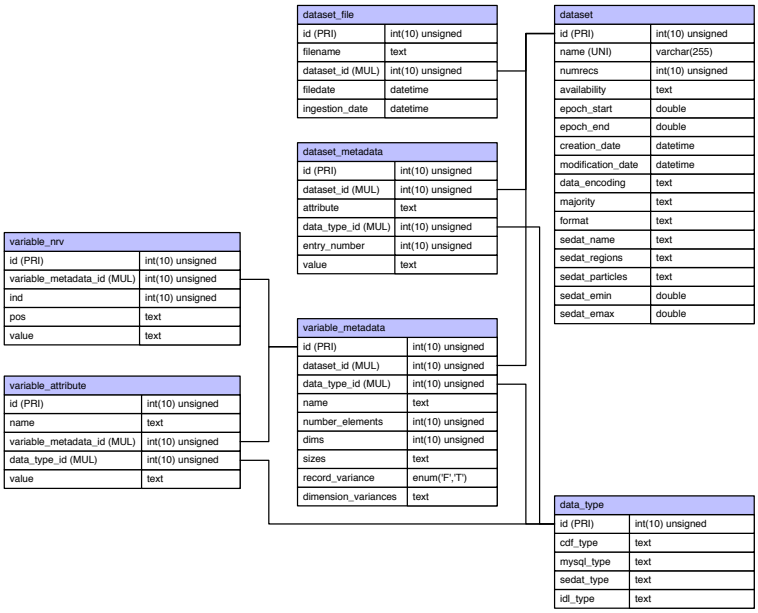
ODI Administrator Guide
ESTEC/Contract No. 21964/08/NL/AT

March 12, 2010



Peter Wintoft, Swedish Institute of Space Physics

Daniel Heynderickx, DH Consultancy

Document status sheet

Version	Date	Comment
1.0	2009-06-01	First version.
1.1	2009-09-25	Updated.
1.2	2009-10-16	Updated.
1.3	2009-12-15	Updated.
1.4	2010-03-12	Summary updated. Prerequisites updated. Installing the ODI software updated. New section: Creating the core ODI tables. New section: Downloading data. Ingesting data updated. Updating the database with live data updated. Command line tools updated. Scripts for setting up the ODI database removed.

Source document: `$Id: odi_admin_guide.tex 11 2010-03-30 13:11:04Z peter $`

Contents

1 Summary

The Open Data Interface (ODI) was developed to store space physics data and metadata compliant with the CDF/ISTP/PRBEM guide lines. The ODI system shall also be capable to ingest data from various sources, such as ASCII files, HTML pages, and CDF files. As a CDF file contains a complete description of metadata, and as the ISTP guide lines builds on CDF, is is natural to use the CDF metadata syntax in ODI. However, the data is not stored in files in ODI but rather in a database and it was chosen to use MySQL.

In addition to the core ODI system three software packages have been upgraded to make use of ODI, namely: SAAPS, SEDAT, and SPENVIS. Theses packages are part of the ODI distribution.

In this manual we describe how to set up the database and how to ingest data.

2 Prerequisites

The ODI system builds on a set of different software components. There should also exist three different user levels that have different degrees of access privileges. The local settings are defined in a set of environment variables. These items will be discussed in the following.

2.1 Software

The core ODI system requires MySQL, PHP, and Wget. If CDF files are to be imported the CDF software is also required. If it is required to export ODI data to CDF files then the IDL and IDL/Dataminer are also necessary. The necessary software are listed in the table below. It is also described for which part of the system the software is critical.

Software	**System**
MySQL 5.0 or higher	ODI
PHP 5.2 or higher	ODI
Wget 1.10 or higher	Download of external datasets
Cron	Automatic updates of datasets
CDF program	To ingest data from CDF files into ODI.
IDL and Dataminer	ODI CDF Export tool
Java 2 Runtime Environment, version 1.5 or higher	SAAPS
IDL and Dataminer	SEDAT
IDL and Dataminer	SPENVIS
Perl and Perl DBI	SEDAT

The PHP software must be compiled with MySQL support, which probably most

binaries are. The PHP initialisation file (`php.ini`) may be edited if necessary. E.g. the `allow_url_fopen` flag must be set to `On` for the parsing of remote files. The location of the `php.ini` file can be found by executing `php --info`.For more details on PHP and MySQL see `http://php.net/manual/en/book.mysql.php`.

The Java libraries for the MySQL connector must be available. The libraries are distributed with the ODI software. See also `http://dev.mysql.com/doc/refman/-5.1/en/connector-j.html`.

2.2 Database and users

A MySQL database must also be created with three levels of users with different privileges. In the following we assume the database is named `odi`. The three users levels are

User Level 1 (UL1) which has all privileges on the database,

User Level 2 (UL2) which has select, update, and insert privileges,

User Level 3 (UL3) which has only select privileges.

The UL1 user is typically an administrator user of the database. The UL2 user is used by the automatic routines that updates the database. Finally, the UL3 user is the normal end user or application who only needs to pull data out of the system. The names and passwords of each UL are specified in the environment variables below.

3 Installing the ODI software

An archive of the ODI software can be found at `http://www.lund.irf.se/odi/`. Download this file and assume that is is placed in `my_dir`. To extract the files do the following:

```
>> cd my_dir
>> tar xvf odi-rev278.tar.gz
```

where `>>` means the Unix shell. For this to work the current directory (".") must be in the search path, alternatively the commands must be be preceded with "./". The file `odi-rev278.tar.gz` should be replaced with the actual name of the downloaded file. This will create the directory `odi` containing the following:

`doc`	Various documentation, e.g. this page.
`lib`	ODI library code.
`parsers`	Code to parse the raw datasets and ingest them into ODI.
`readme.txt`	Various information.
`revision.txt`	The revision number of the installed ODI system.
`SAAPS`	The SAAPS system.
`SEDAT`	The SEDAT system.
`setup_database`	Code to create the core ODI database tables.
`SPENVIS`	The SPENVIS system.
`test`	Various test code.
`tools`	Various tools to use on the ODI database.

The next step is to define necessary environment variables by first copying the `setup.sh.template` file:

```
>> cd odi/setup_database
>> cp setup.sh.template setup.sh
```

and then edit `setup.sh` to reflect the local settings. The setup file sets the ODI environment variables. The variables have the following purpose:

`$ODI_HOME`	Required	The base directory where ODI is installed.
`$ODI_RAWDATA`	Required	The base directory where the imported files are stored.
`$ODI_DB`	Required	The name of the database.
`$ODI_HOST`	Optional	The host name of the MySQL server.
`$ODI_PORT`	Optional	The port number through which to communicate with the MySQL server.
`$ODI_SOCKET`	Optional	The MySQL socket.
`$ODI_USER_1`	Required	The name of the User Level 1 user. This user has all privileges on the tables in the ODI database.
`$ODI_PW_1`	Required	The password of the User Level 1 user.
`$ODI_USER_2`	Required	The name of the User Level 2 user. This user has the privileges to read data from and write data to the ODI database.
`$ODI_PW_2`	Required	The password of the User Level 2 user.
`$ODI_USER_3`	Required	The name of the User Level 3 user. This user has only privileges read data from the ODI database.
`$ODI_PW_3`	Required	The password of the User Level 3 user.

The variables `$ODI_RAWDATA`, `$ODI_DB`, `$ODI_USER_*`, and `$ODI_PW_*` can be set to any values decided by the ODI administrator.

If the optional environment variables are not set then the default values are used:

```
ODI_HOST=localhost
ODI_PORT=3306
ODI_SOCKET=/tmp/mysql.sock
```

The the setup script is run by simply typing

```
>> setup.sh
```

Preferably a call to this script should be placed in a file read at log-in, e.g. `.profile`. All the scripts in this section are listed in Appendix **??**.

4 Creating the ODI database

The next step is to create the ODI database. It is assumed that the MySQL server is running. The MySQL root user name and password must also be known.

First the database create script must be copied and edited:

```
>> cd $ODI_HOME/setup_database
>> cp create_database.sh.template create_database.sh
```

Edit `create_database.sh` so that is has the correct root and password for the MySQL server. Then run the create database script

```
>> create_database.sh
```

which will create a database with the name given by `$ODI_DB` with the correct privileges for the different user levels.

5 Creating the core ODI tables

To create the core ODI tables the following is executed

```
>> drop_all_tables.php
>> create_odi_tables.sh
```

where the first command removes all tables in the ODI database. If the database was just created there will be no tables and the command is not necessary. The second command will create the 7 ODI tables which are shortly described below, illustrated in Figure 1, and listed in Appendix A.

`dataset`

The table dataset contains global information of all datasets that exist in ODI and is the top level description of a dataset in ODI. This includes the name of the table holding the data for a specific dataset. The names of the data tables always start with `dataset_` followed by a unique name, e.g. `dataset_rosetta_srem_pacc`. The information in the CDF header block also goes into dataset, namely: `DATA ENCODING`, `MAJORITY`, and `FORMAT`. The `CDF NAME` is not stored as it changes when new files are ingested into the dataset. However, `CDF NAME` can be recreated from the information stored in the database.

`dataset_metadata`

The table `dataset_meta` contains the next level of metadata for each dataset. This metadata comes from the global attributes block in the CDF skeleton: `Attribute Name`, `Entry Number`, `Data Type`, and `Value`.

`variable_metadata`

The table `variable_metadata` contains the metadata for each variable. In the CDF skeleton each z-Variable is defined by `Variable Name`, `Data Type`, `Number Elements`, `Dims`, `Sizes`, `Record Variance`, and `Dimension Variances`. All this information goes into `variable_metadata`.

`variable_attribute`

To each variable in the CDF dataset there are a varying number of associated attributes.

The `Attribute Name`, `Data Type`, and `Value` for each attribute are stored in the table `variable_attribute`.

`variable_nrv`

Any data that does note vary from record to record has the CDF "Record Variance"=F(alse) and is known as non-record-variant (nrv) data. The nrv values appears in the skeleton file and are stored in the table `variable_nrv`.

`data_type`

The table `data_type` contains all CDF data types and associated MySQL and SEDAT data types.

`dataset_file`

The table `dataset_file` stores the names and dates of all files that have been ingested into ODI.

At this stage, the only table that contains data is the `data_type` table. The other tables will later on be populated with various metadata.

6 Downloading data

Data that shall be stored in the ODI database can first be downloaded into the `$ODI_RAWDATA` directory. The data must be organised into subdirectories under `$ODI_RAWDATA` where the name of the subdirectory should correspond to the `<Data directory>` entry in the `datasets.txt` file (see next section and Appendix B). Examples of subdirectories are `ACE/MFI`, `GOES/SEM` and `index/Dst`, where the directories are named according to platform/instrument (although this syntax is not mandatory).

Under `$ODI_HOME/parsers/download/` there are several scripts for downloading data using the Wget program. New scripts can easily be added by copying and editing the existing scripts.

7 Ingesting data

The procedure to ingest data is to execute the `populate.php` program (in `$ODI_HOME/parsers/`) which will read the file `datasets.txt`.

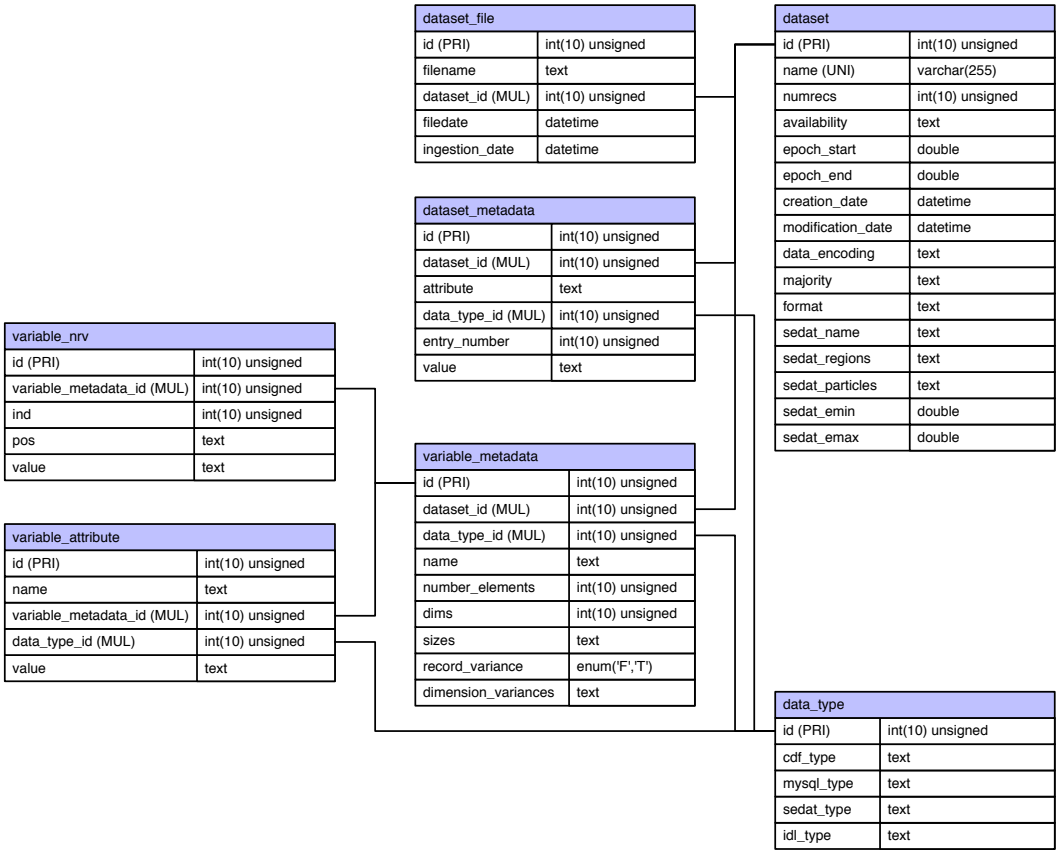


Figure 1: The ODI metadata tables.

```
>> cd $ODI_HOME/parsers
>> populate.php [<dataset 1> [<dataset 2> ... ] ]
```

In order for this to work a number of files are needed. Firstly, the *dataset definition file* and the *skeleton file* must exist. The syntax of these files are given in Appendix B together with examples. Then the *raw data files* must exist, either locally under `$ODI_RAWDATA` or remotely. If the raw data files are CDF files then there must also exist a CDF *settings file* (Appendix B.3).

When `populate.php` is executed without any arguments all lines in the `datasets.txt` file that do not start with a `#`-character are parsed. If `populate.php` is run with arguments like

```
populate.php <dataset 1> <dataset 2> ...
```

then only the lines with matching datasets in `datasets.txt` will be parsed, irrespective of whether there is a leading `#`-character.

When the raw data comes from CDF files they are automatically converted to text files with comma-separated fields using the CDFexport program. The text files are ingested into the `dataset_` table. It is **very important to order the data fields correctly** in the text files, which is controlled by the settings file (Appendix B.3), to match the order of the columns in the `dataset_*` table. The columns in the `dataset_*` table have the same order as the variables are defined in the skeleton file.

When the raw data is a text file there are so many different format possibilities that it is not possible to write a general parsing routine. Therefore, to ingest text data a couple of files need to be added and edited. The procedure is described in Appendix B.4.

If the dataset does not already exist in the ODI database the `datasets.txt` and skeleton files will be parsed and the data will be stored in the ODI metadata tables. Then the corresponding `dataset_*` table is created. Next, the actual data files, in `$ODI_RAWDATA`, are parsed and ingested into the `dataset_*` table. However, a check is first made on the data file name to see whether it has already been stored in ODI and if it has no parsing or ingesting will be performed. The procedure is illustrated in Figure 2.

To summarise, the following steps must be performed to add a new dataset to the ODI database:

1. Add a row to the `datasets.txt` file.
2. Add a directory under `$ODI_RAWDATA` with subdirectories according to the entry `<Data directory>` in `datasets.txt`.
3. In `$ODI_HOME/parsers` create a directory named `<platform>`.
4. In `$ODI_HOME/parsers/<platform>` add the file `<instrument>.skt` that contain the dataset metadata according to CDF skeleton syntax.
5. Then depending on whether it is a text file or CDF file to be ingested:
   (a) If it is a text file see Appendix B.4.
   (b) If it is a CDF file add the file `<instrument>.set` under `$ODI_HOME/parsers/-<platform>` that contain the CDF export configuration.
6. Run `popoulate.php [<dataset>]`.

8 Updating the database with live data

Datasets in the ODI database can be updated automatically using the Cron program. Jobs to be run by Cron are added using the `crontab` program. To ensure the environment variables are available to Cron it is recommended to use the `cronjobs_install.php` program in `$ODI_HOME/parsers`.

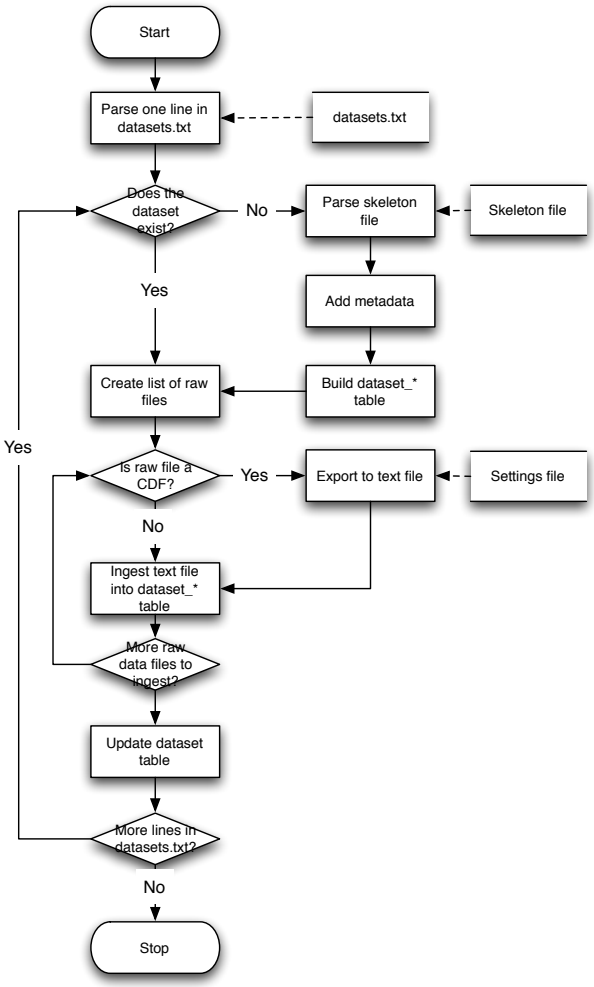


Figure 2: The diagram illustrates the procedure to ingest data into ODI.

The `cronjobs_install.php` program reads the file `cronjobs.txt` (also in `$ODI_HOME/parsers`). The entries in `cronjobs.txt` follows the Cron syntax. To start Cron jobs do

```
>> cd $ODI_HOME/parsers
>> cronjobs_install.php
```

9 Command line tools

Under the directories `$ODI_HOME/setup_database/` and `$ODI_HOME/tools/` there are a set of programs that can be used on the ODI database. They are listed here in alphabetical order.

9.1 Tools in `$ODI_HOME/setup_database/`

`add_revision_info.php`

Add the revision number to the comment field in the ODI tables. This program is automatically run during installation.

`create_database.sh`

Create the users with passwords and privileges on the MySQL server. This program is only run at first time installation.

`create_odi_tables.sh`

Create the ODI tables that will hold the metadata. The program calls `metadata_tables_create.sql` which contains the SQL commands that creates the tables. This program is only run at first time installation.

`drop_all_tables.php`

Delete all tables in ODI.

`setup.sh`

Create all necessary environment variables. This script should also be called by the log-in script (like `.profile`).

9.2 Tools in `$ODI_HOME/tools/`

`check_epoch.php`

Check if the epochs in a dataset are equidistant. The program determines the median difference between consecutive epoch values and then lists all epochs which have a epoch differences that are different from the median.

`check_skeleton.php <skeleton file>`

Perform a basic check on the skeleton file. This program does a simple check on the skeleton file and warns for the following: If the number of global attributes does not equal the number of defined global attributes; If the number of variable attributes does not equal the number of defined variable attributes; If the number of variables does not equal the number of defined variables; If there are attributes listed but not actually used by any variable.

`delete_dataset.php`

Delete a dataset from the ODI database. The program deletes all data and metadata for the dataset.

`export_build_xml.php`

Create an XML template file for a dataset that describes the variables and epoch range to be exported.

`export_to_xml.php`

Export a dataset to a text file in XML format. The variables to be exported and the epoch range is specified in the export template file created using the previous program. The export template file may first be edited to reflect the chosen variables and epoch range.

`show_datasets.php`

Get a list of the ODI datasets.

`show_metadata.php`

Show the metadata for a selected dataset.

`show_updates.php`

List all datasets that have been updated for a given epoch range.

A ODI metadata tables

dataset	**A table holding the names of each dataset_* table together with some key information.**
id	A unique identifier.
name	The name of the dataset table in the ODI database.
numrecs	The number of records in the data table.
availability	A flag indicating whether the dataset is public or private.
epoch_start	The first epoch in the dataset.
epoch_end	The last epoch in the dataset.
creation_date	The date when the dataset_* table was created.
modification_date	The last date when the dataset was modified.
data_encoding	The CDF header attribute DATA ENCODING.
majority	The CDF header attribute MAJORITY.
format	The CDF header attribute FORMAT.
sedat_name	SEDAT datset name without SYSTEM! prefix.
sedat_regions	SEDAT region code.
sedat_particles	SEDAT particle code.
sedat_emin	SEDAT min. energy (MeV).
sedat_emax	SEDAT max. energy (MeV).

dataset_file	**A table to hold the file names of all ingested data files.**
id	A unique identifier.
filename	The file name of the ingested file.
dataset_id	A key to the associated dataset in table dataset.
filedate	The time stamp of the raw data file.
ingestion_date	The date when the file was ingested into ODI.

dataset_metadata	**Metadata for each dataset.**
id	A unique identifier.
dataset_id	A key to the associated dataset in table dataset.
attribute	The attribute name. This corresponds to the CDF global attribute.
data_type_id	A key to the associated data type in table data_type.
entry_number	The CDF global parameter entry number.
value	The value (or contents) of the dataset attribute.

data_type	**CDF data types together with associated MySQL and SEDAT data types.**
id	A unique identifier.
cdf_type	The CDF data type.
mysql_type	The MySQL data type.
sedat_type	The SEDAT data type.
idl_type	The IDL data type.

variable_attribute	**The attributes for each variable.**
id	A unique identifier.
name	The name of the variable attribute.
variable_metadata_id	A key to the associated variable in table variable_metadata.
data_type_id	A key to the associated data type in table data_type.
value	The value (or contents) of the variable attribute.

variable_metadata	**The metadata for each variable.**
id	A unique identifier.
dataset_id	A key to the associated dataset in table dataset.
data_type_id	A key to the associated data type in table data_type.
name	The name of the variable.
number_elements	The CDF variable parameter Number Elements.
dims	The CDF variable parameter Dims.
sizes	The CDF variable parameter Sizes.
record_variance	The CDF variable parameter Record Variance.
dimension_variances	The CDF variable parameter Dimension Variances.

variable_nrv	**The values of the non-record-variant data.**
id	A unique identifier.
variable_metadata_id	A key to the associated variable in table variable_metadata.
ind	An index to the nrv variable. It goes from 1 to the product of the number of elements in each dimension.
pos	The index to the nrv variable as given in the CDF.
value	The value of the nrv variable.

B Files used for ingesting data

B.1 Dataset definition file (`datasets.txt`)

The dataset definition file, named `datasets.txt`, contains information about the dataset to be ingested that describes e.g. the directory of the raw data files, the file name pattern of the files, and the ODI dataset name. Each line in `datasets.txt` must contain 13 fields according to

```
<ODI data table name>;;; <Data directory>;;; <File name pattern>;;;\
<Platform>;;; <Platform type>;;; <Instrument>;;;\
<Skeleton file name>;;; <Availability>;;;\
<SEDAT dataset name>;;; <SEDAT region code>;;; <SEDAT particle code>;;;\
<SEDAT min. energy (MeV)>;;; <SEDAT max. energy (MeV)>
```

where the row has been split over several lines to fit into this document. Each field is separated with a triple-colon (`;;;`) and the text within angle brackets (`<>`) should be replaced with actual values. This file is parsed and the value of the field `<ODI data table name>` is stored in `dataset.name`.

The `<Data directory>` gives the location of the data files relative to `$ODI_RAWDATA`. The actual data files must not necessarily be placed directly under `<Data directory>` but can be placed in subdirectories.

The data files that are to be ingested must match the `<File name pattern>` field. The `<File name pattern>` may contain the %-sign which is treated as a wild card and will match any string.

The `<Platform>`, `<Platform type>`, and `<Instrument>` are stored in the `dataset_metadata` table together with the other global attributes for the dataset.

The `<Skeleton file name>` field gives the name of the skeleton file that shall be used.

The `<Availability>` field indicates whether the dataset should be publicly available (`public`) or not (`private`).

The SEDAT fields are stored in the table `dataset` in the columns `sedat_name`, `sedat_regions`, `sedat_particles`, `sedat_emin`, and `sedat_emax`.

When all files are present the `populate.sh` script is executed, which will create dataset tables when necessary and ingest the data.

An example `datasets.txt` file containing four datasets is given below.

```
#rosetta_srem_pacc;;;  Rosetta/SREM;;;  SREMRosetta_PACC_%.cdf.gz;;; \
  Rosetta;;;  satellite;;;  SREM;;;  SREM_PACC.skt;;;  ROSETTA_SREM_PACC;;; \
  i;;;     ep;;;     1e6;;;    1e9
#imp8_gme;;;          IMP8/GME;;;      i8_h0_gme_%.cdf;;;           \
  IMP8;;;     satellite;;;  GME;;;   GME.skt;;;        IMP8_GME;;;          \
  i;;;     epa;;;    3e5;;;    5e8
#proba1_srem_pacc;;;  PROBA1/SREM;;;   PROBA1_PACC_%.cdf;;;         \
  PROBA1;;;   satellite;;;  SREM;;;  SREM_PACC.skt;;;  PROBA1_SREM_PACC;;;  \
```

```
  m;;;     ep;;;     1e6;;;    1e9
#index_dst;;;          index/Dst;;;     %dst%;;;                        \
  index;;;    virtual;;;    Dst;;;   Dst.skt;;;        DST;;;                \
  m;;;     o;;;      0;;;      1e99
```

B.2 Skeleton file (.skt)

The skeleton file contains metadata and non-record variant data for a dataset. If the raw file to be ingested into ODI is a CDF file then the skeleton file is created by

```
>> skeletontable <CDF file>
```

where `skeletontable` is a CDF command line tool. This command will produce a skeleton file with the same name but with the `.skt` extension.

If the raw file is not a CDF file the skeleton file is still needed to ingest data into ODI. The skeleton file must then be created manually. The format of the file can be seen by the example given below. The CDF User's Guide contains a complete description of the skeleton files.

Below an example skeleton file is shown that have been produced from the Rosetta SREM CDF file.

```
! Skeleton table for the "SREMRosetta_PACC_20090101" CDF.
! Generated: Friday, 16-Oct-2009 09:36:26
! CDF created/modified by CDF V2.7.2
! Skeleton table created by CDF V3.3.0

#header

                     CDF NAME: SREMRosetta_PACC_20090101
                DATA ENCODING: IBMPC
                     MAJORITY: ROW
                       FORMAT: SINGLE

! Variables  G.Attributes  V.Attributes  Records  Dims  Sizes
! ---------  ------------  ------------  -------  ----  -----
    0/12          13            13         0/z      0


#GLOBALattributes

! Attribute         Entry       Data
! Name              Number      Type       Value
```

```
! ---------          ------      ----       -----

  "Project"             1:    CDF_CHAR     { "SREM aboard Rosetta" } .

  "Source_name"         1:    CDF_CHAR     { "Rosetta" } .

  "Discipline"          1:    CDF_CHAR     { "Space " -
                                             "Physics>Interplanetary " -
                                             "Science" } .

  "Data_type"           1:    CDF_CHAR     { "PACC>Processed " -
                                             "Accumulation Data" } .

  "Descriptor"          1:    CDF_CHAR     { "SREM>Standard Radiation " -
                                             "Environment Monitor" } .

  "Data_version"        1:    CDF_CHAR     { "preliminary" } .

  "Logical_file_id"     1:    CDF_CHAR     { "SREMRosetta_PACC_20090101" } .

  "PI_name"             1:    CDF_CHAR     { "P. Buehler, " -
                                             "paul.buehler@psi.ch" } .

  "PI_affiliation"      1:    CDF_CHAR     { "PB - Paul Buehler" } .

  "Instrument_type"     1:    CDF_CHAR     { "Particles (space)" } .

  "Mission_group"       1:    CDF_CHAR     { "Rosetta" } .

  "Logical_source"      1:    CDF_CHAR     { "SREMRosetta_PACC" } .

  "Logical_source_description"
                        1:    CDF_CHAR     { "SREM/Rosetta Processed " -
                                             "Accumulation Data" } .


#VARIABLEattributes

  "FIELDNAME"
  "VALIDMIN"
  "VALIDMAX"
  "LABLAXIS"
  "UNITS"
```

```
  "FILLVAL"
  "VAR_TYPE"
  "DICT_KEY"
  "SCALETYP"
  "MONOTON"
  "CATDESC"
  "DEPEND_0"
  "DEPEND_1"


#variables

! No rVariables.


#zVariables

! Variable          Data      Number                 Record   Dimension
! Name              Type     Elements  Dims  Sizes  Variance  Variances
! --------          ----     --------  ----  -----  --------  ---------

  "EPOCH"         CDF_EPOCH      1       0              T

  ! Attribute       Data
  ! Name            Type       Value
  ! --------        ----       -----

    "FIELDNAME"   CDF_CHAR     { "Time since 0 AD of accumulation" }
    "VALIDMIN"    CDF_EPOCH    { 01-Oct-2001 00:00:00.000 }
    "VALIDMAX"    CDF_EPOCH    { 01-Jan-2010 00:00:00.000 }
    "LABLAXIS"    CDF_CHAR     { "Epoch" }
    "UNITS"       CDF_CHAR     { "ms" }
    "FILLVAL"     CDF_DOUBLE   { -1.0e+31 }
    "VAR_TYPE"    CDF_CHAR     { "support_data" }
    "DICT_KEY"    CDF_CHAR     { "time>epoch" }
    "SCALETYP"    CDF_CHAR     { "linear" }
    "MONOTON"     CDF_CHAR     { "INCREASE" }
    "CATDESC"     CDF_CHAR     { "Accumulation interval centered epoch" } .

  ! RV values were not requested.


! Variable          Data      Number                 Record   Dimension
```

```
! Name              Type     Elements  Dims  Sizes  Variance  Variances
! --------          ----     --------  ----  -----  --------  ---------

  "label_COUNTERS"
                  CDF_CHAR       3       1     15       F         T

  ! Attribute       Data
  ! Name            Type       Value
  ! --------        ----       -----

    "FIELDNAME"   CDF_CHAR     { "SREM counter label" }
    "VAR_TYPE"    CDF_CHAR     { "metadata" }
    "CATDESC"     CDF_CHAR     { "SREM counter label" } .

  ! NRV values follow...

    [1] = { "TC1" }
    [2] = { "S12" }
    [3] = { "S13" }
    [4] = { "S14" }
    [5] = { "S15" }
    [6] = { "TC2" }
    [7] = { "S25" }
    [8] = { "C1 " }
    [9] = { "C2 " }
    [10] = { "C3 " }
    [11] = { "C4 " }
    [12] = { "TC3" }
    [13] = { "S32" }
    [14] = { "S33" }
    [15] = { "S34" }


! Variable          Data      Number                 Record   Dimension
! Name              Type     Elements  Dims  Sizes  Variance  Variances
! --------          ----     --------  ----  -----  --------  ---------

  "COUNTRATE"     CDF_DOUBLE     1       1     15       T         T

  ! Attribute       Data
  ! Name            Type       Value
  ! --------        ----       -----
```

```
    "FIELDNAME"   CDF_CHAR     { "SREM deadtime corrected count rates" }
    "VALIDMIN"    CDF_DOUBLE   { 0.0 }
    "VALIDMAX"    CDF_DOUBLE   { 4.29497e+09 }
    "LABLAXIS"    CDF_CHAR     { "Count rate" }
    "UNITS"       CDF_CHAR     { "1/sec" }
    "FILLVAL"     CDF_DOUBLE   { -1.0e+31 }
    "VAR_TYPE"    CDF_CHAR     { "data" }
    "DICT_KEY"    CDF_CHAR     { "particle_flux>number" }
    "SCALETYP"    CDF_CHAR     { "linear" }
    "CATDESC"     CDF_CHAR     { "SREM deadtime corrected count rates" }
    "DEPEND_0"    CDF_CHAR     { "EPOCH" }
    "DEPEND_1"    CDF_CHAR     { "label_COUNTERS" } .

  ! RV values were not requested.


! Variable          Data      Number                 Record   Dimension
! Name              Type     Elements  Dims  Sizes  Variance  Variances
! --------          ----     --------  ----  -----  --------  ---------

  "label_ORBIT"   CDF_CHAR       6       1     3        F         T

  ! Attribute       Data
  ! Name            Type       Value
  ! --------        ----       -----

    "FIELDNAME"   CDF_CHAR     { "SREM orbit label" }
    "VAR_TYPE"    CDF_CHAR     { "metadata" }
    "CATDESC"     CDF_CHAR     { "SREM orbit label" } .

  ! NRV values follow...

    [1] = { "X, ECI" }
    [2] = { "Y, ECI" }
    [3] = { "Z, ECI" }


! Variable          Data      Number                 Record   Dimension
! Name              Type     Elements  Dims  Sizes  Variance  Variances
! --------          ----     --------  ----  -----  --------  ---------

  "ORBIT"         CDF_DOUBLE     1       1     3        T         T
```

```
  ! Attribute       Data
  ! Name            Type       Value
  ! --------        ----       -----

    "FIELDNAME"   CDF_CHAR     { "Rosetta position" }
    "VALIDMIN"    CDF_DOUBLE   { 0.0 }
    "VALIDMAX"    CDF_DOUBLE   { 4.29497e+09 }
    "LABLAXIS"    CDF_CHAR     { "Distance" }
    "UNITS"       CDF_CHAR     { "km" }
    "FILLVAL"     CDF_DOUBLE   { -1.0e+31 }
    "VAR_TYPE"    CDF_CHAR     { "data" }
    "DICT_KEY"    CDF_CHAR     { "position>distance" }
    "SCALETYP"    CDF_CHAR     { "linear" }
    "CATDESC"     CDF_CHAR     { "Position of Rosetta satellite in " -
                                 "ECI-system" }
    "DEPEND_0"    CDF_CHAR     { "EPOCH" }
    "DEPEND_1"    CDF_CHAR     { "label_ORBIT" } .

  ! RV values were not requested.


! Variable          Data      Number                 Record   Dimension
! Name              Type     Elements  Dims  Sizes  Variance  Variances
! --------          ----     --------  ----  -----  --------  ---------

  "FITQUAL"       CDF_INT2       1       0              T

  ! Attribute       Data
  ! Name            Type       Value
  ! --------        ----       -----

    "FIELDNAME"   CDF_CHAR     { "Quality of fit indicator" }
    "VALIDMIN"    CDF_INT2     { 1 }
    "VALIDMAX"    CDF_INT2     { 3 }
    "LABLAXIS"    CDF_CHAR     { "Quality of fit" }
    "UNITS"       CDF_CHAR     { "1" }
    "FILLVAL"     CDF_INT4     { -65535 }
    "VAR_TYPE"    CDF_CHAR     { "data" }
    "DICT_KEY"    CDF_CHAR     { "particle_flux>quality" }
    "SCALETYP"    CDF_CHAR     { "linear" }
    "CATDESC"     CDF_CHAR     { "Indicator for the quality of the " -
                                 "spectral fit" }
    "DEPEND_0"    CDF_CHAR     { "EPOCH" } .
```

```
  ! RV values were not requested.


! Variable          Data      Number                 Record   Dimension
! Name              Type     Elements  Dims  Sizes  Variance  Variances
! --------          ----     --------  ----  -----  --------  ---------

  "label_PROTPARAM"
                  CDF_CHAR      40       1     3        F         T

  ! Attribute       Data
  ! Name            Type       Value
  ! --------        ----       -----

    "FIELDNAME"   CDF_CHAR     { "Proton spectral parameter label" }
    "VAR_TYPE"    CDF_CHAR     { "metadata" }
    "CATDESC"     CDF_CHAR     { "Proton spectral parameter label" } .

  ! NRV values follow...

    [1] = { "POW, E0                                 " }
    [2] = { "POW, norm                               " }
    [3] = { "POW, index                              " }


! Variable          Data      Number                 Record   Dimension
! Name              Type     Elements  Dims  Sizes  Variance  Variances
! --------          ----     --------  ----  -----  --------  ---------

  "PROTPARAM"     CDF_DOUBLE     1       1     3        T         T

  ! Attribute       Data
  ! Name            Type       Value
  ! --------        ----       -----

    "FIELDNAME"   CDF_CHAR     { "Proton spectral parameters, POWER LAW," -
                                 " f=norm*(E/E0)^index                  " -
                                 "                                    " }
    "VALIDMIN"    CDF_DOUBLE   { 0.0 }
    "VALIDMAX"    CDF_DOUBLE   { 4.29497e+09 }
    "LABLAXIS"    CDF_CHAR     { "Proton spectral parameter" }
    "UNITS"       CDF_CHAR     { "-" }
```

```
    "FILLVAL"     CDF_DOUBLE   { -1.0e+31 }
    "VAR_TYPE"    CDF_CHAR     { "data" }
    "DICT_KEY"    CDF_CHAR     { "particle_flux>differential" }
    "SCALETYP"    CDF_CHAR     { "linear" }
    "CATDESC"     CDF_CHAR     { "Fitted proton spectral parameters, " -
                                 "POWER LAW, f=norm*(E/E0)^index        " -
                                 "                                      " -
                                 "    " }
    "DEPEND_0"    CDF_CHAR     { "EPOCH" }
    "DEPEND_1"    CDF_CHAR     { "label_PROTPARAM" } .

  ! RV values were not requested.


! Variable          Data      Number                 Record   Dimension
! Name              Type     Elements  Dims  Sizes  Variance  Variances
! --------          ----     --------  ----  -----  --------  ---------

  "PROTPARERR"    CDF_DOUBLE     1       1     3        T         T

  ! Attribute       Data
  ! Name            Type       Value
  ! --------        ----       -----

    "FIELDNAME"   CDF_CHAR     { "Proton spectral parameter errors" }
    "VALIDMIN"    CDF_DOUBLE   { 0.0 }
    "VALIDMAX"    CDF_DOUBLE   { 4.29497e+09 }
    "LABLAXIS"    CDF_CHAR     { "Proton spectral parameter error" }
    "UNITS"       CDF_CHAR     { "-" }
    "FILLVAL"     CDF_DOUBLE   { -1.0e+31 }
    "VAR_TYPE"    CDF_CHAR     { "data" }
    "DICT_KEY"    CDF_CHAR     { "particle_flux>differential" }
    "SCALETYP"    CDF_CHAR     { "linear" }
    "CATDESC"     CDF_CHAR     { "Error estimate of fitted proton " -
                                 "spectral parameters" }
    "DEPEND_0"    CDF_CHAR     { "EPOCH" }
    "DEPEND_1"    CDF_CHAR     { "label_PROTPARAM" } .

  ! RV values were not requested.


! Variable          Data      Number                 Record   Dimension
! Name              Type     Elements  Dims  Sizes  Variance  Variances
```

```
! --------          ----     --------  ----  -----  --------  ---------

  "label_ELECPARAM"
                 CDF_CHAR      40       1     3        F         T

  ! Attribute       Data
  ! Name            Type       Value
  ! --------        ----       -----

    "FIELDNAME"   CDF_CHAR     { "Electron spectral parameter label" }
    "VAR_TYPE"    CDF_CHAR     { "metadata" }
    "CATDESC"     CDF_CHAR     { "Electron spectral parameter label" } .

  ! NRV values follow...

    [1] = { "EXP, E0                                " }
    [2] = { "EXP, norm                              " }
    [3] = { "EXP, index                             " }


! Variable          Data      Number                 Record   Dimension
! Name              Type     Elements  Dims  Sizes  Variance  Variances
! --------          ----     --------  ----  -----  --------  ---------

  "ELECPARAM"     CDF_DOUBLE     1       1     3        T         T

  ! Attribute       Data
  ! Name            Type       Value
  ! --------        ----       -----

    "FIELDNAME"   CDF_CHAR     { "Electron spectral parameters, " -
                                 "EXPONENTIAL, f=norm*exp(index*(E-E0)) " -
                                 "                                      " -
                                 "    " }
    "VALIDMIN"    CDF_DOUBLE   { 0.0 }
    "VALIDMAX"    CDF_DOUBLE   { 4.29497e+09 }
    "LABLAXIS"    CDF_CHAR     { "Electron spectral parameter" }
    "UNITS"       CDF_CHAR     { "-" }
    "FILLVAL"     CDF_DOUBLE   { -1.0e+31 }
    "VAR_TYPE"    CDF_CHAR     { "data" }
    "DICT_KEY"    CDF_CHAR     { "particle_flux>differential" }
    "SCALETYP"    CDF_CHAR     { "linear" }
    "CATDESC"     CDF_CHAR     { "Fitted electron spectral parameters, " -
```

```
                                  "EXPONENTIAL, f=norm*exp(index*(E-E0)) " -
                                  "                                      " -
                                  "    " }
    "DEPEND_0"    CDF_CHAR     { "EPOCH" }
    "DEPEND_1"    CDF_CHAR     { "label_ELECPARAM" } .

  ! RV values were not requested.


! Variable          Data      Number                 Record   Dimension
! Name              Type     Elements  Dims  Sizes  Variance  Variances
! --------          ----     --------  ----  -----  --------  ---------

  "ELECPARERR"    CDF_DOUBLE     1       1     3        T         T

  ! Attribute       Data
  ! Name            Type       Value
  ! --------        ----       -----

    "FIELDNAME"   CDF_CHAR     { "Electron spectral parameter errors" }
    "VALIDMIN"    CDF_DOUBLE   { 0.0 }
    "VALIDMAX"    CDF_DOUBLE   { 4.29497e+09 }
    "LABLAXIS"    CDF_CHAR     { "Electron spectral parameter error" }
    "UNITS"       CDF_CHAR     { "-" }
    "FILLVAL"     CDF_DOUBLE   { -1.0e+31 }
    "VAR_TYPE"    CDF_CHAR     { "data" }
    "DICT_KEY"    CDF_CHAR     { "particle_flux>differential" }
    "SCALETYP"    CDF_CHAR     { "linear" }
    "CATDESC"     CDF_CHAR     { "Errors estimate of fitted electron " -
                                 "spectral parameters" }
    "DEPEND_0"    CDF_CHAR     { "EPOCH" }
    "DEPEND_1"    CDF_CHAR     { "label_ELECPARAM" } .

  ! RV values were not requested.


#end
```

B.3 CDF settings file (`.set`)

The settings file is a text file that contains what variables should be exported and the order of the variables. A default settings file can be created using the CDFExport program on a specific CDF file. The program must be run interactively as

```
>> cdfexport <CDF file>
```

where `<CDF file>` is a CDF file name without the `.cdf` extension. There are now two alternatives to edit the settings file, either using the CDFExport program or using a text editor.

Using the CDFExport program all record variant variables, except `<Record>` and `<Indices>`, are selected to be outputted (*Output* column set to *YES*). The other variables are selected to not to be outputted. Then from the *OptionMenu* (`CTRL-B`) *Use filters* is set to *no* and *Horizontal listing* is chosen. Back to the main menu the *ActionMenu* (`CTRL-A`) is opened from which *Save settings...* is selected.

Alternatively, the *Save settings...* may be chosen directly to produce a default settings file which is then manually edited. Again, it should be noted that only the record variant variables should be exported (`YES` after item name), except the `Record` and `Indices` (`no` after item name). The `USE_FILTERS` is set to `no` and the `ORIENTATION` is set to `horizontal`.

Running one of the above on the e.g. a Rosetta SREM CDF file produces the settings file shown below (long lines have been truncated).

```
CDF V3.3.0
ITEM=Record,no,,,no,6,
ITEM=Indices,no,,,no,11,
ITEM=Variable,"EPOCH",YES,[6.3169113600000e+13],[6.3429523200000e+13],...
ITEM=Variable,"label_COUNTERS",no,,,no,,Unknown,,8,None,None,0,
ITEM=Variable,"COUNTRATE",YES,[0.0],[4.29497e+09],no,[-1.0e+31],n/a,,...
ITEM=Variable,"label_ORBIT",no,,,no,,Unknown,,8,None,None,0,
ITEM=Variable,"ORBIT",YES,[0.0],[4.29497e+09],no,[-1.0e+31],n/a,,12,...
ITEM=Variable,"FITQUAL",YES,[1],[3],no,[1],Unknown,,8,None,None,0,
ITEM=Variable,"label_PROTPARAM",no,,,no,,Unknown,,42,None,None,0,
ITEM=Variable,"PROTPARAM",YES,[0.0],[4.29497e+09],no,[-1.0e+31],n/a,,...
ITEM=Variable,"PROTPARERR",YES,[0.0],[4.29497e+09],no,[-1.0e+31],n/a,,...
ITEM=Variable,"label_ELECPARAM",no,,,no,,Unknown,,42,None,None,0,
ITEM=Variable,"ELECPARAM",YES,[0.0],[4.29497e+09],no,[-1.0e+31],n/a,,...
ITEM=Variable,"ELECPARERR",YES,[0.0],[4.29497e+09],no,[-1.0e+31],n/a,,...
USE_FILTERS=no
USE_FILLS=YES
CDF_FORMAT=single
CDF_ENCODING=NETWORK
EPOCH_STYLE=standard
ORIENTATION=horizontal
MAJORITY=input
SHOW_FILTERED=no
```

```
SPACING=1
DELETE_EXISTING=no
PREALLOCATE=YES
```

B.4 Text parsing scripts

The generic and custom parser scripts are placed under `$ODI_HOME/parsers/functions`. We first describe the generic script and then show how scripts are added for custom parsing.

The generic script is named `generic.php` and contains two functions:

```
$dfiles = dirscan_generic( $basedir , $filepattern )
$result = parse_generic( $rname , $fid , $sk_file )
```

The `dirscan_generic` function does a recursive directory listing of the files in the directory `$ODI_RAWDATA/$basedir` and returns the array `$dfiles` that contains all files that match the file pattern given by `$filepattern`.

The `parse_generic` function parses the individual raw data files and exports the data into a temporary text file called `load.tmp`. The generic function only works for CDF (`.cdf`) files and gzipped CDF (`.cdf.gz`) files. The raw data file that shall be parsed is given by the string `$rname`. The next argument (`$fid`) is the file id number from the `dataset_file` table. The third argument (`sk_file`) is the skeleton file name.

When the generic parsing functions should be used it is indicated by the `generic` keyword in the eighths column of the `datasets.txt` file.

When the raw dataset file is not a CDF file then custom parsing need to be written. Several examples are found in `$ODI_HOME/parsers/functions/`. The name of the custom parsing script `<name>` is given in the `datasets.txt` file. The the following steps need to be completed:

1. Add a file `<name>.php` under `$ODI_HOME/parsers/functions`.

2. Write a function named `parse_<name>` in the `<name>.php` file. It must take two arguments: `$rname` and `$fid`. The string `$rname` is the name of the raw data file to be ingested. The integer `$fid` is the file id which is the last column in the `dataset_*` table.

3. If a special directory listing function is needed it should also be placed in `<name>.php` and called `dirscan_<name>`. The function must return an array of arrays, where each array contains the file name, file date, and file path. The function takes two arguments: the base directory (`$basedir`) and the file matching pattern (`filepattern`).

The arguments that are supplied to the functions are automatically created by the `populate.sh` script.

C Listing of `cron_jobs.txt`

```
0,10,20,30,40,50 * * * * /home/odi/odi/live/ace_swepam_download.php \
 > /dev/null
0,10,20,30,40,50 * * * * /home/odi/odi/live/ace_mag_download.php \
 > /dev/null
```

D Datasets included in ODI

Table 1: The table lists all data sets included in ODI. The `dataset_` prefix in the ODI Name is not shown.

ODI Name	SEDAT Name	Description
ace_sis	ACE_SIS	ACE-SIS data
ampte_uks	AMPTE	AMPTE UKS electron data
azur	AZUR	AZUR Proton/Alpha particle telescope data
crres_mea	CRRES	CRRES/MEA data
equator_s_aux	EQUATOR_S_AUX	Equator-S AUX Dataset
equator_s_epi	EQUATOR_S_EPI	Equator-S EPI Dataset
equator_s_mam	EQUATOR_S_MAM	Equator-S MAM Dataset
gioveb_srem_pacc	GIOVEB_SREM_PACC	GIOVE-B/SREM PACC Data
goes_sem_a05_5	SPIDR_GOES_A05_5	SPIDR GOES-5 A dataset 5 Minute resolution
goes_sem_a06_5	SPIDR_GOES_A06_5	SPIDR GOES-6 A dataset 5 Minute resolution
goes_sem_a07_5	SPIDR_GOES_A07_5	SPIDR GOES-7 A dataset 5 Minute resolution
goes_sem_a08_5	SPIDR_GOES_A08_5	SPIDR GOES-8 A dataset 5 Minute resolution
goes_sem_a09_5	SPIDR_GOES_A09_5	SPIDR GOES-9 A dataset 5 Minute resolution
goes_sem_a10_5	SPIDR_GOES_A10_5	SPIDR GOES-10 A dataset 5 Minute resolution
goes_sem_a11_5	SPIDR_GOES_A11_5	SPIDR GOES-11 A dataset 5 Minute resolution
goes_sem_a12_5	SPIDR_GOES_A12_5	SPIDR GOES-12 A dataset 5 Minute resolution
goes_sem_g05_1	SPIDR_GOES_G05_1	SPIDR GOES-5 G dataset 1 Minute resolution
goes_sem_g06_1	SPIDR_GOES_G06_1	SPIDR GOES-6 G dataset 1 Minute resolution
goes_sem_g07_1	SPIDR_GOES_G07_1	SPIDR GOES-7 G dataset 1 Minute resolution
goes_sem_g08_1	SPIDR_GOES_G08_1	SPIDR GOES-8 G dataset 1 Minute resolution
goes_sem_g09_1	SPIDR_GOES_G09_1	SPIDR GOES-9 G dataset 1 Minute resolution

Table 1: (continued)

ODI Name	SEDAT Name	Description
goes_sem_g10_1	SPIDR_GOES_G10_1	SPIDR GOES-10 G dataset 1 Minute resolution
goes_sem_g11_1	SPIDR_GOES_G11_1	SPIDR GOES-11 G dataset 1 Minute resolution
goes_sem_g12_1	SPIDR_GOES_G12_1	SPIDR GOES-12 G dataset 1 Minute resolution
goes_sem_h06_5	SPIDR_GOES_H06_5	SPIDR GOES-6 H dataset 5 Minute resolution
goes_sem_h07_5	SPIDR_GOES_H07_5	SPIDR GOES-7 H dataset 5 Minute resolution
goes_sem_h08_5	SPIDR_GOES_H08_5	SPIDR GOES-8 H dataset 5 Minute resolution
goes_sem_h09_5	SPIDR_GOES_H09_5	SPIDR GOES-9 H dataset 5 Minute resolution
goes_sem_h10_5	SPIDR_GOES_H10_5	SPIDR GOES-10 H dataset 5 Minute resolution
goes_sem_h11_5	SPIDR_GOES_H11_5	SPIDR GOES-11 H dataset 5 Minute resolution
goes_sem_h12_5	SPIDR_GOES_H12_5	SPIDR GOES-12 H dataset 5 Minute resolution
goes_sem_i05_5	SPIDR_GOES_I05_5	SPIDR GOES-5 I dataset 5 Minute resolution
goes_sem_i06_5	SPIDR_GOES_I06_5	SPIDR GOES-6 I dataset 5 Minute resolution
goes_sem_i07_5	SPIDR_GOES_I07_5	SPIDR GOES-7 I dataset 5 Minute resolution
goes_sem_i08_5	SPIDR_GOES_I08_5	SPIDR GOES-8 I dataset 5 Minute resolution
goes_sem_i09_5	SPIDR_GOES_I09_5	SPIDR GOES-9 I dataset 5 Minute resolution
goes_sem_i10_5	SPIDR_GOES_I10_5	SPIDR GOES-10 I dataset 5 Minute resolution
goes_sem_i11_5	SPIDR_GOES_I11_5	SPIDR GOES-11 I dataset 5 Minute resolution
goes_sem_i12_5	SPIDR_GOES_I12_5	SPIDR GOES-12 I dataset 5 Minute resolution
goes_mag_06	SPIDR_GOES06_MAG	SPIDR GOES-6 MAG dataset 5 Minute resolution
goes_mag_07	SPIDR_GOES07_MAG	SPIDR GOES-7 MAG dataset 5 Minute resolution

Table 1: (continued)

ODI Name	SEDAT Name	Description
goes_mag_08	SPIDR_GOES08_MAG	SPIDR GOES-8 MAG dataset 5 Minute resolution
goes_mag_09	SPIDR_GOES09_MAG	SPIDR GOES-9 MAG dataset 5 Minute resolution
goes_mag_10	SPIDR_GOES10_MAG	SPIDR GOES-10 MAG dataset 5 Minute resolution
goes_mag_11	SPIDR_GOES11_MAG	SPIDR GOES-11 MAG dataset 5 Minute resolution
goes_mag_12	SPIDR_GOES12_MAG	SPIDR GOES-12 MAG dataset 5 Minute resolution
goes_z05_5	SPIDR_GOES_Z05_5	SPIDR GOES-5 Z dataset 5 Minute resolution
goes_z06_5	SPIDR_GOES_Z06_5	SPIDR GOES-6 Z dataset 5 Minute resolution
goes_z07_5	SPIDR_GOES_Z07_5	SPIDR GOES-7 Z dataset 5 Minute resolution
goes_z08_5	SPIDR_GOES_Z08_5	SPIDR GOES-8 Z dataset 5 Minute resolution
goes_z09_5	SPIDR_GOES_Z09_5	SPIDR GOES-9 Z dataset 5 Minute resolution
goes_z10_5	SPIDR_GOES_Z10_5	SPIDR GOES-10 Z dataset 5 Minute resolution
goes_z11_5	SPIDR_GOES_Z11_5	SPIDR GOES-11 Z dataset 5 Minute resolution
goes_z12_5	SPIDR_GOES_Z12_5	SPIDR GOES-12 Z dataset 5 Minute resolution
helios_a_e6	HELIOS_A_E6	HELIOS-A E6 Data
helios_a_e7	HELIOS_A_E7	HELIOS-A E7 Data
helios_b_e6	HELIOS_B_E6	HELIOS-B E6 Data
helios_b_e7	HELIOS_B_E7	HELIOS-B E7 Data
imp8_cpme_e_330s	IMP8_CPME_E_330S	IMP-8 CPME e data
imp8_cpme_h_330s	IMP8_CPME_H_330S	IMP-8 CPME H data
imp8_cpme_he_330s	IMP8_CPME_HE_330S	IMP-8 CPME He data
imp8_cpme_mh_330s	IMP8_CPME_MH_330S	IMP-8 CPME heavy ion data
imp8_crnc_phint	IMP8_CRNC_PHINT	IMP-8 CRNC (U. Chicago) PHINT Data Tape
imp8_gme	IMP8_GME	IMP-8 GME (GSFC Instrument)
index_dst	DST	DST index 1957-1997
index_kpap_1d	AP	Ap global geomagnetic index

Table 1: (continued)

ODI Name	SEDAT Name	Description
index_kpap_3h	KPAP	Kp and Ap global geomagnetic index
index_omni2	NSSDC_OMNI2	NSSDC OMNI-2 Dataset
index_ssn_1m	SSN	Monthly sunspot numbers
integral_irem	INTEGRAL_IREM_PACC	INTEGRAL/IREM PACC Data
isee1_hi	ISEE1_HI	ISEE1 high resolution data
isee1_lo	ISEE1_LO	ISEE1 low resolution data
isee1_mepi	ISEE1_MEPI	ISEE1 MEPI data
isee2	ISEE2	ISEE2 data
meteosat_anomalies	METEOSAT_ANOMALIES	METEOSAT anomalies
meteosat_hr	METEOSAT_HR	METEOSAT high resolution data
meteosat_lr	METEOSAT_LR	METEOSAT low resolution data
metop_02	METOP_02	METOP-02 Space Environment Monitor
mir_a	MIR_A	MIR Raw data
mir_b	MIR_B	MIR reduced and supplementary data
ns41_bdd2r	GPS_NS41	GPS NavStar41 - Burst Dosimeter Detector IIR
poes_n15	POES_N15	NOAA POES N15 Space Environment Monitor
poes_n16	POES_N16	NOAA POES N16 Space Environment Monitor
poes_n17	POES_N17	NOAA POES N17 Space Environment Monitor
poes_n18	POES_N18	NOAA POES N18 Space Environment Monitor
proba1_srem_pacc	PROBA1_SREM_PACC	PROBA-1 SREM PACC Data
rosetta_srem_pacc	ROSETTA_SREM	Rosetta SREM Radiation Monitor
sac_c	SAC_C	SAC-C data
sampex_pet	SAMPEX	SAMPEX PET data
soho_erne_a	SOHO_ERNE_A	SOHO-ERNE Alpha Data
soho_erne_p	SOHO_ERNE_P	SOHO-ERNE Proton Data
strv1b_a	STRV1B_A	STRV1B Raw data
strv1b_b	STRV1B_B	STRV1B reduced and supplementary data

Table 1: (continued)

ODI Name	**SEDAT Name**	**Description**
swpc_ace_1m		One minute resolution ACE SWEPAM and MAG data.
uars_pem	UARS	UARS Particle Environment Monitor data
xmm_rm	XMM_RM	XMM Radiation Monitor