

# DES (Data Exchange Server) documentation

The DES software, implemented entirely by authors at the [Center for Sensing Solutions](#) at [Eurac research](#), is a standalone application to enable the concurrent execution of tasks and data transfers from and to different systems by means of a set of XML Processor configuration files. It has been primarily developed to fit the requirements of the EURAC receiving station processing chain and Data Infrastructure here at EURAC Research where big amounts of data need to be processed and transferred from system to system or delivered to some end users. The DES implements a concurrent, multi-threaded mechanism to transfer different kind of data in PUSH and GET mode using standard protocols like HTTP, SFTP, FTP, SSH. However, the DES may be seen as a general, multi-tasking application that can execute concurrently configurable tasks or jobs implemented in dedicated plug-ins or scripts. The basic concept behind the DES is that data is organized in a hierarchical structure that reflects a \$AOI/\$PROCESSOR scheme and that Events or Tasks are triggered by files filtered with regular expressions.

[Center for Sensing Solutions \(Eurac research\)](#)

Administration: [sensing.solutions@eurac.edu](mailto:sensing.solutions@eurac.edu)

Technical support: [support.css@eurac.edu](mailto:support.css@eurac.edu)

## Processor definition example:

This example Processor definition in xml format defines and implements Data Processing, Metadata Generation and Data Feeding into an Spatial Data Infrastructure

[EURAC\\_RADBEAM\\_P.processor\\_config.xml](#)

Latest release: DES 1.7

## Version history

Version	Date	Changes	Authors
V0.1	23.06.2009	Implementation	<a href="mailto:armin.costa@eurac.edu">armin.costa@eurac.edu</a>
V1.0	01.05.2010	Implementation	<a href="mailto:armin.costa@eurac.edu">armin.costa@eurac.edu</a>
V1.5	01.09.2015	Implementation	<a href="mailto:armin.costa@eurac.edu">armin.costa@eurac.edu</a>
V1.6	24.02.2017	Documentation update	<a href="mailto:armin.costa@eurac.edu">armin.costa@eurac.edu</a> , <a href="mailto:bartolomeo.ventura@eurac.edu">bartolomeo.ventura@eurac.edu</a>
V1.7	29.03.2019	Implementation	<a href="mailto:armin.costa@eurac.edu">armin.costa@eurac.edu</a>
V1.7	18.04.2019	Documentation migration	<a href="mailto:armin.costa@eurac.edu">armin.costa@eurac.edu</a>
V1.7	23.04.2019	Documentation update	<a href="mailto:armin.costa@eurac.edu">armin.costa@eurac.edu</a>
V1.7	24.04.2019	Documentation update	<a href="mailto:armin.costa@eurac.edu">armin.costa@eurac.edu</a>

## Features

The DES application currently implements the following features

## Concurrent Distribution of Data

- supported protocols: SFTP, FTP
- recursive upload of directories
- Notification of data delivery via e-mail

```
<AOI name="aoi" active="yes">
  <Distribution type="sftp" active="no" priority="NORM">
    <consumer name="dummy" active="no">
      <ProductType>SAMPLE.aoi.*.AA.txt</ProductType>
      <EMail>support.rs@eurac.edu</EMail>
      <NotifyEmail>>false</NotifyEmail>
    </consumer>
  </Distribution>
  <Distribution type="ftp" active="no" priority="NORM">
    <consumer name="CoA" active="yes">
      <ProductType>*.txt</ProductType>
      <EMail>support.rs@eurac.edu</EMail>
      <NotifyEmail>>false</NotifyEmail>
      <Recursive>>true</Recursive>
    </consumer>
  </Distribution>
</AOI>
```

## Concurrent Download of Data

- supported protocols: SFTP, FTP
- Notification of data received via e-mail

```
<AOI name="ReceivingStation" active="yes">
  <Download type="sftp" active="yes" priority="NORM">
    <consumer name="rsuser@eomount.eurac.edu" active="yes">
      <ProductType>NPP.*.Eurac.dat</ProductType>
      <DestinationPath>./nppdata</DestinationPath>
      <EMail>support.rs@eurac.edu</EMail>
      <NotifyEmail>true</NotifyEmail>
    </consumer>
  </Download>
</AOI>
```

## Concurrent Execution of Tasks

- Plugins in form of java classes, shell scripts/commands, perl scripts
- Task execution is driven and triggered by filtered FileType's
- On the fly java code compilation is supported

```
<AOI name="aoi" active="yes">
  <Task name="renameFiles" type="class" active="yes" priority="NORM">
    <Command>class</Command>
    <FileType>SAMPLE.*.AA.txt</FileType>
    <Parameter>edu.eurac.distributionserver.tasks.generic.FileRename
  </Parameter>
    <FilePrefix>RENAME</FilePrefix>
  </Task>
  <Task name="testCmd" type="cmd" active="no" priority="NORM">
    <EMail>support.rs@eurac.edu</EMail>
    <NotifyEmail>false</NotifyEmail>
    <Command>ping</Command>
    <Parameter>127.0.0.1</Parameter>
  </Task>
  <Task name="testPerl" type="sh" active="no" priority="NORM">
    <EMail>support.rs@eurac.edu</EMail>
    <NotifyEmail>false</NotifyEmail>
    <Command>perl</Command>
    <Parameter>./test_data/scripts/testPerl.pl</Parameter>
    <Parameter>$CWD</Parameter>
  </Task>
  <Task name="moveToStorage" type="shell" active="no" priority="NORM">
    <RunOnTrigger>/tmp/RunMoveToStorage</RunOnTrigger>
    <Command>sh</Command>
    <Parameter>./test_data/scripts/moveToStorage.sh</Parameter>
    <Parameter>/raid0/NISDS2/raid/pub/gsfcddata/npp/viirs/level2/*
    </Parameter>
    <Parameter>/raid0/level2STORAGE/</Parameter>
  </Task>
</AOI>
```

## Concurrent Parallel/Serial Execution of Task Groups

- Task groups can be serial or parallel

```
<AOI name="aoi" active="yes">
  <TaskGroup name="testGroup" type="parallel" active="yes"
    priority="NORM">
    <Task name="testCmd1" type="cmd" active="yes" priority="NORM">
      <EMail>support.rs@eurac.edu</EMail>
      <NotifyEmail>false</NotifyEmail>
      <Command>ping</Command>
      <Parameter>127.0.0.1</Parameter>
    </Task>
    <Task name="testCmd2" type="cmd" active="yes" priority="NORM">
      <EMail>support.rs@eurac.edu</EMail>
      <NotifyEmail>false</NotifyEmail>
      <Command>ping</Command>
      <Parameter>127.0.0.1</Parameter>
    </Task>
  </TaskGroup>
  <TaskGroup name="testGroup2" type="serial" active="no"
    priority="NORM">
    <Task name="testCmd1" type="cmd" active="yes" priority="NORM">
      <EMail>support.rs@eurac.edu</EMail>
      <NotifyEmail>false</NotifyEmail>
      <Command>ping</Command>
      <Parameter>127.0.0.1</Parameter>
    </Task>
    <Task name="testCmd2" type="cmd" active="yes" priority="NORM">
      <EMail>support.rs@eurac.edu</EMail>
```

```

    <NotifyEmail>false</NotifyEmail>
    <Command>ping</Command>
    <Parameter>127.0.0.1</Parameter>
  </Task>
</TaskGroup>
</AOI>

```

## Concurrent Cleaning of data

- implements all switches available in linux find command
- On Windows OS a CyWIN installation is required

```

<AOI name="cleanSATWorkspaces" active="yes">
  <DataCleaner name="01_Data" type="filesystem" active="yes"
    priority="NORM">
    <RunOnTrigger>/tmp/RunSATWS_cleanup</RunOnTrigger>
    <Email>support.rs@eurac.edu</Email>
    <NotifyEmail>yes</NotifyEmail>
    <IncludePath>/mnt/SATWS/*01_Data/*</IncludePath>
    <ExcludeFileType>*.pro</ExcludeFileType>
    <ExcludeFileType>*.java</ExcludeFileType>
    <ExcludeFileType>*.f</ExcludeFileType>
    <ExcludeFileType>*.m</ExcludeFileType>
    <ExcludeFileType>*.r</ExcludeFileType>
    <ExcludeFileType>*.c</ExcludeFileType>
    <ExcludeFileType>*.h</ExcludeFileType>
    <ExcludeFileType>*.doc</ExcludeFileType>
    <Size>+1k</Size>
    <LastAccess>+60</LastAccess>
  </DataCleaner>
</AOI>

```

## Configuration

### Startup configuration

The DES startup configuration contains initialization variables that are loaded at DES startup or restart

#### Parameters

[System]

RDS\_ID DES identifier (ex. DES\_yourID@yourdomain.org)

RDS\_HOME HOME directory of DES

CONFIG\_FILE\_PATH Path where processor configuration files are located ( \$PROCESSOR.processor\_config.xml)

LOGS\_PATH STOUT print to stdout | LOG4J : use log4j2| Path of the log file dir (ending with / , ex: ./testdata/logs/ ) directory will be created

BASE\_PATH Basepath at which logical directory tree starts: \$BASE\_PATH/\$AoiName/\$ProcessorName

STAMPS\_PATH Directory for Stamp files

COMPILECODE\_PATH Temporary directory where class files are compiled on the fly. Content will be deleted at every restart

MAX\_PROCESSOR\_THREADS Max processor Threads

MAX\_AOI\_THREADS Max Threads per AOI

MAX\_AOI\_TASKGROUP\_THREADS Max Threads per TaskGroup

AOI\_THREAD\_SLEEP Time interval for AOI Thread

MAX\_NR\_FILTER\_FILES Max number of files filtered in each iteration

DEBUG Write to log files

STDOUT\_STDERR Print Tasks output streams to STDOUT/STDERR)

[DB]

USE\_STATUS\_DB Enable Task status logging in DB

DB\_SERVER = Database server

DB\_DATABASE = Database name

DB\_PORT = Database port

DB\_USER = Database user

DB\_PWD Database password

[Mail]

FROM\_EMAIL e-mail for delivery

SMTP = URL for SMTP

SMTP\_PORT Port for SMTP

[Plugin]

CONFIG\_PLUGIN Configuration file for Plugins. Java class variable: DistributionServerConfig.CONFIG\_PLUGIN

### Sample configuration

```

[System]
RDS_ID=DES_YOURIDC@yourdomain.org
RDS_HOME=/DES
CONFIG_FILE_PATH=./test_data/config/
LOGS_PATH=LOG4J
BASE_PATH=./test_data/data/
STAMPS_PATH=./test_data/stamps
COMPILECODE_PATH=./test_data/tmp
MAX_PROCESSOR_THREADS=50
MAX_AOI_THREADS=50
MAX_AOI_TASKGROUP_THREADS=50
AOI_THREAD_SLEEP=10000
MAX_NR_FILTER_FILES=5
DEBUG=true
STDOUT_STDERR=false
[DB]
USE_STATUS_DB=false
DB_SERVER=server.yourdomain
DB_DATABASE=des
DB_PORT=5432
DB_USER=user
DB_PWD=pwd
[Mail]
FROM_EMAIL=des@yourdomain.org
SMTP=mailsubmit.yourdomain.org
SMTP_PORT=25
[Plugin]
CONFIG_PLUGIN=./plugins/plugin.ini

```

## Processors

The DES executes concurrently a given set of Processor definition files that are defined in XML format. A Processor configuraiton file ( \*.processor\_config.xml ) is the main executable entity for the DES.

- There can be 1 - n Processor configuration files running on a given DES instance.
- Each processor configuration file can have 1 - n AOI (Areas Of Interest).
- Each AOI can have 1 - n functional entities (Distribution, Download, DataCleaner, Task, TaskGroup)
- The same xml format is used also by EGPF (Eurac Generic Processor Framework)

A typical DES \*Processor" may have the following layout:

```

<EURAC_GENERIC_P>
  <Processing>
    <AOI name="myAOI" active="yes">

      <!-- Download data -->
      <Download type="sftp" active="yes" priority="NORM">
        ...
      </Download>

      <!-- Process data -->
      <TaskGroup name="processAOI" type="serial" active="yes" priority="NORM">
        <Task name="task1" type="class" active="yes" priority="NORM">
          ...
        </Task>
        <Task name="task2" type="class" active="yes" priority="NORM">
          ...
        </Task>
      </TaskGroup>

      <!-- Distribute data -->
      <Distribution type="sftp" active="yes" priority="NORM">
        ...
      </Distribution>
    </AOI>
  </Processing>
</EURAC_GENERIC_P>

```

The Processor configuration options available are documented here:

[https://gitlab.inf.unibz.it/css-public/des\\_documentation/blob/master/Processors.md](https://gitlab.inf.unibz.it/css-public/des_documentation/blob/master/Processors.md)

## Authentication configuration file

Authentication parameters used in the operations and are stored within the file :

DistributionAuth.xml: ./config/DistributionAuth.xml

Sample entry:

```
<Auth>
  <AuthRef name="testuser@server.yourdomain" active="yes">
    <Host>server.yourdomain</Host>
    <User>testuser</User>
    <Pwd>test</Pwd>
    <Param>nativejava</Param>
  </AuthRef>
</Auth>
```

In the DES processor operation declarations the authentication entry can be references as follows:

```
<consumer name="testuser@server.yourdomain" active="yes">
  <ProductType>*.txt</ProductType>
  <EMail>notifymail@yourdomain</EMail>
  <NotifyEmail>>false</NotifyEmail>
  <Recursive>>true</Recursive>
</Consumer>
```

## Built-in variables

Inside a Processor definition configuration file some build-in parameter can be used to abstract some configurations.

The build-in parameters are replaced at runtime in all and in all custom tags (eg. \$DAY)

\$BASEPATH The base path variable configured in DES.ini

\$AOI The name of the executing AOI

\$PROCESSOR The name of the executing Processor

\$CWD The current working directory, that expands to \$BASEPATH/\$AOI/\$PROCESSOR

\$STAMPS\_PATH The path for Stamp files configured in DES.ini

\$FILE Iff is configured, references the current File being processed ( is deprecated but still valid for backward compatibility)

\$YEAR The current year

\$MONTH The current month

\$DAY The current day

## Start a DES instance

1. Edit the ./config/DistributionServer.ini to fit your environment
2. Create the set of \*.processor\_config.xml files that you want to run and place them in the path CONFIG\_FILE\_PATH as defined in DistributionServer.ini config file
3. Choose a suitable operation mode

print program arguments:

```
java -jar DES.jar -h
```

## Operation modes

- 1) Start a DES instance with default configuration defined in DES.ini and run all Processors (\*.processor\_config.xml files) located in CONFIG\_PATH

```
java -jar DES.jar
```

- 2) Start a DES instance with default configuration defined in DES.ini and run a single Processor

```
java -jar DES.jar -p yourtest.processor_config.xml
```

- 3) Start a DES instance with a different configuration (eg. DES\_test.ini) and run all Processors (\*.processor\_config.xml files) located in CONFIG\_PATH

```
java -jar DES.jar -c ./config/DES_test.ini
```

- 4) Start a DES instance with a different configuration (eg. DES\_test.ini) and run a given Processor

```
java -jar DES.jar -c ./config/DES_test.ini -p yourtest.processor_config.xml
```

## Passing command line startup variables to be referenced in DES Processor

Custom arguments can be passed to the startup command line

```
java -jar DES.jar -v HOSTNAME=des.yourdomain -v IP=192.168.2.2
```

The defined variables can then be referenced in a given Processor definition with the '@' reference (eg. @HOSTNAME)

```
<Task name="testCmd" type="cmd" active="no" priority="NORM">
  <EMail>support.rs@eurac.edu</EMail>
  <NotifyEmail>false</NotifyEmail>
  <Command>ping</Command>
  <Parameter>@HOSTNAME</Parameter>
</Task>
```

## Conventions

### Logical directory hierarchy

The DES starts its operations (eg. filtering) considering a logical hierarchy of directories starting from a given BASE\_PATH that reflects the configuration within a given processor configuration file.

The directories, if not already present, are created automatically with the following structure:

```
$BASE_PATH/$AOI/$PROCESSOR
```

### FileType filtering

The files to be handled by a given Task or that are used for a given Task triggering, are filtered by regular expressions defined in the FileType tag of the Task

Files are detected by the FileType filtering IFF: - No stamp file exists (eg. yourfile.txt.stamp) in the path defined by STAMPS\_PATH/\$PROCESSOR - An equivalent .md5 file exists in the logical directory hierarchy (ex.yourfile.txt.md5), unless the tag MD5Filter is set explicitly to value false

Stamp files are used to track that a file has already been processed by a given Task.

IFF a given Task returns either EXIT\_SUCESS (0) or EXIT\_WARNING (1), the DES will create a corresponding stamp file for the file processed.

If a given Task does not comply to the EXIT\_CODES defined above, the Task implementation, upon completion, should therefore either delete the input file and its equivalent md5, and if necessary, to avoid a reprocessing of a given file, create a corresponding stamp file in STAMPS\_PATH/\$PROCESSOR directory

### Stamp files

A Stamp file is an empty file that is created iff a given Task has been completed for a given input file filtered by a FileType expression. In presence of a stamp file a subsequent FileType filtering will not detect the file that has an equivalent stamp file.

Stamp files are stored in the directory specified by STAMPS\_PATH/\$PROCESSOR

Stamp files have the following naming convention:

```
[PUSH | GET | TASK]_[Consumer name | Task name]_[Procesor name]_[AOI name]_[File name].stamp
```

Example:

```
TASK_createMap_EURAC_RADBEAM_P_southtyrol_TREURAC_RADBEAM_P.southtyrol.demST25m.year.2017.2018-06-19T16:45:00.Eurac.01.00.trigger.stam
```

### Exit Codes

The Task entities registered within a Processor definition should use the following ExitCodes when returning from execution

```
EXIT_NONE = -999; // No files are available in the DES task pipeline
EXIT_SUCESS = 0;
EXIT_WARNING = 1; //1 - 127
EXIT_ERROR = -1;
```

```
EXIT_ERROR_SHELL = 255; // Shell scripts that exit -1 are converted to 255
EXIT_NOPROC = -666; // Task was executed but, either did not any job at all, OR the Task plugin did skip the files
EXIT_MISSING_PARAM = 2; // Task did not receive all or valid parameters
```

# Processor configuration

## AOI

The AOI entity represents a logical hierarchy in a processor definition that is mapped also in the DES folder structure.

Each processor can have 1 - n AOI, also with the same name if required.

```
<EURAC_GENERIC_P>
  <Processing>
    <AOI name="yourAOI" active="yes">
      ...
    </AOI>
  </Processing>
</EURAC_GENERIC_P>
```

### Attributes

name = : Name of the AOI

active = yes | no : Enable or disable AOI

[sleepFactor] = positive int : Factor that multiplies the AOI\_THREAD\_SLEEP, namely the sleeping interval for the threads inside AOIs. This enables to have AOIs where Threads have bigger sleeping intervals

### Tags

[[Distribution] [Download] [Task] [TaskGroup] [DataCleaner]] (1 - n)

## Distribution

The Distribution entity provides file transfer functionality for uploading data

```
<Distribution type="ftp" active="no" priority="NORM">
  <consumer>
  </consumer>
</Distribution>
```

### Attributes

type = ftp | sftp : Transfer protocol

active = yes | no : Enable or disable Distribution

priority = MIN | NORM | MAX : Thread priority

### Tags

Consumer (1 - n)

## Download

The Download entity provides file transfer functionality for downloading data

```
<Download type="sftp" active="yes" priority="NORM">
  <consumer>
  </consumer>
</Download>
```

### Attributes

type = sftp : Transfer protocol

active = yes | no : Enable or disable Distribution

priority = MIN | NORM | MAX : Thread priority

Tags

Consumer (1 - n)

Consumer

The Consumer entity represents the configuration for Distribution and Download entities

```
<consumer name="user@host.yourdomain" active="yes">
  <ProductType>*.txt</ProductType>
  <EMail>mail@yourdomain</EMail>
  <NotifyEmail>>false</NotifyEmail>
  <Recursive>>true</Recursive>
</consumer>
```

Attributes

name = : Name reference for authentication in DistributionAuth.xml

active = yes | no : Enable or disable Consumer

NOTE: It is worthwhile noting that in this file each name reference should be different from the others. It is strongly suggested to use a naming convention such as user@hostname as indicated in the follow:

```
<AuthRef name="user@host.yourdomain" active="yes">
  <Host>host.yourdomain</Host>
  <User>user</User>
  <Pwd>pass</Pwd>
  <Param></Param>
</AuthRef>
```

Tags

RunOnTrigger = : Absolute path of file that serves as trigger (has priority over StartDate/StopDate and overrides Runnable state). Trigger file will be deleted IFF ExitCode == 0 || ExitCode == 1

StartDate = Date Time to start the Task (YYY-MM-DDTmm:hh:ss)

StopDate = Date Time to stop the Task (YYY-MM-DDTmm:hh:ss)

FileType (1 - n) = : regular expresison for file type filtering (default: .)

MD5Filter = true | false : Filters files defined in FileType/ProductType with check on exiting MD5 file. NOTE: for Downloads this can be set to 'false' whenever the server side file does not have an equivalent md5 file. For Distributions and Tasks it is advisable to have the default 'true' value in order to have a file completion validation, else the dataflow logic needs to guarantee file completion (default: true)

MD5Generation = true | false : If not present, generates the MD5 file. NOTE: must be false in this case in order to have effect

MD5Check = true | false: IFF AND ==true recomputes the md5 checksum and compares to .md5 file

Email (1 -n) = : Email notification recipient

NotifyEmail = true (OnError) | OnSuccess | OnError | OnWarning | false : To enable e-mail notifications on data delivery

RemoteDestinationPath = : Path for directory in destination Host (default: \$AOI/\$PROCESSOR ) [ Deprecated keyword 'DestinationPath' is still valid ]

LocalDestinationPath = : Path for directory local Host (default: \$BASEPATH/\$AOI/\$PROCESSOR)

ProductRename = : File name delivered to remote Host (default: filename as filtered by FileType)

Recursive = true | false : Enable recurive filtering and upload of data in directory hierarchies. Currently only applicable to

DataCleaner

The DataCleaner entity provides functionality for deleting filesystem contents

```
<DataCleaner name="HOME" type="filesystem" active="yes" priority="NORM">
  <EMail>mail@host.yourdomain</EMail>
  <NotifyEmail>yes</NotifyEmail>
  <IncludePath>/raid0/abz01rtest.eurac.edu/AdminCoA/*</IncludePath>
  <ExcludePath>/raid0/abz01rtest.eurac.edu/AdminCoA/doNOTdelete/*</ExcludePath>
```

```
<ExcludeFileType>*.java</ExcludeFileType>
<Size>+1k</Size>
<LastAccess>+1</LastAccess>
</DataCleaner>
```

## Attributes

name = : Name of DataCleaner

type = filesystem : Type of data Cleaner

active = yes | no : Enable or disable DataCleaner

priority = MIN | NORM | MAX : Thread priority

## Tags

RunOnTrigger = : Absolute path of file that serves as trigger (has priority over StartDate/StopDate and overrides Runnable state). Trigger file will be deleted IFF ExitCode == 0 || ExitCode == 1

StartDate = Date Time to start the Task (YYY-MM-DDTmm:hh:ss)

StopDate = Date Time to stop the Task (YYY-MM-DDTmm:hh:ss)

Email (1 -n ) = : Email notification recipient

NotifyEmail = true (OnError) | OnSuccess | OnError | OnWarning | false : To enable e-mail

IncludePath (1 - n) = : Path to be included in the cleaning policy

ExcludePath (1 - n) = : Path to be excluded in the cleaning policy

ExcludeFileType (1 - n) = : Regular expression for file types to be excluded from the cleaning policy

Size = +int ( +1c, +1k, +1M, +1G) : Size in byte, kilobyte, megabyte, gigabyte (default: MIN\_SIZE = "+1000k")

LastAccess = +int : Nr of days (default: LAST\_ACCESS = "30")

LastModified = : +int : Nr of days (default: LAST\_MODIFIED = "+30"; // Days ago from the current date)

Type = f (file) | d (directory) | l (link) : Type of data, only a single value allowed (default: f)

NOTE: Only available on Linux/UNIX compatible environments OR Windows with CyWin

## Task

The Task entity encapsulates a given command or code to be executed concurrently in form of a dedicated execution thread

```
<Task name="testPerl" type="cmd" active="yes" priority="NORM">
  <Email>mail@host.yourdomain</Email>
  <NotifyEmail>false</NotifyEmail>
  <Command>perl</Command>
  <FileType>*.txt</FileType>
  <Parameter>./test_data/scripts/testPerl.pl</Parameter>
  <Parameter>${CWD}</Parameter>
</Task>
```

## Attributes

name = : Name of executing Task

type = mail | class | code | cmd : Execution type

active = yes | no : Enable or disable Task

priority = MIN | NORM | MAX : Thread priority

## Tags

RunOnTrigger = : Absolute path of file that serves as trigger (has priority over StartDate/StopDate and overrides Runnable state). Trigger file will be deleted IFF ExitCode == 0 || ExitCode == 1

StartDate = Date Time to start the Task (YYY-MM-DDTmm:hh:ss)

StopDate = Date Time to stop the Task (YYY-MM-DDTmm:hh:ss)

Email (1 -n ) = : Email notification recipient

NotifyEmail = true (OnError) | OnSuccess | OnError | OnWarning | false : To enable e-mail

Command = class | code | cmd (sh | perl | python | cmd ) : Command to be executed

FileType | ProductType (1 - n) = : regular expresison for file type filtering. If not defined task will be executed without passing a file list

LocalDestinationPath = : Relative Path for directory to list File specified by a given (relative to directory: \$BASEPATH/\$AOI/\$PROCESSOR)

MD5Filter = true | false : Filters files defined in FileType/ProductType with check on exiting MD5 file. NOTE: for Downloads this can be set to 'false' whenever the server side file does not have an equivalent md5 file. For Distributions and Tasks it is advisable to have the default 'true' value in order to have a file completion validation, else the dataflow logic needs to guarantee file completion (default: true)

LOCKFilter = true | false : Filters files defined in FileType/ProductType with check on exiting LOCK file. This is used whenever a given Task should have exclusive access to a file, i.e. for parallelization. The LOCK file is deleted automatically whenever a Tasks returns form the execution of a given file

Parameter (1 - n) = : Parameters to Command

<...> (1 - n) = : Custom Tags that can be added and are passed as hastable (only available to Commands of type class

## TaskGroup

The TaskGroup entity encapsulates a given set of Task entities and provides serial or parallel execution

```
<TaskGroup name="taskGroup1" type="parallel" active="yes" priority="NORM">
  <Task>
    ...
  </Task>
  <Task>
    ...
  </Task>
</TaskGroup>
```

### Attributes

name = : Name of executing Task

type = parallel | serial : Run tasks in parallel or serial mode within this TaskGroup

active = yes | no : Enable or disable the TaskGroup

priority = MIN | NORM | MAX : Threads priority

### Tags

RunOnTrigger = : Absolute path of file that serves as trigger (has priority over StartDate/StopDate and overrides Runnable state). Trigger file will be deleted IFF ExitCode == 0 || ExitCode == 1

StartDate = Date Time to start the Task (YYY-MM-DDTmm:hh:ss)

StopDate = Date Time to stop the Task (YYY-MM-DDTmm:hh:ss)

EEmail (1 -n ) = : Email notification recipient

NotifyEmail = true (OnError) | OnSuccess | OnError | OnWarning | false : To enable e-mail

Task (1 -n)

## Built-in variables

Inside a Processor definition configuration file some build-in parameter can be used to abstract some configurations.

The build-in parameters are replaced at runtime in all and in all custom tags (eg. \$DAY)

\$BASEPATH The base path variable configured in DES.ini

\$AOI The name of the executing AOI

\$PROCESSOR The name of the executing Processor

\$CWD The current working directory, that expands to \$BASEPATH/\$AOI/\$PROCESSOR

\$STAMPS\_PATH The path for Stamp files configured in DES.ini

\$FILE Iff is configured, references the current File being processed ( is deprecated but still valid for backward compatibility)

\$YEAR The current year

\$MONTH The current month

\$DAY The current day

# Task Plugins

The Task entities defined in a Processor definition can be implemented as dedicated native java plugins or as scripts written in Perl, Python or Shell scripting languages.

## Native plugins

Dedicated native java plugins can be developed by implementing the GenericTaskInterface process method defined in the DES library

GenericTaskInterface Interface

```
public interface GenericTaskInterface {

    /**
     *
     * @param basepath Basepath as configured in DistributionServer.ini
     * @param aoi_name AOI name contained in the $processor.processor_config.xml
     * @param processor_name Processor name related to the $processor.processor_config.xml
     * @param entity_name Name of executing entity (Task, Consumer, etc.)
     * @param parameters Parameters as defined by <Parameter>
     * @param args List of arguments of the Task (String or Hashtables, to be checked by the plugin)
     * @param files List of files of the Task IFF <FileType>regex</FileType> is defined, else null
     * @return ExitCodes
     */
    public int process(String basepath, String aoi_name,
        String processor_name, String entity_name, Hashtable parameters, Object[] args, String[] files);

}
```

The implemented plugin should be packaged as a jar file and placed in the DES ./plugins directory where it is automatically loaded at Runtime.

The plugin can be executed in a given Processor Task with the following definition:

Example:

```
<Task name="renameFiles" type="class" active="yes" priority="NORM">
  <Command>class</Command>
  <FileType>SAMPLE.*.AA.txt</FileType>
  <Parameter>edu.eurac.distributionserver.tasks.generic.FileRename</Parameter>
  <FilePrefix>RENAME</FilePrefix>
</Task>
```

The parameters can be passed to the Task plugin with one of the following options:

- By using a custom keyword tag eg. FilePrefix in the Task definition and then accessing the parameter with the Hashtable parameters in the Interface

```
String file_prefix = (String)parameters.get("FilePrefix");
```

- By using the Parameter tag in the Task definition and than referencing the parameter with the Object[] args type in the Interface

```
String classname = (String)args[1];
```

Note: Note: args[0] is the class name

## Script plugins

If a given Task is implemented with a scripting language (Perl, Python, Shell), the script must consider the following input parameters:

```
[BASE_PATH] [AOI] [PROCESSOR] [Parameter_1] [Parameter_N] ... [$FILE*]
```

IFF a FileType filtering is defined for the Task, the last parameter, defined above as \$FILE, is the filename of the 1st file filtered by the Task execution. In this case the script will be executed N times, for each file filtered by the FileType regular expression.

Example:

```
#!/usr/bin/perl -w

# This script is part of the Task plugin collection available for the DES (Data Exchange Server)
# version: 1.0
# usage: testPerlExitCode.pl [base_path] [aoi] [procesor_name] [Sleep] [ExitCodeToReturn]

#main
print "running testPerlExitCode.pl\n";
my $base_path = "null";
my $aoi = "null";
my $processor_name = "null";
my $sleep_sec = 1; # 1 second default
my $exit_code = -1;
print "Parameters: @ARGV\n";
if(scalar(@ARGV) < 4){
    print "missing arguments!\n";
    print "usage: testPerlExitCode.pl [base_path] [aoi] [Sleep] [ExitCode]\n";
    exit -1;
}else{
    $base_path = $ARGV[0];
    $aoi = $ARGV[1];
    $processor_name = $ARGV[2];
    $sleep_sec = $ARGV[3];
    $exit_code = $ARGV[4];
    print "base_path: $base_path, aoi: $aoi, processor_name: $processor_name, Sleep: $sleep_sec , ExitCode: $exit_code\n";
    print "sleep $sleep_sec seconds...\n";
    sleep $sleep_sec;
    exit $exit_code;
}
```

The Task definition for the script above is:

```
<Task name="testExitCode" type="cmd" active="yes" priority="NORM">
  <EMail>mail@host.yourdomain</EMail>
  <NotifyEmail>OnError</NotifyEmail>
  <Command>perl</Command>
  <Parameter>./test_data/scripts/testPerlExitCode.pl</Parameter>
  <Parameter>@mySleep</Parameter>
  <Parameter>@myExitCode</Parameter>
</Task>
```

## Guidelines

### Processing of large set of files

For some tasks and processes it might be necessary to choose some optimizations for running a given processor, for examples when reprocessing a large set of files:

- Setup a dedicated independent DES instance for executing given tasks or processor
- Ensure that \$BASE\_PATH and \$BASE\_PATH are placed on a local disc rather than a shared network filesystem that might have extra latencies
- Make sure that the \$CWD contains a subset of the files at a given time, and possibly only symbolic links to the actual data (See DES\_Plugin recursiveSymlinkSetup.pl)

### Use of symbolik links when referencing files

In certain situations, for better management and more security, it might be more appropriate to have just the references (Symbolic Links) to the data inside the \$BASEPATH/\$AOI/\$PROCESSOR directory, instead of the data itself.

For security reason, the use of symbolic link is strongly recommended because it will avoid a potential, not foreseen, deletion of files by the Task implementation.

A dedicated Task plugin can be used to create the symbolic links and prepare the data environment for subsequent Tasks:

```
<AOI name="southtyrol" active="yes" sleepFactor="10">
  <Task name="createSymlinks" type="cmd" active="yes" priority="NORM">
    <RunOnTrigger>null</RunOnTrigger>
    <EMail>mail@host.yourdomain</EMail>
    <NotifyEmail>OnError</NotifyEmail>
    <Command>perl</Command>
```

```

<Parameter>./plugins/recursiveSymlinkSetup.pl</Parameter>
<Parameter>$CWD</Parameter>
<Parameter>/mnt/COMPUTE/GeoServer/coverages/PROCESSING/$PROCESSOR_$AOI/data
</Parameter>
<Parameter>EURAC_RADBEAM_P.*.tif</Parameter>
</Task>
</AOI>

```

The Task entity defined above calls a perl script (recursiveSymlinkSetup.pl), which creates the symbolic links with respect to the parameters indicated:

1) the first parameter represents the folder where the symbolic links will be created i.e. \$CWD 2) the second parameter represents the folder where the data to be pushed towards the FTP server are located i.e. /mnt/COMPUTE/GeoServer/coverages/PROCESSING/\$PROCESSOR\_\$AOI/data 3) the third parameter indicates a regular expression that can be used to filter the data i.e. EURAC\_RADBEAM\_P.\*.tif

Note: The sleep factor has been set to 10 to let the Task run with a different time frequency in order to optimize the overall DES execution performance

## Howto's

### Create a Trigger for a Task

There are different methods to trigger a Task execution process: - Using the tag RunOnTrigger - Using the tags SartDate and StopDate - Using the filtering tag FileType

The most flexible method is actually the FileType filter, as everything is treated as a file being processed

#### Using FileType filter method

The presence of a dedicated trigger file (eg. TR\$PROCESSOR.\*.trigger) , or also any other file, filtered by the expression, acts as a driver for the Task execution. This allows than to create a list of trigger files that the DES can then elaborate in a FIFO queue.

Because the Date for example is often used as input parameter for a processing time series for example, the Date has to be passed as input parameter to the Task executing a particular job.

One approach to implement this, is to ensure that the trigger file TR\$PROCESSOR.\*.trigger includes the date at a certain position, i.e position 5 (. separated):

```
TREURAC_RADBEAM_P.southtyrol_highres.6120_6120.hour.8.2017-02-10T15:00:00.Eurac.01.00.trigger
```

So the implemented wrapper Script (Perl, Python, Shell) or Java Class for the Task can extract the processing parameters as needed, i.e. The date to be processed. The trigger file itself might contain even a more structured meta information in form of XML for example.

By convention the trigger file should have the following naming convention:

```
TR$PROCESSOR.$AOI.TileID.Type.Value.Date.Eurac.$versionMajor.$versionMinor.trigger
```

Any other naming convention can be adapted by providing an appropriate wrapper script that is called.

#### Example

Processor configuration:

```

<TaskGroup name="Processing" type="serial" active="yes" priority="NORM">
  <Task name="createMap" type="cmd" active="yes" priority="NORM">
    <EMail>mail@host.yourdomain</EMail>
    <NotifyEmail>OnError</NotifyEmail>
    <FileType>TREURAC_RADBEAM_P.*.trigger</FileType>
    <Command>sh</Command>
    <Parameter>/Algorithms/runProc.sh</Parameter>
  </Task>
  <Task>
    ...
  </Task>
</TaskGroup>

```

Task implementation: (runProc.sh)

```

#!/bin/sh
arg=(*)
BASE_PATH=$1

```

```

AOI_NAME=$2
PROCESSOR_NAME=$3
TRIGGER_FILE_NAME=$4

echo "creating environment vars.."
# User specific aliases and functions
export PATH=$PATH:/Algorithms/

echo "start processing..."

file=$TRIGGER_FILE_NAME
echo "  processing file: $file"

DATE_FROM_FILE=$(echo "$file" | cut -d '.' -f 6)
echo "date: $DATE_FROM_FILE"

DATE=`date -d "$DATE_FROM_FILE" +%Y-%m-%d`
HOUR=`date -d "$DATE_FROM_FILE" +%H`
HOUR2=$HOUR
HOUR=`expr $HOUR2 + 1`
MIN=`date -d "$DATE_FROM_FILE" +%M`

echo "running algorithm for date: $DATE"

```

Note: The Date format contained in the Trigger file naming convention should comply if possible with the format `yyyymmddThhmmss`

The trigger files may than be generated manually or via a script called by a Cronjob (see example below):

```
/Cronjobs/createDESTaskTriggers.sh /raid0/DES/data southtyrol_highres EURAC_RADBEAM_P 6120_6120 [ 20170101T091500 ]
```

Cronjob Script: (createDESTaskTriggers.sh)

```

#!/bin/sh
arg=(*)
BASE_PATH=$1
AOI_NAME=$2
PROCESSOR_NAME=$3
TILE_ID=$4

echo `date`
echo "creating trigger..."
TRIGGER_PATH=$BASE_PATH/$AOI_NAME/$PROCESSOR_NAME
DATE=`date +%Y-%m-%dT%H:%M:%S`
echo "Date: $DATE"
TR_FILE="TR$PROCESSOR_NAME.$AOI_NAME.$TILE_ID.year.2017.$DATE.Eurac.01.00.trigger"
echo "creating trigger $TRIGGER_PATH/$TR_FILE"
touch $TRIGGER_PATH/$TR_FILE
touch $TRIGGER_PATH/$TR_FILE.md5
echo "done"
echo "-----"

```

## Run a DES instance in Test Mode

When testing a new processor it could be useful to have a test environment where it is possible to check the output. To do this is straightforward because for each installed DES instance there is a `test_data` folder where test can be performed.

Steps: - Create a symbolic link to your processor: `ln -s DES.jar DES_test.jar` to have a better control on your `DES_test` (if you decide to call it in this way) processor - Copy your `YOUR_PROCESSOR.processor_config.xml` file in the folder `/DES/test_data/processor_config/` - Do your changes to this file. For example you can enable or disable AOI, Task, TaskGroup, Distribution, Download, etc. - Open the file `/DES/config/DistributionServerTest.ini` and change the value of `LOGS_PATH` variable from `LOGS_PATH=LOG4J` to `LOGS_PATH=STDOUT`. This will allow to have the log in the standard output to facilitate the debug. - Change to the DES folder: `cd /DES` - Run the following command:

```
java -jar DES_test.jar -c ./config/DistributionServerTest.ini -p YOUR_PROCESSOR.processor_config.xml
```