

conversion service

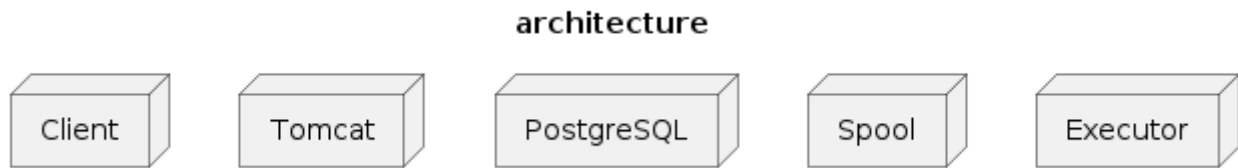
Table of Contents

1. Overview	1
1.1. Version information	2
1.2. URI scheme	2
1.3. Tags	2
2. Paths	3
2.1. POST /rest/cancelTask	3
2.1.1. Description	3
2.1.2. Parameters	3
2.1.3. Responses	3
2.1.4. Tags	3
2.2. POST /rest/createTask	3
2.2.1. Description	3
2.2.2. Parameters	3
2.2.3. Responses	4
2.2.4. Consumes	4
2.2.5. Produces	4
2.2.6. Tags	4
2.3. POST /rest/createTaskWithPriority	4
2.3.1. Description	4
2.3.2. Parameters	4
2.3.3. Responses	5
2.3.4. Consumes	5
2.3.5. Produces	5
2.3.6. Tags	5
2.4. GET /rest/downloadData	5
2.4.1. Description	5
2.4.2. Parameters	5
2.4.3. Responses	6
2.4.4. Produces	6
2.4.5. Tags	6
2.5. GET /rest/findTask	6
2.5.1. Description	6
2.5.2. Parameters	6
2.5.3. Responses	6
2.5.4. Produces	6
2.5.5. Tags	6
2.5.6. Example HTTP response	7
2.6. GET /rest/getFilters	7

2.6.1. Description	7
2.6.2. Responses	7
2.6.3. Produces	8
2.6.4. Tags	8
2.6.5. Example HTTP response	8
2.7. GET /rest/getStats	8
2.7.1. Description	8
2.7.2. Responses	8
2.7.3. Produces	9
2.7.4. Tags	9
2.7.5. Example HTTP response	9
2.8. GET /rest/getTask	9
2.8.1. Description	9
2.8.2. Parameters	9
2.8.3. Responses	9
2.8.4. Produces	10
2.8.5. Tags	10
2.8.6. Example HTTP response	10
2.9. GET /rest/listTaskIds	11
2.9.1. Description	11
2.9.2. Parameters	11
2.9.3. Responses	11
2.9.4. Produces	11
2.9.5. Tags	11
2.10. GET /rest/listTasks	12
2.10.1. Description	12
2.10.2. Parameters	12
2.10.3. Responses	12
2.10.4. Produces	12
2.10.5. Tags	12
2.10.6. Example HTTP response	12
2.11. POST /rest/reloadConfiguration	13
2.11.1. Description	13
2.11.2. Responses	13
2.11.3. Tags	14
2.12. POST /rest/sanity	14
2.12.1. Description	14
2.12.2. Responses	14
2.12.3. Produces	14
2.12.4. Tags	14
2.12.5. Example HTTP response	14

2.13. POST /rest/startExecutors	15
2.13.1. Description	15
2.13.2. Responses	15
2.13.3. Tags	15
2.14. POST /rest/startTask	15
2.14.1. Description	15
2.14.2. Parameters	15
2.14.3. Responses	15
2.14.4. Tags	15
2.15. POST /rest/stopExecutors	16
2.15.1. Description	16
2.15.2. Responses	16
2.15.3. Tags	16
2.16. POST /rest/terminateTask	16
2.16.1. Description	16
2.16.2. Parameters	16
2.16.3. Responses	16
2.16.4. Tags	16
2.17. POST /rest/uploadData	17
2.17.1. Description	17
2.17.2. Parameters	17
2.17.3. Responses	17
2.17.4. Consumes	17
2.17.5. Tags	17
2.18. GET /rest/version	17
2.18.1. Description	17
2.18.2. Responses	17
2.18.3. Produces	18
2.18.4. Tags	18
2.18.5. Example HTTP response	18
3. Definitions	19
3.1. Filter	19
3.2. FilterMixin	19
3.3. SanityResponse	19
3.4. SanityState	20
3.5. StatsResponse	20
3.6. Task	20
3.7. TaskMixin	21
3.8. TaskState	22
3.9. VersionResponse	22

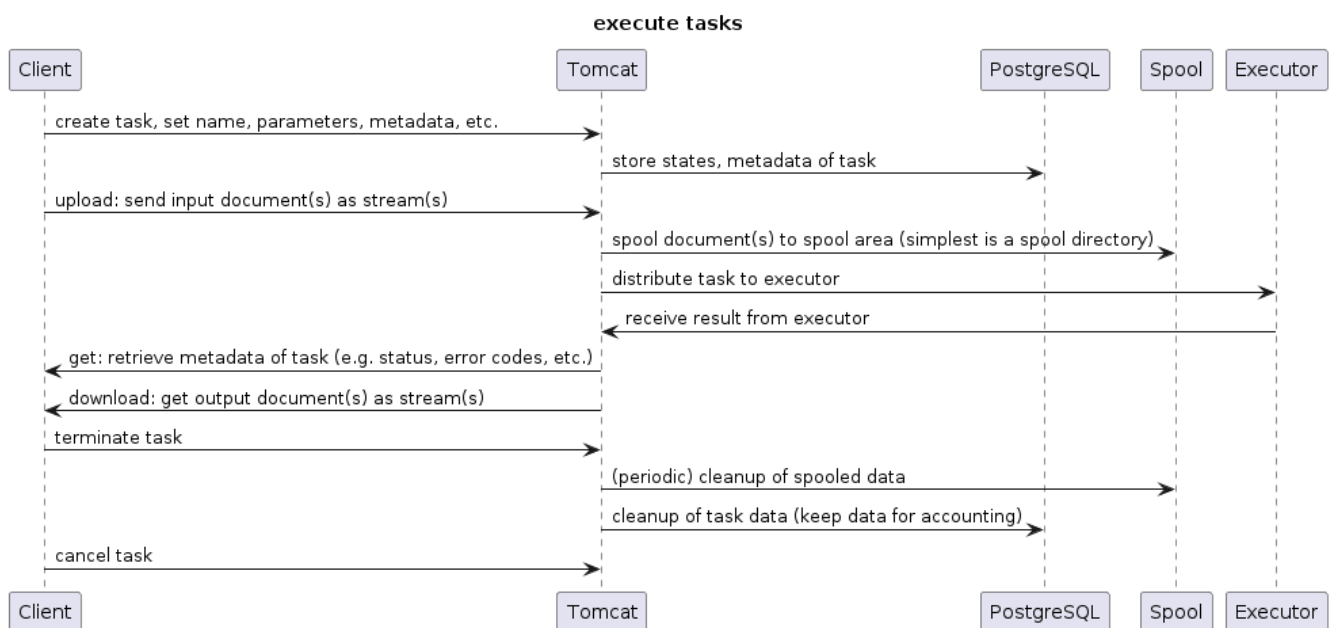
Chapter 1. Overview



Via the rest service you can:

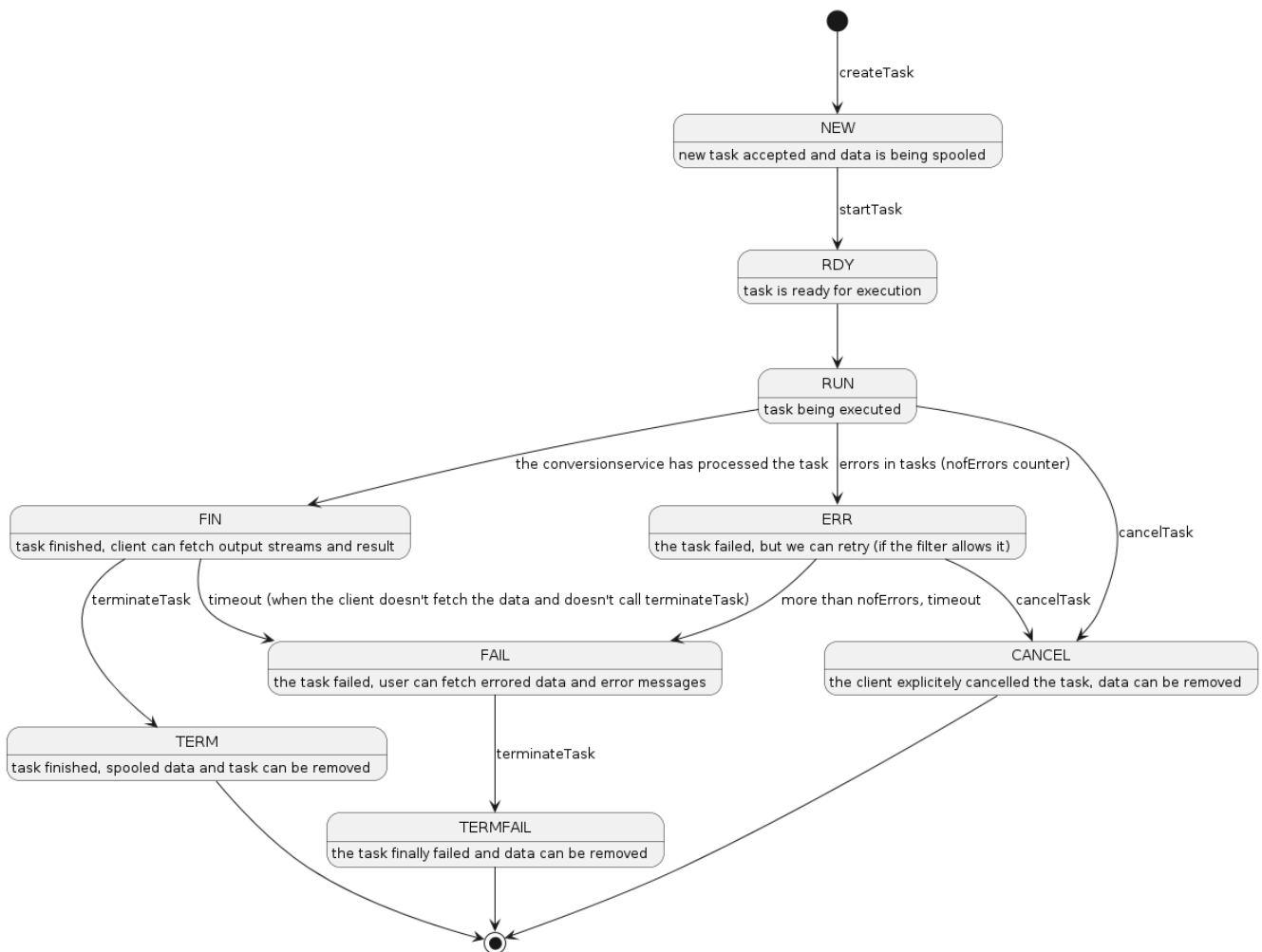
- Create and execute tasks
- Transfer big data for tasks in streams
- Poll for the status of the task
- Terminate or cancel tasks
- Get list of available filters
- Start/stop the internal task execution
- Reconfigure the service without restarting the service
- Check sanity of service for monitoring

Tasks can be executed on the service as follows:



Tasks transition through the following states:

task states



conversion service

1.1. Version information

Version : 1.0-SNAPSHOT

1.2. URI scheme

Host : conversionservice.eurospider.com

BasePath : /service/rest

1.3. Tags

- ConversionResourceImpl

Chapter 2. Paths

2.1. POST /rest/cancelTask

2.1.1. Description

Cancel a task.

2.1.2. Parameters

Type	Name	Schema
Query	id <i>required</i>	integer (int32)

2.1.3. Responses

HTTP Code	Description	Schema
200	Success	No Content
500	Internal Server Error	No Content

2.1.4. Tags

- ConversionResourceImpl

2.2. POST /rest/createTask

2.2.1. Description

Create a new task, give it a name, we can also pass parameters for filters or metadata to it. This task gets the standard priority of the owner of the task (the user of the API cannot override it).

2.2.2. Parameters

Type	Name	Schema
Query	filter <i>optional</i>	string

Type	Name	Schema
Query	name <i>optional</i>	string
Body	body <i>optional</i>	< string, string > map

2.2.3. Responses

HTTP Code	Description	Schema
200	Success	integer (int32)
500	Internal Server Error	No Content

2.2.4. Consumes

- `application/json`

2.2.5. Produces

- `application/json`

2.2.6. Tags

- `ConversionResourceImpl`

2.3. POST /rest/createTaskWithPriority

2.3.1. Description

Create a new task, give it a name, we can also pass parameters for filters or metadata to it. This task is prioritized by the user of the API.

2.3.2. Parameters

Type	Name	Schema
Query	filter <i>optional</i>	string
Query	name <i>optional</i>	string

Type	Name	Schema
Query	priority <i>required</i>	integer (int32)
Body	body <i>optional</i>	< string, string > map

2.3.3. Responses

HTTP Code	Description	Schema
200	Success	integer (int32)
500	Internal Server Error	No Content

2.3.4. Consumes

- `application/json`

2.3.5. Produces

- `application/json`

2.3.6. Tags

- `ConversionResourceImpl`

2.4. GET /rest/downloadData

2.4.1. Description

Download data associated to a task

2.4.2. Parameters

Type	Name	Schema
Query	id <i>required</i>	integer (int32)
Query	purpose <i>optional</i>	string

2.4.3. Responses

HTTP Code	Description	Schema
200	Success	file
500	Internal Server Error	No Content

2.4.4. Produces

- `application/json`

2.4.5. Tags

- `ConversionResourceImpl`

2.5. GET /rest/findTask

2.5.1. Description

Returns data about the task with a given name.

2.5.2. Parameters

Type	Name	Schema
Query	name <i>optional</i>	string

2.5.3. Responses

HTTP Code	Description	Schema
200	Success	Task
500	Internal Server Error	No Content

2.5.4. Produces

- `application/json`

2.5.5. Tags

- `ConversionResourceImpl`

2.5.6. Example HTTP response

Response 200

```
{
  "id" : 12345,
  "name" : "...",
  "parameters" : "...",
  "filterId" : 12345,
  "ownerId" : 12345,
  "state" : "NEW",
  "code" : 12345,
  "nofErrors" : 12345,
  "created" : 12345,
  "lastModification" : 12345,
  "priority" : 12345
}
```

Response 500

```
{
  "id" : 12345,
  "name" : "...",
  "parameters" : "...",
  "filterId" : 12345,
  "ownerId" : 12345,
  "state" : "RUN",
  "code" : 12345,
  "nofErrors" : 12345,
  "created" : 12345,
  "lastModification" : 12345,
  "priority" : 12345
}
```

2.6. GET /rest/getFilters

2.6.1. Description

Return the list of available filters.

2.6.2. Responses

HTTP Code	Description	Schema
200	Success	< Filter > array

HTTP Code	Description	Schema
500	Internal Server Error	No Content

2.6.3. Produces

- `application/json`

2.6.4. Tags

- `ConversionResourceImpl`

2.6.5. Example HTTP response

Response 200

```
[ {
  "id" : 12345,
  "name" : "...",
  "maxNofErrors" : 12345
} ]
```

Response 500

```
[ {
  "id" : 12345,
  "name" : "...",
  "maxNofErrors" : 12345
} ]
```

2.7. GET /rest/getStats

2.7.1. Description

Return a map of statistics.

2.7.2. Responses

HTTP Code	Description	Schema
200	Success	StatsResponse
500	Internal Server Error	No Content

2.7.3. Produces

- `application/json`

2.7.4. Tags

- `ConversionResourceImpl`

2.7.5. Example HTTP response

Response 200

```
{
  "stats" : {
    "property1" : { },
    "property2" : { }
  }
}
```

Response 500

```
{
  "stats" : {
    "property1" : { },
    "property2" : { }
  }
}
```

2.8. GET /rest/getTask

2.8.1. Description

Returns data about the task with a given id.

2.8.2. Parameters

Type	Name	Schema
Query	id <i>required</i>	integer (int32)

2.8.3. Responses

HTTP Code	Description	Schema
200	Success	Task
500	Internal Server Error	No Content

2.8.4. Produces

- `application/json`

2.8.5. Tags

- `ConversionResourceImpl`

2.8.6. Example HTTP response

Response 200

```
{
  "id" : 12345,
  "name" : "...",
  "parameters" : "...",
  "filterId" : 12345,
  "ownerId" : 12345,
  "state" : "TERM",
  "code" : 12345,
  "nofErrors" : 12345,
  "created" : 12345,
  "lastModification" : 12345,
  "priority" : 12345
}
```

Response 500

```
{
  "id" : 12345,
  "name" : "...",
  "parameters" : "...",
  "filterId" : 12345,
  "ownerId" : 12345,
  "state" : "ERR",
  "code" : 12345,
  "nofErrors" : 12345,
  "created" : 12345,
  "lastModification" : 12345,
  "priority" : 12345
}
```

2.9. GET /rest/listTaskIds

2.9.1. Description

Returns list of task ids between two dates.

2.9.2. Parameters

Type	Name	Schema
Query	from <i>optional</i>	string
Query	to <i>optional</i>	string

2.9.3. Responses

HTTP Code	Description	Schema
200	Success	< integer > array
500	Internal Server Error	No Content

2.9.4. Produces

- `application/json`

2.9.5. Tags

- `ConversionResourceImpl`

2.10. GET /rest/listTasks

2.10.1. Description

Returns list of task data between two dates.

2.10.2. Parameters

Type	Name	Schema
Query	from <i>optional</i>	string
Query	to <i>optional</i>	string

2.10.3. Responses

HTTP Code	Description	Schema
200	Success	< Task > array
500	Internal Server Error	No Content

2.10.4. Produces

- `application/json`

2.10.5. Tags

- `ConversionResourceImpl`

2.10.6. Example HTTP response

Response 200

```
[ {
  "id" : 12345,
  "name" : "...",
  "parameters" : "...",
  "filterId" : 12345,
  "ownerId" : 12345,
  "state" : "FAIL",
  "code" : 12345,
  "nofErrors" : 12345,
  "created" : 12345,
  "lastModification" : 12345,
  "priority" : 12345
} ]
```

Response 500

```
[ {
  "id" : 12345,
  "name" : "...",
  "parameters" : "...",
  "filterId" : 12345,
  "ownerId" : 12345,
  "state" : "RUN",
  "code" : 12345,
  "nofErrors" : 12345,
  "created" : 12345,
  "lastModification" : 12345,
  "priority" : 12345
} ]
```

2.11. POST /rest/reloadConfiguration

2.11.1. Description

Reload configuration of service without restarting it, this is mainly to change parameters to filter or adding/removing filters at runtime

2.11.2. Responses

HTTP Code	Description	Schema
200	Success	No Content
500	Internal Server Error	No Content

2.11.3. Tags

- `ConversionResourceImpl`

2.12. POST /rest/sanity

2.12.1. Description

Check sanity of service, used for monitoring

2.12.2. Responses

HTTP Code	Description	Schema
200	Success	SanityResponse
500	Internal Server Error	No Content

2.12.3. Produces

- `application/json`

2.12.4. Tags

- `ConversionResourceImpl`

2.12.5. Example HTTP response

Response 200

```
{
  "state" : "WARNING",
  "message" : "encountered 2 service exceptions, "
}
```

Response 500

```
{
  "state" : "WARNING",
  "message" : "encountered 2 service exceptions, "
}
```

2.13. POST /rest/startExecutors

2.13.1. Description

Start executors to handle tasks

2.13.2. Responses

HTTP Code	Description	Schema
200	Success	No Content
500	Internal Server Error	No Content

2.13.3. Tags

- ConversionResourceImpl

2.14. POST /rest/startTask

2.14.1. Description

Start the task, all parameters, metadata and data has been uploaded and spool. Mark task ready for execution.

2.14.2. Parameters

Type	Name	Schema
Query	id <i>required</i>	integer (int32)

2.14.3. Responses

HTTP Code	Description	Schema
200	Success	No Content
500	Internal Server Error	No Content

2.14.4. Tags

- ConversionResourceImpl

2.15. POST /rest/stopExecutors

2.15.1. Description

Stop executors to handle tasks

2.15.2. Responses

HTTP Code	Description	Schema
200	Success	No Content
500	Internal Server Error	No Content

2.15.3. Tags

- ConversionResourceImpl

2.16. POST /rest/terminateTask

2.16.1. Description

Terminate a task after having fetched all data from it (metadata results data, logfiles, etc.)

2.16.2. Parameters

Type	Name	Schema
Query	id <i>required</i>	integer (int32)

2.16.3. Responses

HTTP Code	Description	Schema
200	Success	No Content
500	Internal Server Error	No Content

2.16.4. Tags

- ConversionResourceImpl

2.17. POST /rest/uploadData

2.17.1. Description

Upload data associated to a task

2.17.2. Parameters

Type	Name	Schema
Query	id <i>required</i>	integer (int32)
Query	purpose <i>optional</i>	string
Body	body <i>optional</i>	file

2.17.3. Responses

HTTP Code	Description	Schema
200	Success	No Content
500	Internal Server Error	No Content

2.17.4. Consumes

- `application/json`

2.17.5. Tags

- ConversionResourceImpl

2.18. GET /rest/version

2.18.1. Description

Get version of the service

2.18.2. Responses

HTTP Code	Description	Schema
200	Success	VersionResponse
500	Internal Server Error	No Content

2.18.3. Produces

- `application/json`

2.18.4. Tags

- `ConversionResourceImpl`

2.18.5. Example HTTP response

Response 200

```
{
  "major" : 0,
  "minor" : 1
}
```

Response 500

```
{
  "major" : 0,
  "minor" : 1
}
```

Chapter 3. Definitions

3.1. Filter

Name	Schema
id <i>required</i>	integer (int32)
maxNofErrors <i>required</i>	integer (int32)
name <i>optional</i>	string

3.2. FilterMixin

Polymorphism : Composition

Name	Schema
id <i>required</i>	integer (int32)
maxNofErrors <i>required</i>	integer (int32)
name <i>optional</i>	string

3.3. SanityResponse

Response to a sanity request

Name	Description	Schema
message <i>optional</i>	Messages given an indications why the service is in a bad state (in case the service is not in state OK)	string
state <i>optional</i>	The state of the service	SanityState

3.4. SanityState

Enumeration of all sanity states the service can be in.

OK: Service is running ok, all docfetchers are ok.

WARNING: Service encountered warnings.

ERROR: Service encountered errors.

FATAL: Service is in a critical state and most likely stopped to work properly.

Type : enum (OK, WARNING, ERROR, FATAL)

3.5. StatsResponse

Response to a stats request

Name	Description	Schema
stats <i>optional</i>	Statistics, depend on the configured filters.	< string, object > map

3.6. Task

Name	Schema
code <i>required</i>	integer (int32)
created <i>optional</i>	integer (int32)
filterId <i>required</i>	integer (int32)
id <i>required</i>	integer (int32)
lastModification <i>optional</i>	integer (int32)
name <i>optional</i>	string
nofErrors <i>required</i>	integer (int32)

Name	Schema
ownerId <i>required</i>	integer (int32)
parameters <i>optional</i>	string
priority <i>required</i>	integer (int32)
state <i>optional</i>	TaskState

3.7. TaskMixin

Polymorphism : Composition

Name	Schema
code <i>required</i>	integer (int32)
created <i>optional</i>	integer (int32)
filterId <i>required</i>	integer (int32)
id <i>required</i>	integer (int32)
lastModification <i>optional</i>	integer (int32)
name <i>optional</i>	string
nofErrors <i>required</i>	integer (int32)
ownerId <i>required</i>	integer (int32)

Name	Schema
parameters <i>optional</i>	string
priority <i>required</i>	integer (int32)
state <i>optional</i>	TaskState

3.8. TaskState

Possible states of tasks in the executor state machine and in the service.

Type : enum (NEW, RDY, RUN, FIN, TERM, CANCEL, ERR, FAIL, TERMFAIL)

3.9. VersionResponse

Response to a request of the version of the API, contains major and minor of the version.

Name	Description	Schema
major <i>required</i>	The major version number. Example : 1	integer (int32)
minor <i>required</i>	The minor version number. Example : 0	integer (int32)