828000Z
DeltaT Bounds:	Min: 0.27 seconds, Max: 0.72 seconds
DeltaT Validity:	Valid data values: 100%
Geographic Bounds:	North: 27.632873 ddeg, East: -93.454841 ddeg South: 26.583874 ddeg, West: -93.982445 ddeg
Velocity Bounds:	Min: 0 kts, Max: 14.4 kts
Velocity Validity:	Valid data values: 100%
Distance Traveled:	202.79 nm
Number of Satellites:	Min: 5 sats, Max: 11 sats
Horizontal Degree of Precision:	Min: 0.7 , Max: 2.2
Altitude:	Min: -6.42 m, Max: 2.48 m
Height WGS84:	Min: 0 m, Max: 0 m

The Web UI

Data Dashboard
Clicking "Show Totals" displays the combines stats for all the files of that type

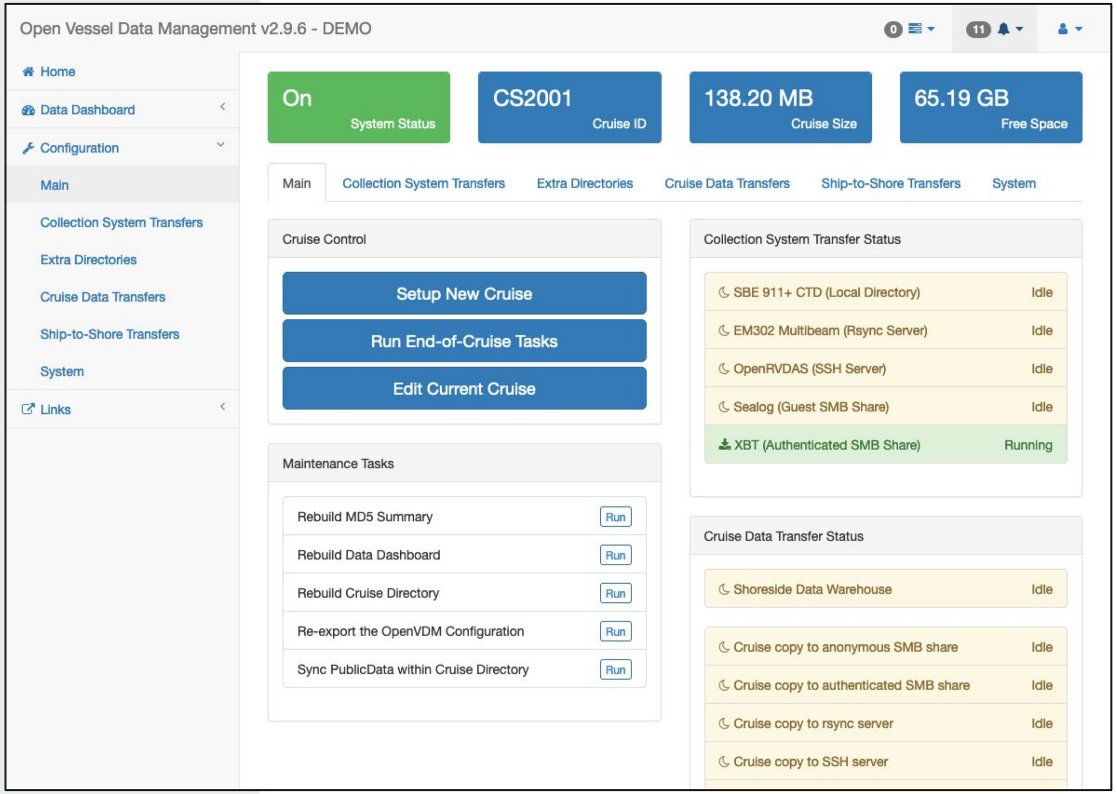


Stats for gga

File Count:	2 files
Row Validity:	Valid rows: 100%
Temporal Bounds:	Start: 2012-04-28T00:00:01.109000Z End: 2012-04-29T15:03:05.648000Z
DeltaT Bounds:	Min: 0.27 seconds, Max: 0.72 seconds
DeltaT Validity:	Valid data values: 100%
Geographic Bounds:	North: 29.346153 ddeg, East: -93.982493 ddeg South: 26.583874 ddeg, West: -94.802083 ddeg
Velocity Bounds:	Min: 0 kts, Max: 15.68 kts
Velocity Validity:	Valid data values: 100%
Distance Traveled:	339.24 nm
Number of Satellites:	Min: 5 sats, Max: 11 sats
Horizontal Degree of Precision:	Min: 0.7 , Max: 2.2
Altitude:	Min: -6.42 m, Max: 2.48 m
Height WGS84:	Min: 0 m, Max: 0 m

The Web UI

Configuration



Rebuild MD5 Summary	Run
Rebuild Data Dashboard	Run
Rebuild Cruise Directory	Run
Re-export the OpenVDM Configuration	Run
Sync PublicData within Cruise Directory	Run

Collection System Transfer Status

SBE 911+ CTD (Local Directory)	Idle
EM302 Multibeam (Rsync Server)	Idle
OpenRVDAS (SSH Server)	Idle
Sealog (Guest SMB Share)	Idle
XBT (Authenticated SMB Share)	Running

Cruise Data Transfer Status

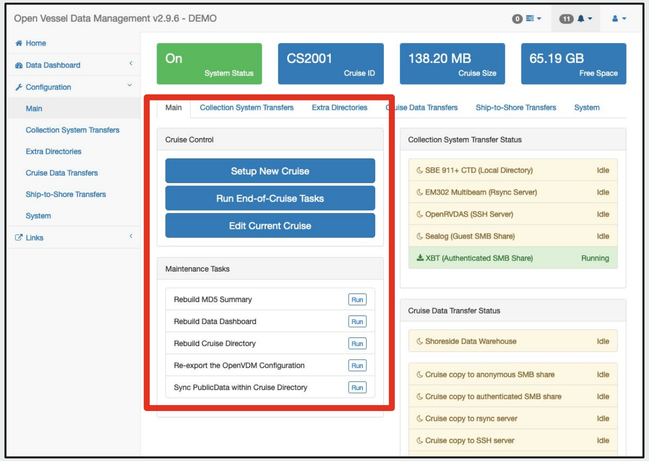
Shoreside Data Warehouse	Idle
Cruise copy to anonymous SMB share	Idle
Cruise copy to authenticated SMB share	Idle
Cruise copy to rsync server	Idle
Cruise copy to SSH server	Idle

The Web UI

Configuration

Cruise Control
Manage the current cruise

Maintenance Tasks
Perform non-transfer related tasks


Open Vessel Data Management v2.9.6 - DEMO

- Home
- Data Dashboard
- Configuration
  - Main
  - Collection System Transfers
  - Extra Directories
  - Cruise Data Transfers
  - Ship-to-Shore Transfers
  - System
- Links

On	CS2001	138.20 MB	65.19 GB
System Status	Cruise ID	Cruise Size	Free Space

Main | Collection System Transfers | Extra Directories | Cruise Data Transfers | Ship-to-Shore Transfers | System

Cruise Control

- Setup New Cruise
- Run End-of-Cruise Tasks
- Edit Current Cruise

Maintenance Tasks

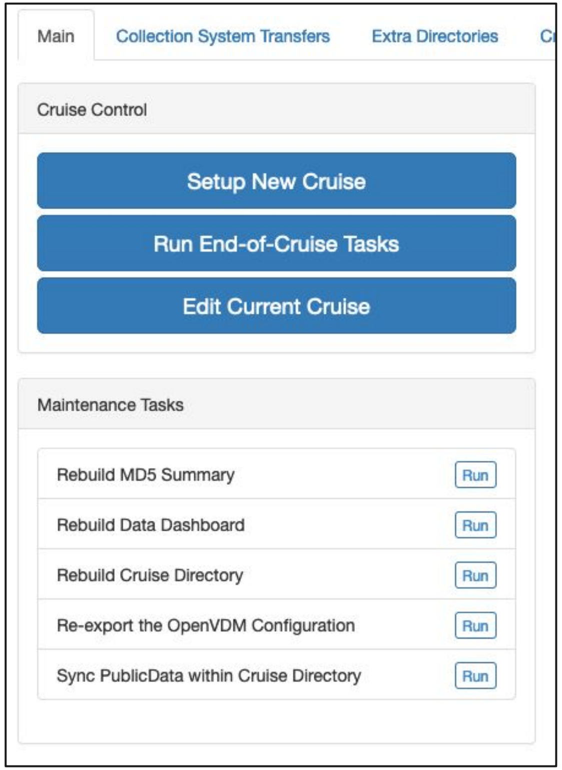
Rebuild MD5 Summary	Run
Rebuild Data Dashboard	Run
Rebuild Cruise Directory	Run
Re-export the OpenVDM Configuration	Run
Sync PublicData within Cruise Directory	Run

Collection System Transfer Status

SBE 911+ CTD (Local Directory)	Idle
EM302 Multibeam (Rsync Server)	Idle
OpenRVDAS (SSH Server)	Idle
Sealog (Guest SMB Share)	Idle
XBT (Authenticated SMB Share)	Running

Cruise Data Transfer Status

Shoreside Data Warehouse	Idle
Cruise copy to anonymous SMB share	Idle
Cruise copy to authenticated SMB share	Idle
Cruise copy to rsync server	Idle
Cruise copy to SSH server	Idle


Main | Collection System Transfers | Extra Directories | C

Cruise Control

- Setup New Cruise
- Run End-of-Cruise Tasks
- Edit Current Cruise

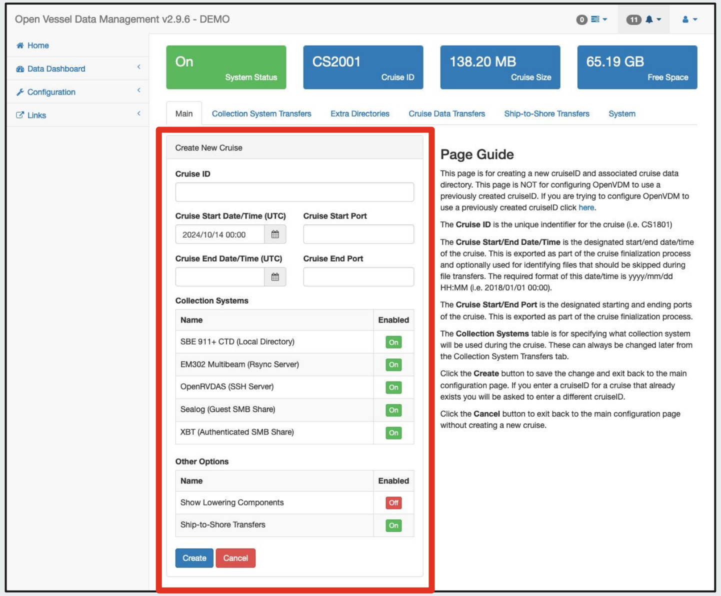
Maintenance Tasks

Rebuild MD5 Summary	Run
Rebuild Data Dashboard	Run
Rebuild Cruise Directory	Run
Re-export the OpenVDM Configuration	Run
Sync PublicData within Cruise Directory	Run

The Web UI

Configuration

Creating a new cruise



Open Vessel Data Management v2.9.6 - DEMO

On System Status	CS2001 Cruise ID	138.20 MB Cruise Size	65.19 GB Free Space

Create New Cruise

Cruise ID

Cruise Start Date/Time (UTC): 2024/10/14 00:00

Cruise Start Port

Cruise End Date/Time (UTC)

Cruise End Port

Collection Systems

Name	Enabled
SBE 911+ CTD (Local Directory)	On
EM302 Multibeam (Rsync Server)	On
OpenRVDAS (SSH Server)	On
Sealog (Guest SMB Share)	On
XBT (Authenticated SMB Share)	On

Other Options

Name	Enabled
Show Lowering Components	Off
Ship-to-Shore Transfers	On

Page Guide

This page is for creating a new cruiseID and associated cruise data directory. This page is NOT for configuring OpenVDM to use a previously created cruiseID. If you are trying to configure OpenVDM to use a previously created cruiseID click here.

The **Cruise ID** is the unique indentifier for the cruise (i.e. CS1801)

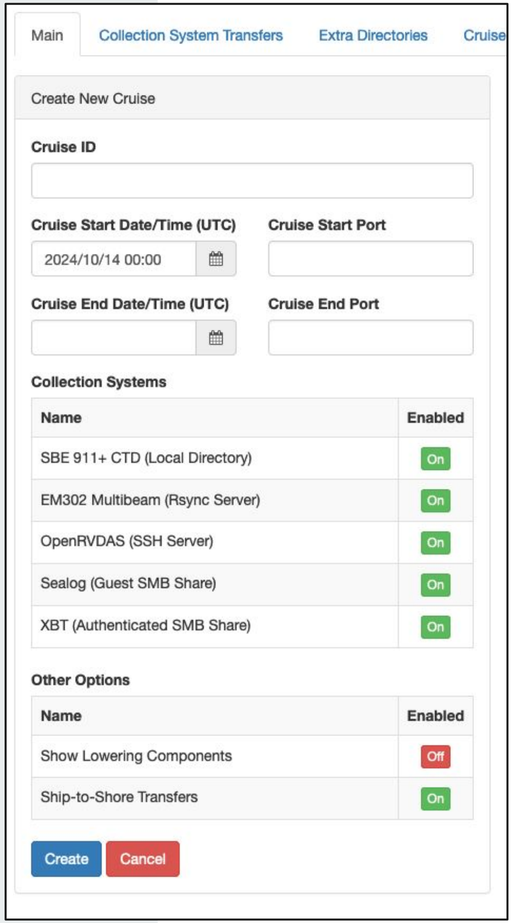
The **Cruise Start/End Date/Time** is the designated start/end date/time of the cruise. This is exported as part of the cruise finialization process and optionally used for identifying files that should be skipped during file transfers. The required format of this date/time is yyyy/mm/dd HH:MM (i.e. 2018/01/01 00:00).

The **Cruise Start/End Port** is the designated starting and ending ports of the cruise. This is exported as part of the cruise finialization process.

The **Collection Systems** table is for specifying what collection system will be used during the cruise. These can always be changed later from the Collection System Transfers tab.

Click the **Create** button to save the change and exit back to the main configuration page. If you enter a cruiseID for a cruise that already exists you will be asked to enter a different cruiseID.

Click the **Cancel** button to exit back to the main configuration page without creating a new cruise.



Create New Cruise

Cruise ID

Cruise Start Date/Time (UTC): 2024/10/14 00:00

Cruise Start Port

Cruise End Date/Time (UTC)

Cruise End Port

Collection Systems

Name	Enabled
SBE 911+ CTD (Local Directory)	On
EM302 Multibeam (Rsync Server)	On
OpenRVDAS (SSH Server)	On
Sealog (Guest SMB Share)	On
XBT (Authenticated SMB Share)	On

Other Options

Name	Enabled
Show Lowering Components	Off
Ship-to-Shore Transfers	On

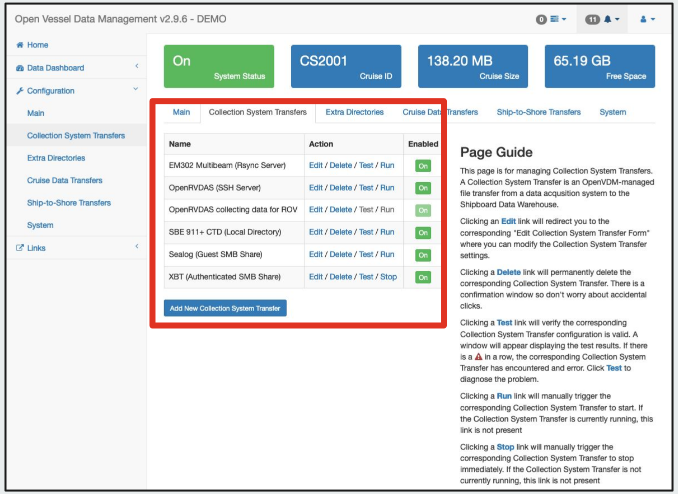
Steps:

1. Define the Cruise ID
2. Define Start/Stop dates and ports
3. Enable pertinent transfers
4. Click “Create”

The Web UI

Configuration

Collection System Transfers



Name	Action	Enabled
EM302 Multibeam (Rsync Server)	Edit / Delete / Test / Run	On
OpenRVDAS (SSH Server)	Edit / Delete / Test / Run	On
OpenRVDAS collecting data for ROV	Edit / Delete / Test / Run	On
SBE 911+ CTD (Local Directory)	Edit / Delete / Test / Run	On
Sealog (Guest SMB Share)	Edit / Delete / Test / Run	On
XBT (Authenticated SMB Share)	Edit / Delete / Test / Stop	On

Add New Collection System Transfer

Page Guide

This page is for managing Collection System Transfers. A Collection System Transfer is an OpenVDM-managed file transfer from a data acqusition system to the Shipboard Data Warehouse.

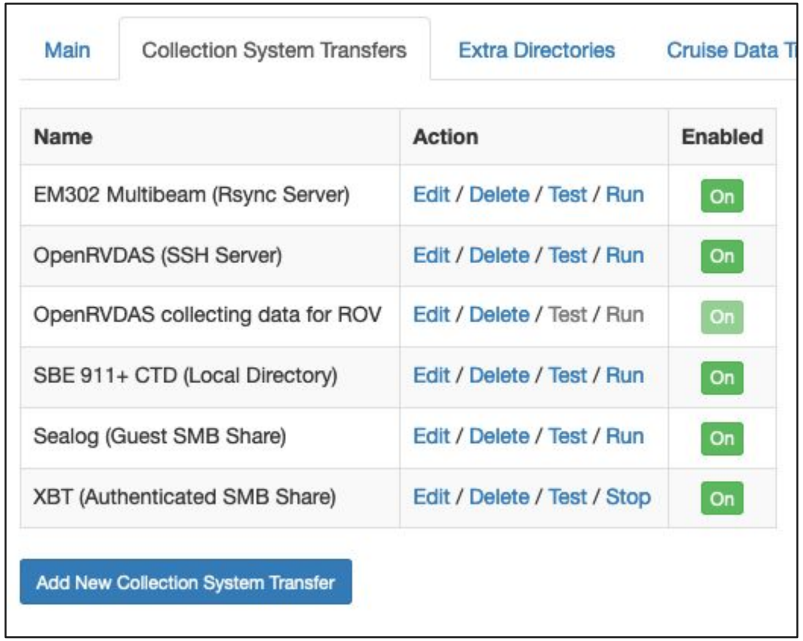
Clicking an **Edit** link will redirect you to the corresponding "Edit Collection System Transfer Form" where you can modify the Collection System Transfer settings.

Clicking a **Delete** link will permanently delete the corresponding Collection System Transfer. There is a confirmation window so don't worry about accidental clicks.

Clicking a **Test** link will verify the corresponding Collection System Transfer configuration is valid. A window will appear displaying the test results. If there is a ⚠ in a row, the corresponding Collection System Transfer has encountered and error. Click **Test** to diagnose the problem.

Clicking a **Run** link will manually trigger the corresponding Collection System Transfer to start. If the Collection System Transfer is currently running, this link is not present

Clicking a **Stop** link will manually trigger the corresponding Collection System Transfer to stop immediately. If the Collection System Transfer is not currently running, this link is not present

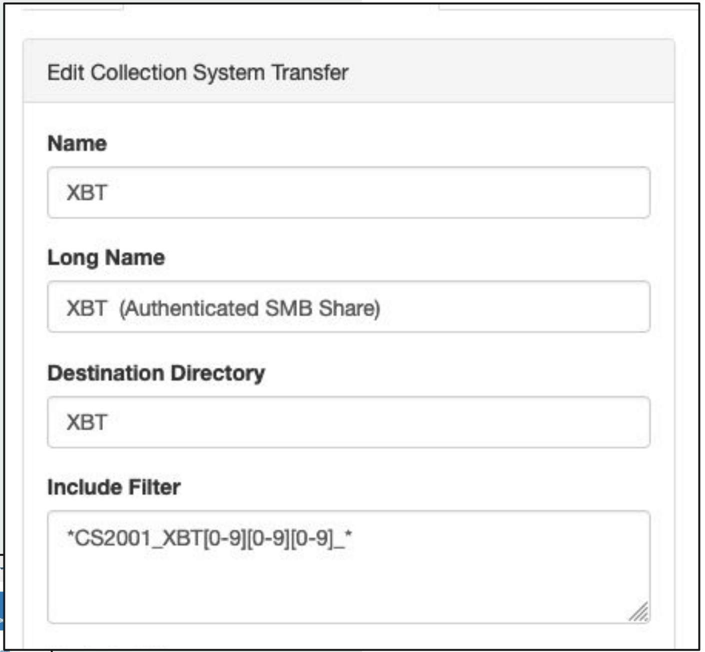


Name	Action	Enabled
EM302 Multibeam (Rsync Server)	Edit / Delete / Test / Run	On
OpenRVDAS (SSH Server)	Edit / Delete / Test / Run	On
OpenRVDAS collecting data for ROV	Edit / Delete / Test / Run	On
SBE 911+ CTD (Local Directory)	Edit / Delete / Test / Run	On
Sealog (Guest SMB Share)	Edit / Delete / Test / Run	On
XBT (Authenticated SMB Share)	Edit / Delete / Test / Stop	On

Add New Collection System Transfer

The Web UI

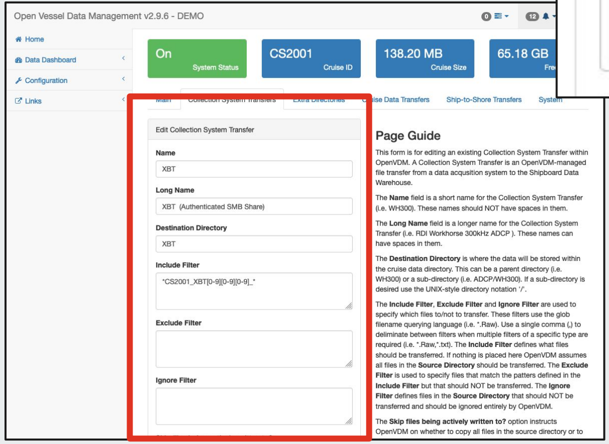
Configuration
Collection System Transfers



Edit Collection System Transfer

Name	XBT
Long Name	XBT (Authenticated SMB Share)
Destination Directory	XBT
Include Filter	*CS2001_XBT[0-9][0-9][0-9]_*

- Name
- Long Name
- Destination Directory
- Include filter



Open Vessel Data Management v2.9.6 - DEMO

On System Status | CS2001 Cruise ID | 138.20 MB Cruise Size | 65.18 GB Free

Edit Collection System Transfer

Name	XBT
Long Name	XBT (Authenticated SMB Share)
Destination Directory	XBT
Include Filter	*CS2001_XBT[0-9][0-9][0-9]_*
Exclude Filter	
Ignore Filter	

Page Guide

This form is for editing an existing Collection System Transfer within OpenVDM. A Collection System Transfer is an OpenVDM-managed file transfer from a data acqusition system to the Shipboard Data Warehouse.

The **Name** field is a short name for the Collection System Transfer (i.e. WH300). These names should NOT have spaces in them.

The **Long Name** field is a longer name for the Collection System Transfer (i.e. RDI Workhorse 300kHz ADCP). These names can have spaces in them.

The **Destination Directory** is where the data will be stored within the cruise data directory. This can be a parent directory (i.e. WH300) or a sub-directory (i.e. ADCP/WH300). If a sub-directory is desired use the UNIX-style directory notation '/'.

The **Include Filter**, **Exclude Filter** and **Ignore Filter** are used to specify which files to/not to transfer. These filters use the glob filename querying language (i.e. *.Raw). Use a single comma (,) to deliminate between filters when multiple filters of a specific type are required (i.e. *.Raw,*.txt). The **Include Filter** defines what files should be transferred. If nothing is placed here OpenVDM assumes all files in the **Source Directory** should be transferred. The **Exclude Filter** is used to specify files that match the patters defined in the **Include Filter** but that should NOT be transferred. The **Ignore Filter** defines files in the **Source Directory** that should NOT be transferred and should be ignored entirely by OpenVDM.

The **Skip files being actively written to?** option instructs OpenVDM on whether to copy all files in the source directory or to

Filter uses glob syntax

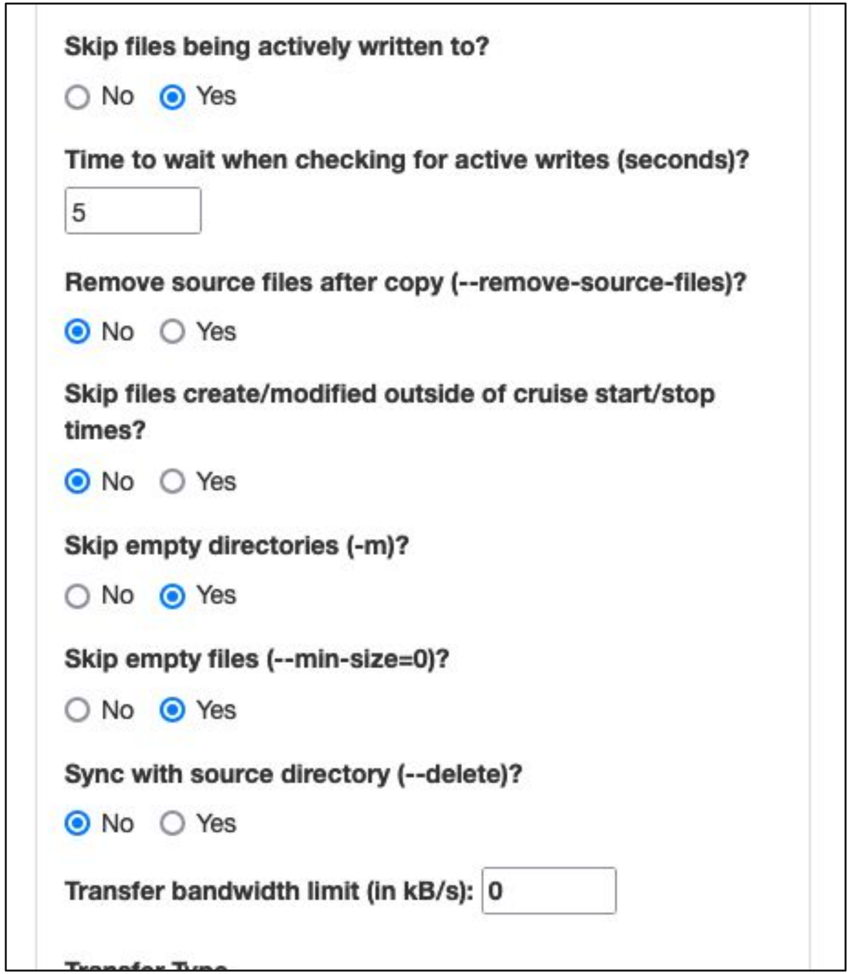
- Supports options (`txt|csv`)
- Supports ranges (`[0-9]`)
- Supports a cruiseID wildcard (`{cruiseID}`)

The Web UI

Configuration
Collection System
Transfers

Filter files by:
- If completed
- Modification times

...and many of the common rsync options



Skip files being actively written to?
No [✓ Yes]

Time to wait when checking for active writes (seconds)?
5

Remove source files after copy (--remove-source-files)?
[✓ No] Yes

Skip files create/modified outside of cruise start/stop times?
[✓ No] Yes

Skip empty directories (-m)?
No [✓ Yes]

Skip empty files (--min-size=0)?
No [✓ Yes]

Sync with source directory (--delete)?
[✓ No] Yes

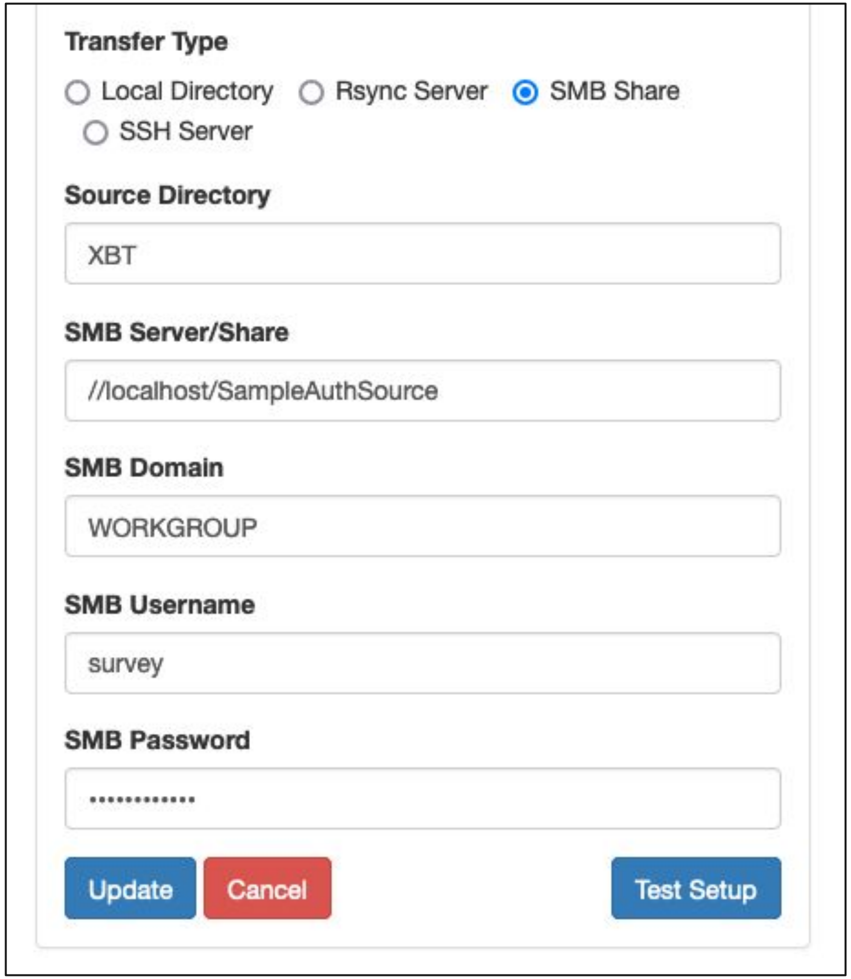
Transfer bandwidth limit (in kB/s): 0

The Web UI

Configuration
Collection System
Transfers

- Multiple options for connecting to remote systems.

- No client-side software to install.



Transfer Type

Local Directory Rsync Server [✓ SMB Share] SSH Server

Source Directory

XBT

SMB Server/Share

//localhost/SampleAuthSource

SMB Domain

WORKGROUP

SMB Username

survey

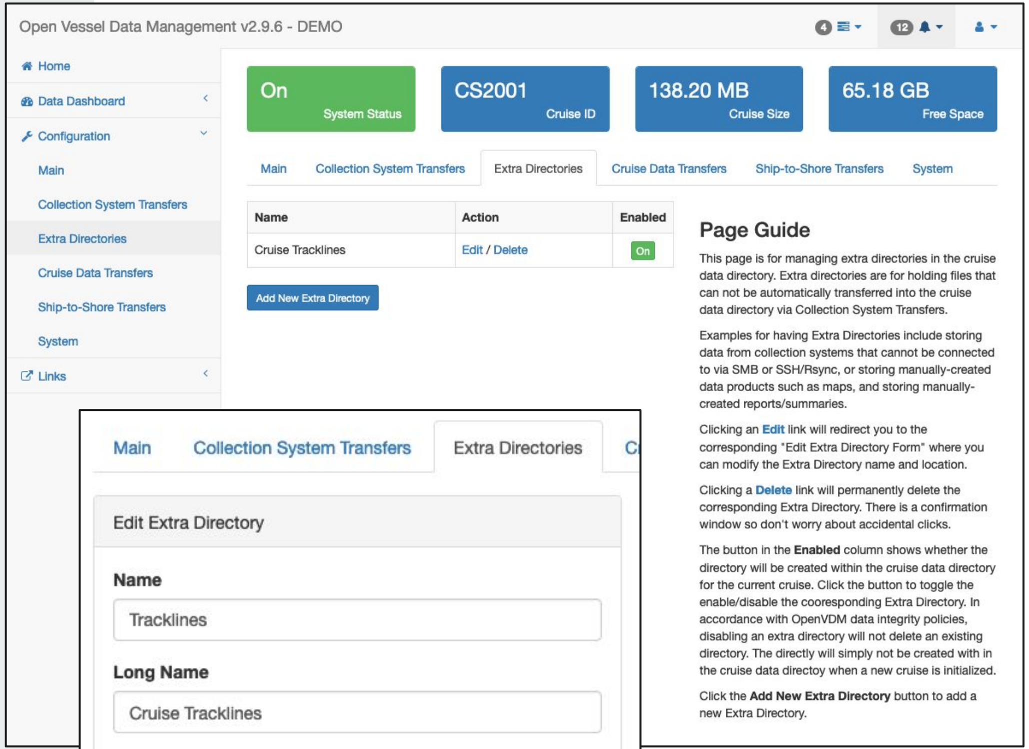
SMB Password

•••••••••••••

The Web UI

Configuration
Extra Directories

Directories creating within the cruise data directory that are ***not*** associated with collection system transfers



Open Vessel Data Management v2.9.6 - DEMO

On	CS2001	138.20 MB	65.18 GB
System Status	Cruise ID	Cruise Size	Free Space

Name	Action	Enabled
Cruise Tracklines	Edit / Delete	On

Page Guide

This page is for managing extra directories in the cruise data directory. Extra directories are for holding files that can not be automatically transferred into the cruise data directory via Collection System Transfers.

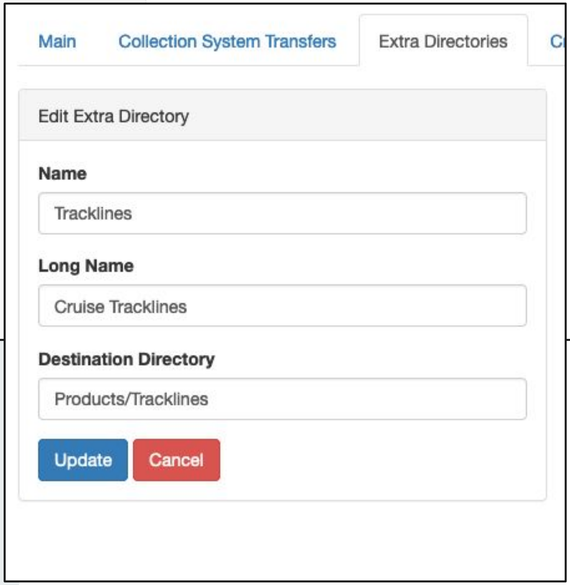
Examples for having Extra Directories include storing data from collection systems that cannot be connected to via SMB or SSH/Rsync, or storing manually-created data products such as maps, and storing manually-created reports/summaries.

Clicking an **Edit** link will redirect you to the corresponding "Edit Extra Directory Form" where you can modify the Extra Directory name and location.

Clicking a **Delete** link will permanently delete the corresponding Extra Directory. There is a confirmation window so don't worry about accidental clicks.

The button in the **Enabled** column shows whether the directory will be created within the cruise data directory for the current cruise. Click the button to toggle the enable/disable the cooresponding Extra Directory. In accordance with OpenVDM data integrity policies, disabling an extra directory will not delete an existing directory. The directly will simply not be created with in the cruise data directoy when a new cruise is initialized.

Click the **Add New Extra Directory** button to add a new Extra Directory.



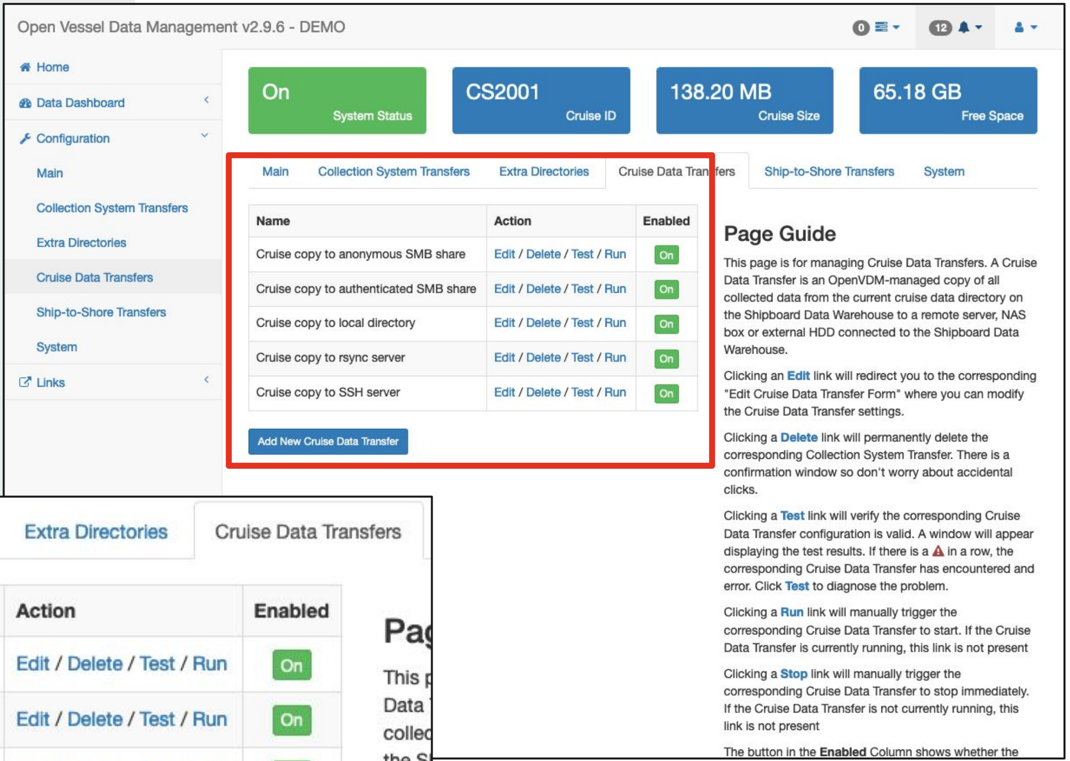
Edit Extra Directory

Name	Tracklines
Long Name	Cruise Tracklines
Destination Directory	Products/Tracklines

The Web UI

Configuration

Cruise Data Transfers



Open Vessel Data Management v2.9.6 - DEMO

- Home
- Data Dashboard
- Configuration
  - Main
  - Collection System Transfers
  - Extra Directories
  - Cruise Data Transfers
  - Ship-to-Shore Transfers
  - System
- Links

On — System Status | CS2001 — Cruise ID | 138.20 MB — Cruise Size | 65.18 GB — Free Space

Main | Collection System Transfers | Extra Directories | Cruise Data Transfers | Ship-to-Shore Transfers | System

Name	Action	Enabled
Cruise copy to anonymous SMB share	Edit / Delete / Test / Run	On
Cruise copy to authenticated SMB share	Edit / Delete / Test / Run	On
Cruise copy to local directory	Edit / Delete / Test / Run	On
Cruise copy to rsync server	Edit / Delete / Test / Run	On
Cruise copy to SSH server	Edit / Delete / Test / Run	On

Add New Cruise Data Transfer

Page Guide

This page is for managing Cruise Data Transfers. A Cruise Data Transfer is an OpenVDM-managed copy of all collected data from the current cruise data directory on the Shipboard Data Warehouse to a remote server, NAS box or external HDD connected to the Shipboard Data Warehouse.

Clicking an **Edit** link will redirect you to the corresponding "Edit Cruise Data Transfer Form" where you can modify the Cruise Data Transfer settings.

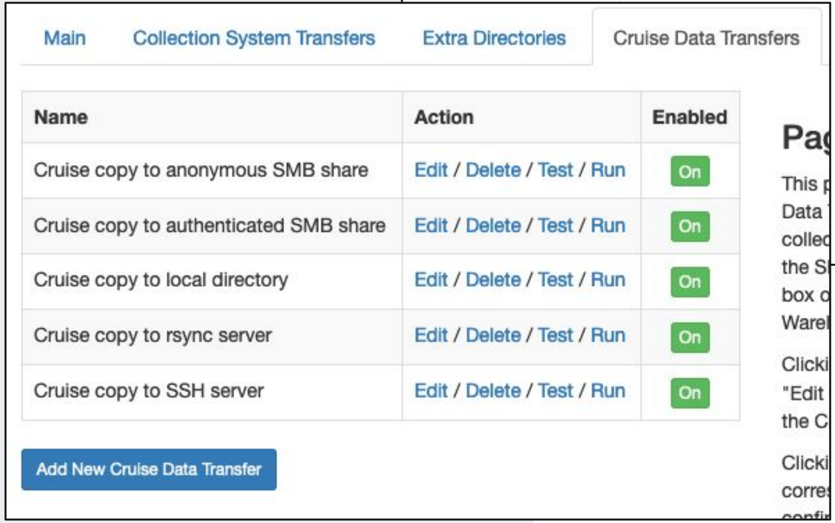
Clicking a **Delete** link will permanently delete the corresponding Collection System Transfer. There is a confirmation window so don't worry about accidental clicks.

Clicking a **Test** link will verify the corresponding Cruise Data Transfer configuration is valid. A window will appear displaying the test results. If there is a ⚠ in a row, the corresponding Cruise Data Transfer has encountered and error. Click **Test** to diagnose the problem.

Clicking a **Run** link will manually trigger the corresponding Cruise Data Transfer to start. If the Cruise Data Transfer is currently running, this link is not present

Clicking a **Stop** link will manually trigger the corresponding Cruise Data Transfer to stop immediately. If the Cruise Data Transfer is not currently running, this link is not present

The button in the **Enabled** Column shows whether the



Main | Collection System Transfers | Extra Directories | Cruise Data Transfers

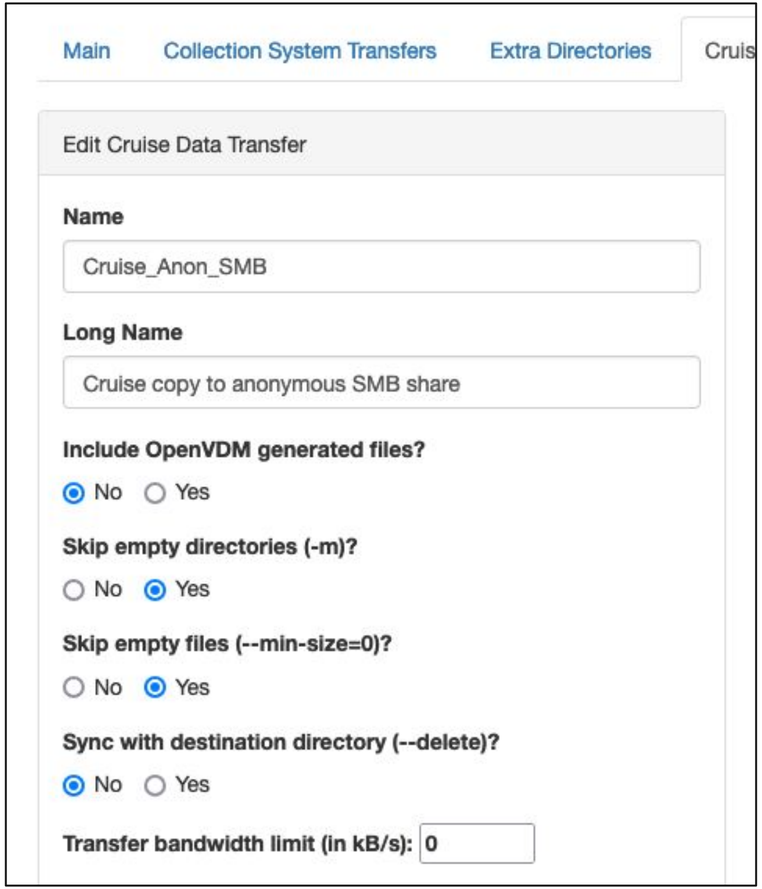
Name	Action	Enabled
Cruise copy to anonymous SMB share	Edit / Delete / Test / Run	On
Cruise copy to authenticated SMB share	Edit / Delete / Test / Run	On
Cruise copy to local directory	Edit / Delete / Test / Run	On
Cruise copy to rsync server	Edit / Delete / Test / Run	On
Cruise copy to SSH server	Edit / Delete / Test / Run	On

Add New Cruise Data Transfer

The Web UI

Configuration
Cruise Data Transfers

- Name
- Long Name
- Rsync options



Edit Cruise Data Transfer

Name

Cruise_Anon_SMB

Long Name

Cruise copy to anonymous SMB share

Include OpenVDM generated files?

[✓ No] Yes

Skip empty directories (-m)?

No [✓ Yes]

Skip empty files (--min-size=0)?

No [✓ Yes]

Sync with destination directory (--delete)?

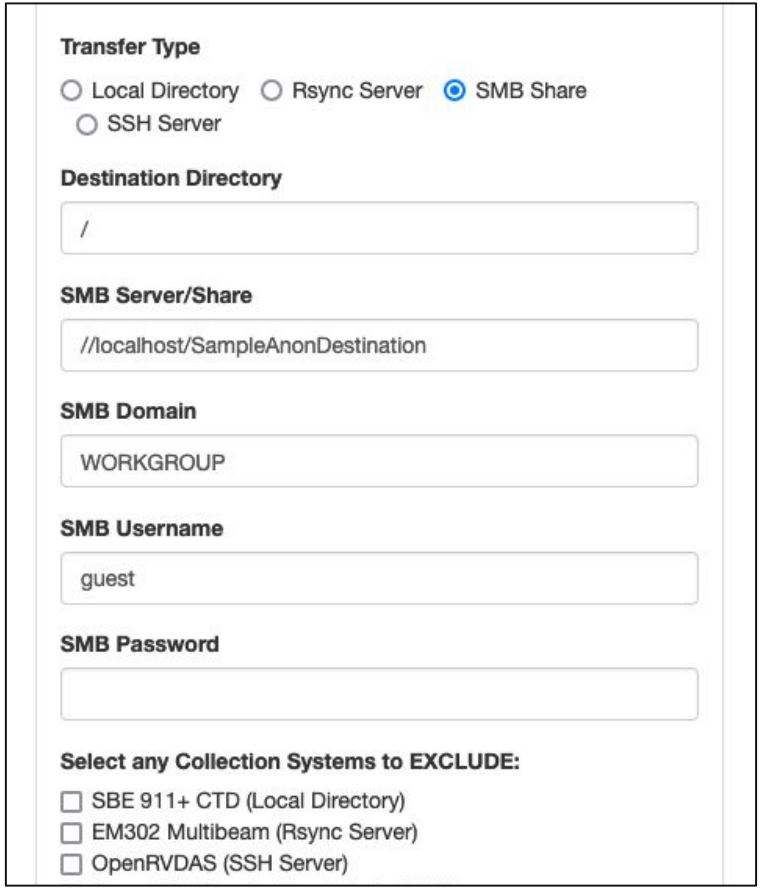
[✓ No] Yes

Transfer bandwidth limit (in kB/s): 0

The Web UI

Configuration
Cruise Data Transfers

- Multiple options for connecting to remote systems.
- No client-side software to install.



Transfer Type

( ) Local Directory ( ) Rsync Server [✓ SMB Share] ( ) SSH Server

Destination Directory

/

SMB Server/Share

//localhost/SampleAnonDestination

SMB Domain

WORKGROUP

SMB Username

guest

SMB Password

Select any Collection Systems to EXCLUDE:

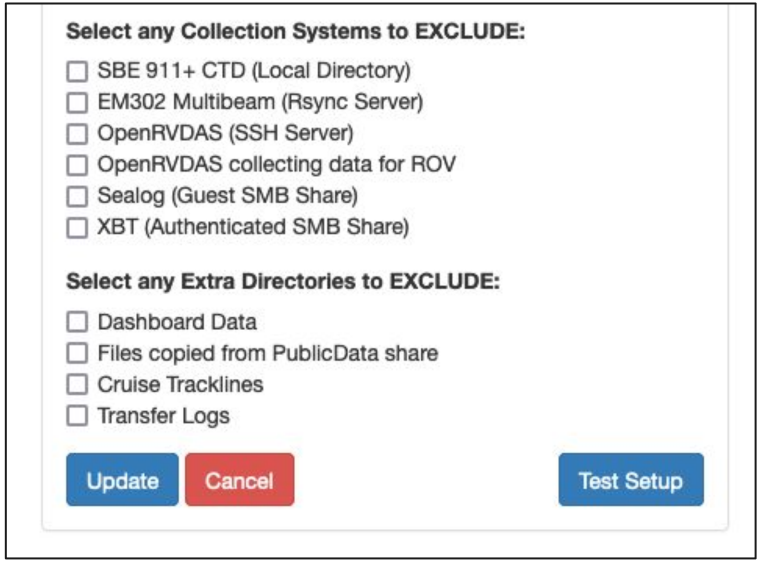
- ☐ SBE 911+ CTD (Local Directory)
- ☐ EM302 Multibeam (Rsync Server)
- ☐ OpenRVDAS (SSH Server)

The Web UI

Configuration

Cruise Data Transfers

- Copy all or a subset of the cruise data directory to destination



Select any Collection Systems to EXCLUDE:

- ☐ SBE 911+ CTD (Local Directory)
- ☐ EM302 Multibeam (Rsync Server)
- ☐ OpenRVDAS (SSH Server)
- ☐ OpenRVDAS collecting data for ROV
- ☐ Sealog (Guest SMB Share)
- ☐ XBT (Authenticated SMB Share)

Select any Extra Directories to EXCLUDE:

- ☐ Dashboard Data
- ☐ Files copied from PublicData share
- ☐ Cruise Tracklines
- ☐ Transfer Logs

OpenVDM

- Introduction - what/why/where
- Lingo 101
- Whole system overview - installation, Web-UI tour, defining/controlling transfers
- **Hooks - attaching processes to key points in processes**
- Displaying data - plugins and parsers
- Leveraging OpenVDM data elsewhere
- Best practices
- Contributing
- Where to from here?

If you want to...

- ...run jobs at the beginning of a cruise
- ...run jobs on files as they appear in the cruise data directory
- ...run jobs at the end of a cruise

Then you hooks are for you.

Hooks

```
# The hooks section contains any additional Gearman tasks that should be
# performed after the successful completion of the primary OpenVDM
# Gearman task.  Any subsequent tasks called with be called as background
# Gearman tasks so to not interfere with OpenVDM's primary operation.
hooks:
      runCollectionSystemTransfer:
      - updateDataDashboard
      - updateMD5Summary
      - postCollectionSystemTransfer
      updateDataDashboard:
      - updateMD5Summary
      - postDataDashboard
      rebuildDataDashboard:
      - updateMD5Summary
      - postDataDashboard
      setupNewCruise:
      - postSetupNewCruise
      setupNewLowering:
      - postSetupNewLowering
      finalizeCurrentCruise:
      - postFinalizeCurrentCruise
      finalizeCurrentLowering:
      - postFinalizeCurrentLowering
```

Hooks

```
# The postHookCommands section contains any additional commands that should be
# performed after the successful completion of the primary OpenVDM Gearman task.
# Any subsequent tasks called with be called as background Gearman tasks so to
# not interfere with OpenVDM's primary operation.
postHookCommands:
      postCollectionSystemTransfer:
      - collectionSystemTransferName: OpenRVDAS
        commandList:
        - name: "R2R NavManager"
          command:
            - /opt/openvdm/venv/bin/python
            - /opt/openvdm/bin/r2r_nav_manager.py
            - OpenRVDAS
      postSetupNewCruise:
        commandList:
        - name: "Build new OpenRVDAS config file"
          command:
      - ssh
      - sio-sts@sp-openrvdas.ucsd.edu
      - "bash /opt/SIO-SP/openrvdas/bin/build_openrvdas_config.sh"
      postFinalizeCurrentCruise:
        commandList:
        - name: "Export Sealog Vessel Data"
          command:
            - /opt/sealog-server-vessel/venv/bin/python
            - /opt/sealog-server-vessel/misc/sealog_vessel_data_export.py
```

Hooks

/opt/openvdm/server/etc/openvdm.yaml

```
postHookCommands:
      postCollectionSystemTransfer:
      - collectionSystemTransferName: SomeTransfer
        commandList:
        - name: "Some process"
          command:
            - bash
            - /home/mtech/process_file.sh
            - {newFiles}
```

/home/mtech/process_file.sh

```
process_file() {
  INPUT_FILE=$1

  echo "Current input file: ${INPUT_FILE}"
  if [ -f "${INPUT_FILE}" ]; then
      echo "Processing file: ${INPUT_FILE}"

      #################################
      #    PUT PROCESSING CODE HERE   #
      #################################

  else
      echo "WARNING: file does not exist... skipping."
  fi
}

main() {

  for data_file in "${DATA_FILES[@]}"; do
      process_file "${data_file}"
  done
}

# -----------------------------------------------------
main "$@"
```

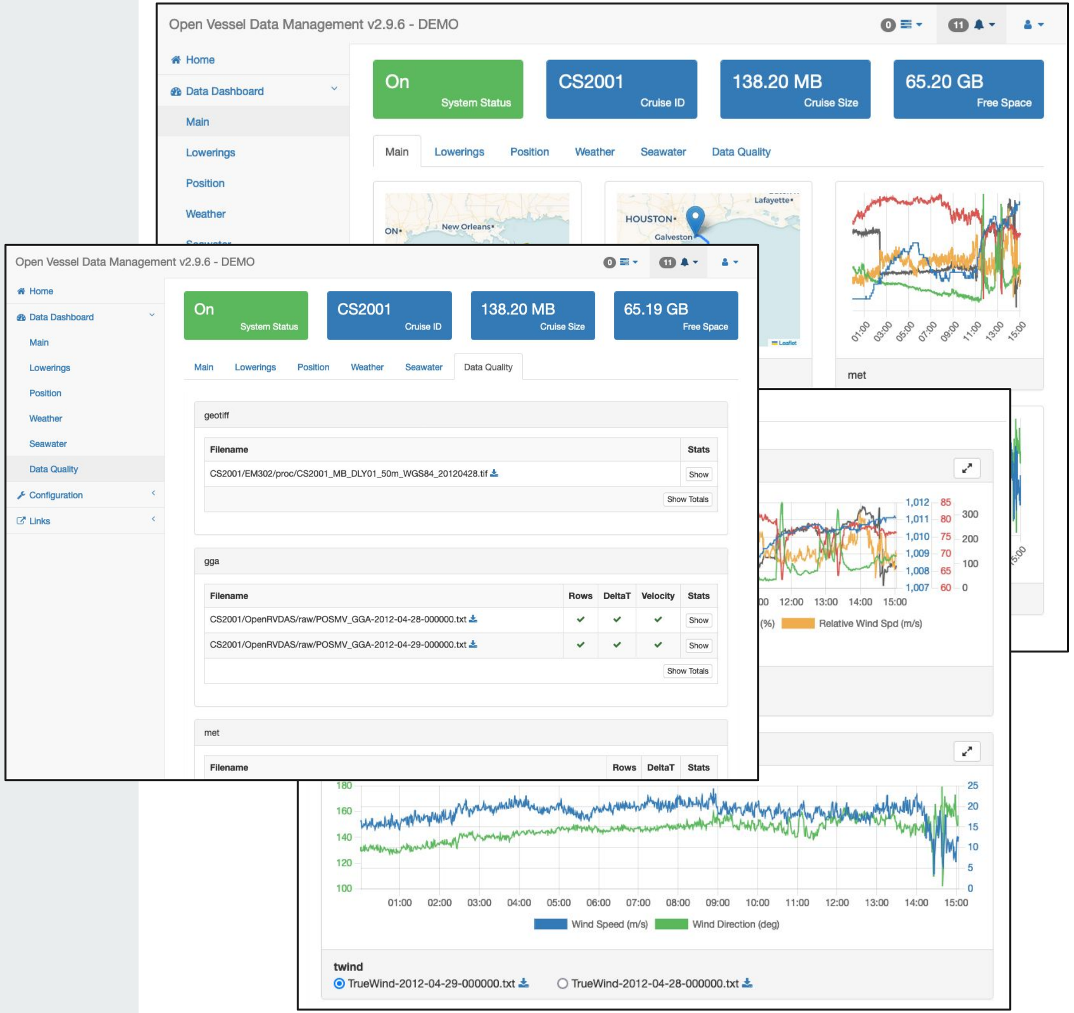
OpenVDM

- Introduction - what/why/where
- Lingo 101
- Whole system overview - installation, Web-UI tour, defining/controlling transfers
- Hooks - attaching processes to key points in processes
- **Displaying data** - plugins and parsers
- Leveraging OpenVDM data elsewhere
- Best practices
- Contributing
- Where to from here?

Plugins & Parsers

Plugins handle collection systems

Parsers handle file types



geotiff

Filename	Stats
CS2001/EM302/proc/CS2001_MB_DLY01_50m_WGS84_20120428.tif	Show
	Show Totals

gga

Filename	Rows	DeltaT	Velocity	Stats
CS2001/OpenRVDAS/raw/POSMV_GGA-2012-04-28-000000.txt	✓	✓	✓	Show
CS2001/OpenRVDAS/raw/POSMV_GGA-2012-04-29-000000.txt	✓	✓	✓	Show
				Show Totals

met

Filename	Rows	DeltaT	Stats

twind

● TrueWind-2012-04-29-000000.txt ○ TrueWind-2012-04-28-000000.txt

Plugins & Parsers

Plugins location: ./server/plugins
Naming convention: <cst_name>_plugin.py

Parser location: ./server/plugins/parser
Naming convention: <filetype>_parser.py>

Plugins

Plugins are called by the runCollectionSystemTransfer hook.

When called, the plugin:

- Determines if a parser exists for the filetype
- Passes the file to the parser
- Saves the output to the cruise data directory
- Updates the data dashboard manifest

Plugins

Look up array passed to the plugin to determine the correct parser and data_type for the given file.

```
from server.plugins.parsers.gga_parser   import GGAParser
from server.plugins.parsers.met_parser   import MetParser
from server.plugins.parsers.svp_parser   import SVPParser
from server.plugins.parsers.tsg_parser   import TSGParser
from server.plugins.parsers.twind_parser import TWindParser

# -----------------------------------------------------------
# This array defines the various dataTypes collected by
# OpenRVDAS and the corresponding file regex expression.
# -----------------------------------------------------------
fileTypeFilters = [
      {"data_type":"gga",   "regex": "*/raw/POSMV_GGA-*.txt", "parser": "GGA",   'parser_options':{}},
      {"data_type":"met",   "regex": "*/raw/MET-*.txt",       "parser": "Met",   'parser_options':{}},
      {"data_type":"svp",   "regex": "*/raw/SVP-*.txt",       "parser": "SVP",   'parser_options':{}},
      {"data_type":"tsg",   "regex": "*/raw/TSG_Raw-*.txt",   "parser": "TSG",   'parser_options':{}},
      {"data_type":"twind", "regex": "*/proc/TrueWind-*.txt", "parser": "TWind", 'parser_options':{}}
]
```

Plugins

Interface:

get_parser(filepath)

- Returns the appropriate parser for the file

get_data_type(filepath)

- Returns the data_type string for the file

get_json_str(filepath)

- Returns the data_type string for the file

```
from server.lib.openvdm_plugin import OpenVDMPlugin

class OpeRVDASPlugin(OpenVDMPlugin):
      def __init__(self):
            super().__init__(fileTypeFilters)

      def get_parser(self, filepath):
            file_type_filter = list(
                  filter(lambda file_type_filter:
                        fnmatch.fnmatch(filepath, file_type_filter['regex']),
                        self.file_type_filters)
            )

            if len(file_type_filter) == 0:
            return None

            file_type_filter = file_type_filter[0]

            if file_type_filter['parser'] == "GGA":
            return GGAParser(**file_type_filter['parser_options'])

            if file_type_filter['parser'] == "SVP":
                  return SVPParser(**file_type_filter['parser_options'])

            if file_type_filter['parser'] == "Met":
                  return MetParser(**file_type_filter['parser_options'])

            return None
```

Plugins

Interface:

Optional Args: --dataType

Required Args: dataFile

```
if __name__ == "__main__":
      parser = argparse.ArgumentParser(description='OpenVDM plugin for OpenRVDAS')
      parser.add_argument('--dataType', action='store_true',
                        help='return the dataType of the file')
      parser.add_argument('dataFile', metavar='dataFile',
                        help='the raw data file to process')

      parsed_args = parser.parse_args()

      if not os.path.isfile(parsed_args.dataFile):
            logging.error("File not found")
            sys.exit(1)
      if os.stat(parsed_args.dataFile).st_size == 0:
            logging.warning("File is empty")
            sys.exit(0)

      plugin = OpeRVDASPlugin()

      if parsed_args.dataType:
            dataType = plugin.get_data_type(parsed_args.dataFile)
            if dataType is None:
                  logging.warning("File is of unknown type")
                  sys.exit(1)
            print(dataType)
      else:
            jsonSTR = plugin.get_json_str(parsed_args.dataFile)
            if jsonSTR is None:
                  logging.warning("Nothing returned from parser")
                  sys.exit(1)
            print(jsonSTR)
```

Parsers

Parsers are called to process files.

When called, the parser:

- Ingests the file into a Pandas dataframe
- Collects Statistics
- Runs QA Tests
- Sub-samples the data
- Outputs the Stats, QA Tests results and sub-sampled data to a JSON-formatted file.

Parsers

JSON Output Format

- VisualizerData
- QualityTests
- Stats

```
{
  "visualizerData": [
    {
      "data": [
        ...
      ],
      “unit": "m/s",
      "label": "Sound Velocity"
    }
  ],
  "qualityTests": [
    { "testName": "DeltaT", "results": "Passed" }
  ],
  "stats": [
    {
      "statName": "DeltaT Bounds",
      "statType": "bounds",
      "statUnit": "seconds",
      "statValue": [ 0.0, 0.266 ]
    },
    {
      "statName": "Sound Velocity Bounds",
      "statType": "bounds",
      "statUnit": "m/s",
      "statValue": [ 1443.83, 1534.3 ]
    }
  ]
}
```

Parsers

VisualizerData

- Supports multiple objects
- Time series Object contains:
    - Data
    - Unit
    - Label
- GeoJSON also supported

```
{
      "visualizerData": [
      {
            "data": [
            [ 1335657660000, 1533.85 ],
            [ 1335657720000, 1533.85 ],
            [ 1335657780000, 1533.85 ],
            [ 1335657840000, 1533.85 ],
            [ 1335657900000, 1533.86 ],
            [ 1335657960000, 1533.86 ],
            [ 1335658020000, 1533.86 ],
            ...
            [ 1335711840000, 1525.89 ]
            ],
            "unit": "m/s",
            "label": "Sound Velocity"
      }
      ]
}
```

Parsers

Quality Tests

- Supports multiple objects
- Each object contains:
  - testName
  - testValue
- Valid values are:
  - Passed
  - Failed
  - Warning

```
"qualityTests": [
  {
    "testName": "Rows",
    "results": "Passed"
  },
  {
    "testName": "DeltaT",
    "results": "Passed"
  },
  {
    "testName": "Velocity",
    "results": "Passed"
  }
]
```

Parsers

Stats

- Supports multiple objects
- Each object contains:
    - statName
    - statType
    - statUnit
    - statValue

```
"stats": [
        {
                "statName": "Row Validity",
                "statType": "rowValidity",
                "statUnit": "",
                "statValue": [ 541850, 4 ]
        },
        {
                "statName": "Temporal Bounds",
                "statType": "timeBounds",
                "statUnit": "seconds",
                "statValue": [
                "2012-04-29T00:00:00.312000Z",
                "2012-04-29T15:03:06.023000Z"
                ]
        },
        {
                "statName": "DeltaT Bounds",
                "statType": "bounds",
                "statUnit": "seconds",
                "statValue": [ 0.0, 0.266 ]
        },
        {
                "statName": "DeltaT Validity",
                "statType": "valueValidity",
                "statUnit": "",
                "statValue": [ 541849, 0 ]
        },
]
```

Parsers

Stats

Format of statValue depends on statType

StatTypes:

- bounds - min/max
- timeBounds - start/stop
- geoBounds - bounding box
- totalValue - sum value
- valueValidity - value percentage good
- rowValidity - row percentage good

```
"stats": [
        {
                "statName": "Row Validity",
                "statType": "rowValidity",
                "statUnit": "",
                "statValue": [ 541850, 4 ]
        },
        {
                "statName": "Temporal Bounds",
                "statType": "timeBounds",
                "statUnit": "seconds",
                "statValue": [
                "2012-04-29T00:00:00.312000Z",
                "2012-04-29T15:03:06.023000Z"
                ]
        },
        {
                "statName": "DeltaT Bounds",
                "statType": "bounds",
                "statUnit": "seconds",
                "statValue": [ 0.0, 0.266 ]
        },
        {
                "statName": "DeltaT Validity",
                "statType": "valueValidity",
                "statUnit": "",
                "statValue": [ 541849, 0 ]
        },
]
```

Parsers

- Define how to parse the csv file
- Define what fields will be in the output
- Define how to round data in output
- Define max time delta between data rows

```
RAW_COLS = ['date_time', 'hdr', 'sensor_date', 'sensor_time',
'air_pres', 'air_temp', 'humidity', 'vector_wind_spd',
'vector_wind_dir', 'scalar_wind_spd', 'max_wind_spd', 'checksum']

PROC_COLS = ['date_time', 'air_pres', 'air_temp', 'humidity',
'vector_wind_spd', 'vector_wind_dir']

ROUNDING = {
      'air_pres': 1,
      'air_temp': 2,
      'humidity': 1,
      'vector_wind_spd': 1,
      'vector_wind_dir': 1
}

MAX_DELTA_T = pd.Timedelta('10 seconds')
```

Parsers

- Instantiate the OpenVDMCSVParser class
- Override the process_file method

```
from server.lib.openvdm_plugin import OpenVDMCSVParser

class MetParser(OpenVDMCSVParser):

      def __init__(self, start_dt=None, stop_dt=None,
                   time_format=None, skip_header=False,
                   use_openvdm_api=False):
            super().__init__(RAW_COLS, PROC_COLS, start_dt=start_dt,
                             stop_dt=stop_dt, time_format=time_format,
                             skip_header=skip_header,
                             use_openvdm_api=use_openvdm_api)

      def process_file(self, filepath):
```

Parsers

Override the process_file method

- Parse data

```
def process_file(self, filepath):
  raw_into_df = { value: [] for key, value in enumerate(self.proc_cols) }
  errors = []

  try:
    with open(filepath, mode='r', encoding="utf-8") as csvfile:
      reader = csv.DictReader(csvfile, self.raw_cols)

      for lineno, line in enumerate(reader):

        try:
          date_time = line['date_time'] # OpenRVDAS style

          air_pres = float(line['air_pres'])
          air_temp = float(line['air_temp'])
          humidity = float(line['humidity'])

        except Exception as err:
          errors.append(lineno)

        else:
          raw_into_df['date_time'].append(date_time)
          raw_into_df['air_pres'].append(air_pres)
          raw_into_df['air_temp'].append(air_temp)
          raw_into_df['humidity'].append(humidity)

  except Exception as err:
    logging.error(str(err))
    return
```

Parsers

Override the process_file method

- Ingest data into Pandas dataframe
- Optionally crop data

```
# If no data ingested from file, quit
if len(raw_into_df['date_time']) == 0:
  logging.warning("Dataframe is empty... quitting")
  return

# Build DataFrame
logging.debug("Building dataframe from parsed data...")
df_proc = pd.DataFrame(raw_into_df)

# Convert Date/time column to datetime objects
logging.debug("Converting data_time to datetime datatype...")

df_proc['date_time'] = pd.to_datetime(df_proc['date_time'],
                                      format=self.time_format)

# Optionally crop data by start/stop times
if self.start_dt or self.stop_dt:
  logging.debug("Cropping data...")
  df_proc = self.crop_data(df_proc)

# If the crop operation emptied the dataframe, quit
if df_proc.shape[0] == 0:
  logging.warning("Cropped dataframe is empty... quitting")
  return
```

Parsers

- Collect Stats
- Run QA Tests

```
logging.debug('Building deltaT column...')
df_proc = df_proc.join(df_proc['date_time'].diff().to_frame(name='deltaT'))

logging.debug("Tabulating statistics...")
self.add_row_validity_stat([len(df_proc), len(errors)])

self.add_time_bounds_stat([df_proc['date_time'].min(),
                           df_proc['date_time'].max()])

self.add_bounds_stat([round(df_proc['deltaT'].min().total_seconds(),3),
                     round(df_proc['deltaT'].max().total_seconds(),3)],
                     'DeltaT Bounds', 'seconds')

self.add_value_validity_stat([len(df_proc[(df_proc['deltaT'] <= MAX_DELTA_T)]),
                             len(df_proc[(df_proc['deltaT'] > MAX_DELTA_T)])],
                             'DeltaT Validity')

self.add_bounds_stat([round(df_proc['vector_wind_spd'].min(),
                     ROUNDING['vector_wind_spd']),
                     round(df_proc['vector_wind_spd'].max(),
                     ROUNDING['vector_wind_spd'])],
                     'Wind Speed Bounds', 'm/s')
...

# % of time gaps in data
error_rate = len(df_proc[(df_proc['deltaT'] > MAX_DELTA_T)]) / len(df_proc)
if error_rate > .25:
  self.add_quality_test_failed("DeltaT")
elif error_rate > .10:
  self.add_quality_test_warning("DeltaT")
else:
  self.add_quality_test_passed("DeltaT")
...
```

Parsers

- Sub-sample data
- Round data for readibility
- Build visualizerData objects

```
logging.debug("Resampling data...")
df_proc = self.resample_data(df_proc)

logging.debug("Rounding data: %s", ROUNDING)
df_proc = self.round_data(df_proc, ROUNDING)

logging.debug("Building visualization data...")

visualizer_data_obj = {'data':[], 'unit':'', 'label':''}
visualizer_data_obj['data'] = json.loads(df_proc[['date_time','air_pres']]
                              .to_json(orient='values'))
visualizer_data_obj['unit'] = 'mBar'
visualizer_data_obj['label'] = 'Air Pressure'
self.add_visualization_data(deepcopy(visualizer_data_obj))

visualizer_data_obj['data'] = json.loads(df_proc[['date_time','air_temp']]
                              .to_json(orient='values'))
visualizer_data_obj['unit'] = 'C'
visualizer_data_obj['label'] = 'Air Temperature'
self.add_visualization_data(deepcopy(visualizer_data_obj))

visualizer_data_obj['data'] = json.loads(df_proc[['date_time','humidity']]
                              .to_json(orient='values'))
visualizer_data_obj['unit'] = '%'
visualizer_data_obj['label'] = 'Relative Humidity'
self.add_visualization_data(deepcopy(visualizer_data_obj))

...

# send message about errors encountered to OpenVDM
if self.openvdm is not None and len(errors) > 0:
  self.openvdm.send_msg('Parsing Error',
  f'Error(s) parsing datafile {filepath} on row(s):'
  f'{", ".join(condense_to_ranges(errors))}')
```

Parsers

Interface:

Optional Args:

--timeFormat
--startDT
--stopDT

Required Args: dataFile

```
if __name__ == "__main__":
      import argparse

      parser = argparse.ArgumentParser(description='Parse Met Sensor data')
      parser.add_argument('--timeFormat', help='timestamp format',
                          default=None)
      parser.add_argument('--startDT', default=None,
                          type=lambda s: datetime.strptime(s,
                          '%Y-%m-%dT%H:%M:%S.%fZ'),
                          help=' crop start timestamp (iso8601)')
      parser.add_argument('--stopDT', default=None,
                          type=lambda s: datetime.strptime(s,
                          '%Y-%m-%dT%H:%M:%S.%fZ'),
                          help=' crop stop timestamp (iso8601)')
      parser.add_argument('dataFile', metavar='dataFile',
                          help='the raw data file to process')

      parsed_args = parser.parse_args()

      ovdm_parser = MetParser(start_dt=parsed_args.startDT,
                              stop_dt=parsed_args.stopDT,
                              time_format=parsed_args.timeFormat)
      try:
            ovdm_parser.process_file(parsed_args.dataFile)
            print(ovdm_parser.to_json())
      except Exception as err:
            logging.error(str(err))
            raise err
```

OpenVDM

- Introduction - what/why/where
- Lingo 101
- Whole system overview - installation, Web-UI tour, defining/controlling transfers
- Hooks - attaching processes to key points in processes
- Displaying data - plugins and parsers
- **Leveraging OpenVDM data elsewhere**
- Best practices
- Contributing
- Where to from here?

OpenVDM API

Most of the configuration and status data is accessible from the OpenVDM API.

Allows vessel operators to leverage the information in other systems (OpenRVDAS and Sealog)

```
//API-related routes
'api/warehouse/getCruiseConfig'
'api/warehouse/getCruiseID'
'api/warehouse/getCruiseSize'
'api/warehouse/getCruiseStartDate'
'api/warehouse/getCruiseEndDate'
'api/warehouse/getCruiseStartPort'
'api/warehouse/getCruiseEndPort'
'api/warehouse/getFreeSpace'
'api/warehouse/getSystemStatus'

'api/collectionSystemTransfers/getCollectionSystemTransfers'
'api/collectionSystemTransfers/getActiveCollectionSystemTransfers'
'api/collectionSystemTransfers/getCollectionSystemTransfer/(:num)'
'api/collectionSystemTransfers/getCollectionSystemTransfersStatuses'

'api/cruiseDataTransfers/getCruiseDataTransfers'
'api/cruiseDataTransfers/getCruiseDataTransfer/(:num)'
'api/cruiseDataTransfers/getCruiseDataTransfersStatuses'
```

OpenVDM API

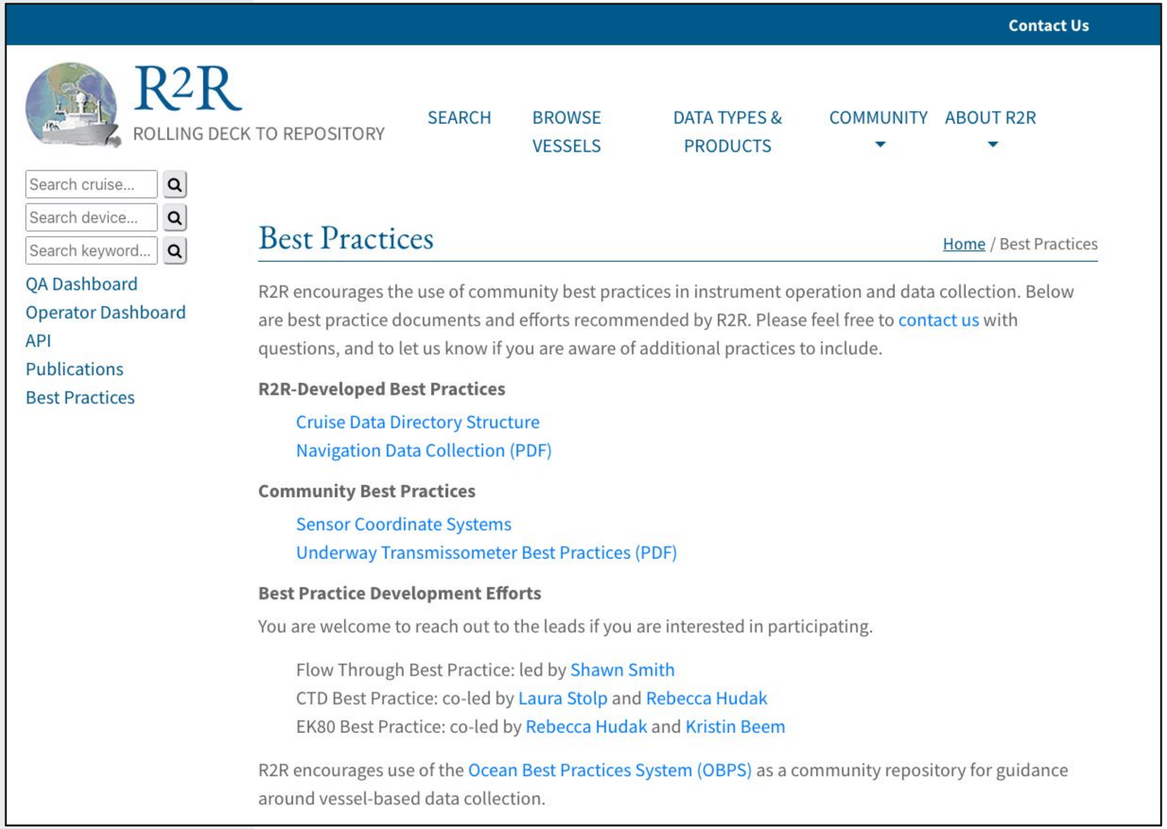
```
OPENVDM_SERVER_URL="http://192.168.0.42"

query_api() {

   CRUISE_ID=`curl -s "${OPENVDM_SERVER_URL}/api/warehouse/getCruiseID" |
      python3 -c "import sys, json; print(json.load(sys.stdin)['cruiseID'])"`

   echo "Cruise ID: ${CRUISE_ID}"

   CRUISE_START_DATE=`curl -s "${OPENVDM_SERVER_URL}/api/warehouse/getCruiseStartDate" |
      python3 -c "import sys, json; print(json.load(sys.stdin)['cruiseStartDate'].split()[0])" | sed 's?/?-?g'`

   echo "Cruise Start Date: ${CRUISE_START_DATE}"

   CRUISE_END_DATE=`curl -s "${OPENVDM_SERVER_URL}/api/warehouse/getCruiseEndDate" |
      python3 -c "import sys, json; print(json.load(sys.stdin)['cruiseEndDate'].split()[0])" | sed 's?/?-?g'`

   echo "Cruise End Date: ${CRUISE_END_DATE}"

}
```

OpenVDM

- Introduction - what/why/where
- Lingo 101
- Whole system overview - installation, Web-UI tour, defining/controlling transfers
- Hooks - attaching processes to key points in processes
- Displaying data - plugins and parsers
- Leveraging OpenVDM data elsewhere
- **Best practices**
- Contributing
- Where to from here?

Best Practices

Why reinvent the wheel?



Best Practices

R2R encourages the use of community best practices in instrument operation and data collection. Below are best practice documents and efforts recommended by R2R. Please feel free to contact us with questions, and to let us know if you are aware of additional practices to include.

R2R-Developed Best Practices

- Cruise Data Directory Structure
- Navigation Data Collection (PDF)

Community Best Practices

- Sensor Coordinate Systems
- Underway Transmissometer Best Practices (PDF)

Best Practice Development Efforts

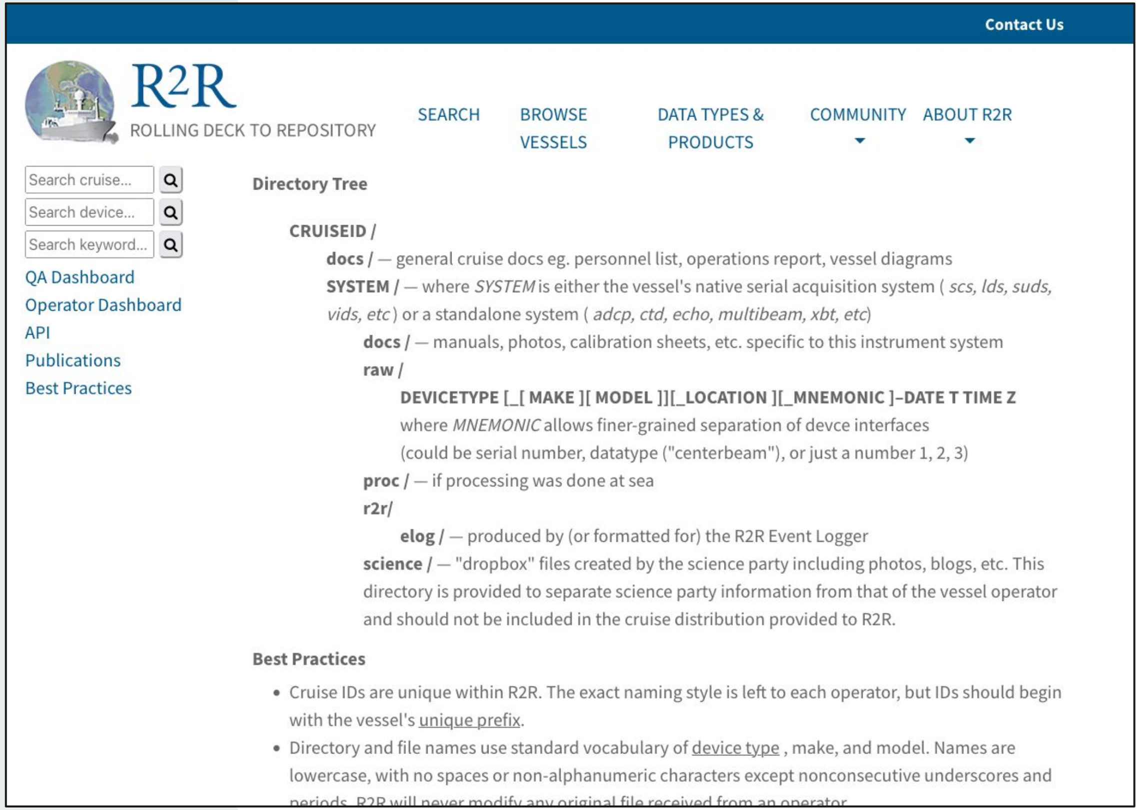
You are welcome to reach out to the leads if you are interested in participating.

- Flow Through Best Practice: led by Shawn Smith
- CTD Best Practice: co-led by Laura Stolp and Rebecca Hudak
- EK80 Best Practice: co-led by Rebecca Hudak and Kristin Beem

R2R encourages use of the Ocean Best Practices System (OBPS) as a community repository for guidance around vessel-based data collection.

Best Practices

Why reinvent the wheel?



Directory Tree

CRUISEID /
- **docs /** — general cruise docs eg. personnel list, operations report, vessel diagrams
- **SYSTEM /** — where *SYSTEM* is either the vessel's native serial acquisition system (*scs, lds, suds, vids, etc*) or a standalone system (*adcp, ctd, echo, multibeam, xbt, etc*)
  - **docs /** — manuals, photos, calibration sheets, etc. specific to this instrument system
  - **raw /**
    - **DEVICETYPE [_[MAKE][MODEL]][_LOCATION][_MNEMONIC]–DATE T TIME Z**
      where *MNEMONIC* allows finer-grained separation of devce interfaces
      (could be serial number, datatype ("centerbeam"), or just a number 1, 2, 3)
  - **proc /** — if processing was done at sea
  - **r2r/**
    - **elog /** — produced by (or formatted for) the R2R Event Logger
  - **science /** — "dropbox" files created by the science party including photos, blogs, etc. This directory is provided to separate science party information from that of the vessel operator and should not be included in the cruise distribution provided to R2R.

Best Practices

- Cruise IDs are unique within R2R. The exact naming style is left to each operator, but IDs should begin with the vessel's unique prefix.
- Directory and file names use standard vocabulary of device type , make, and model. Names are lowercase, with no spaces or non-alphanumeric characters except nonconsecutive underscores and periods. R2R will never modify any original file received from an operator.

OpenVDM

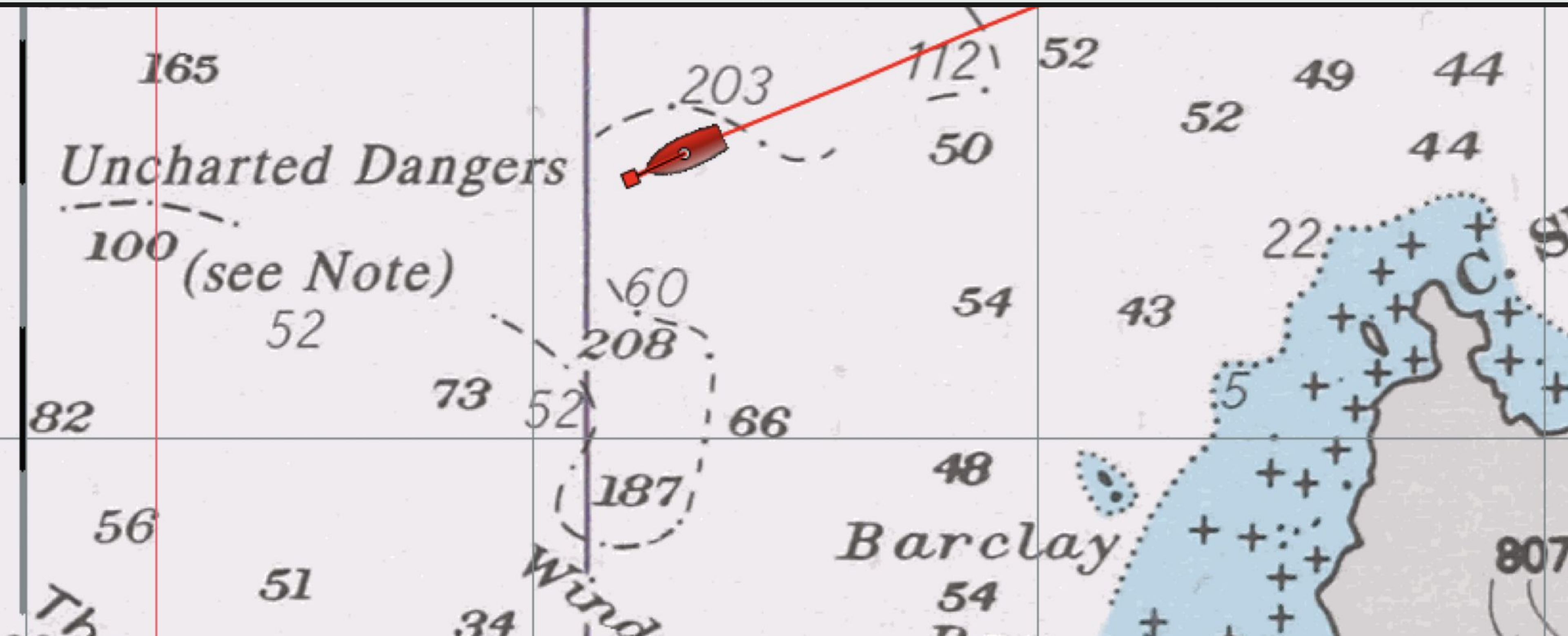
- Introduction - what/why/where
- Lingo 101
- Whole system overview - installation, Web-UI tour, defining/controlling transfers
- Hooks - attaching processes to key points in processes
- Displaying data - plugins and parsers
- Leveraging OpenVDM data elsewhere
- Best practices
- **Contributing**
- Where to from here?

Contributing to OpenVDM

Because sharing is caring! ❤️

- Bug reports/feature requests: **https://github.com/OceanDataTools/openvdm/issues**

Where to from here?



Where to from here?

- OpenVDM 3.0 is on the horizon
- Ditching PHP for python/nodeJS
- Multi-platform support (ROVs, AUVs, HOVs, etc)
- Cloud storage destination support
- Alternative OS support

http://tinyurl.com/oceandatatools-rvtec-2024