API Reference

BTC Embedded Platform - RESTful API documentation

API Version: 2.11p1

This is the documentation for the BTC Embedded Platform RESTful API.

CONTACT

EMAIL: support@btc-es.de

INDEX

API

1. APPLICATION

Handle the EP application itself.

1.1 DELETE /ep/application

Exit EP and save the active profile

After calling the exit method, the current active profile will be saved, if there is one. If the profile has no save path, one is created at a temporary directory and a response containing the path is returned.

REQUEST

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
force-quit	boolean	True will force quit the application without saving the active profile. False will save the profile at the given location, or at a temporary location if provided none. If the parameter is not provided, the latter behavior will be chosen.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 201: Created

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

2. ARCHITECTURES

Import and retrieve architectures.

2.1 POST /ep/architectures/ccode

Import a C-Code architecture

Long running task Import a C-Code architecture. Settings are provided as an import info object.

REQUEST

REQUEST BODY - application/json

```
{
  modelFile*   string  File name of the C-Code XML file
  mappingFile  string  File name of the mapping XML fileMust be provided if architectures are already availablein the profile.
                       The file is used to map existing architecturesto the new imported CCode architecture.
}
```

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed by using the GET '/EP/Progress/{progressId}' using the id received by this command.

RESPONSE MODEL - application/json

```
{
  jobID  string  READ-ONLY
                 The ID of a job.
}
```

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

2.2 POST /ep/architectures/embedded-coder

Import an EmbeddedCoder™ architecture

Long running task Import an EmbeddedCoder™ architecture. Settings are provided as an import info object.

REQUEST

REQUEST BODY - application/json

```
{
  ecModelFile*          string   The absolute or relativ path to the init script of the EmbeddedCoder model.
  ecInitScript          string   The absolute or relativ path to the init script of the EmbeddedCoder model.
  parameterHandling     enum     ALLOWED:OFF, EXPLICIT_PARAMETER
                                 'OFF': Only regular inputs in the interface of subsystems are
                                 observed.<br>'EXPLICIT_PARAMETER': Parameter variables are regarded as
                                 additional<br>inputs to subsystems.<br>Default is 'EXPLICIT_PARAMETER'.
  testMode              enum     ALLOWED:BLACK_BOX, GREY_BOX
                                 If set to 'GreyBox', local variables are regarded as additional outputs of
                                 subsystems.<br>For BlackBox-Testing only the regularoutputs in the interfaces of
                                 subsystems are observed.<br>Default is 'GreyBox'.
  fixedStepSolver       boolean  Handling when a non-fixed-step solver is ecountered: If 'true', the solver is automatically
                                 set to fixed-step.<br>Otherwise an error is issued.
  inDepthCcodeAnalysis  boolean  true: Simulink model and C-Code model are imported.<br>false: indicates that a SL SIL
                                 simulation is supported.<br>C-Code and mapping are omitted.<br>Default is 'true'
}
```

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed by using the GET '/ep/progress/{progress-id}' using the id received by this command.

RESPONSE MODEL - application/json

```
{
  jobID  string  READ-ONLY
                 The ID of a job.
}
```

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

2.3 GET /ep/architectures/{architecture-uid}

Get a specific architecture

Get an existing architecture by UID.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*architecture-uid	string	The UID of the architecture to be returned.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
{
  uid               string  READ-ONLY
                            The unique identifier (UID) of this object.
  architectureKind  string  READ-ONLY
```

The string representation of the concrete architecture.

name string READ-ONLY

The architecture name as specified by the model.

propList {

The Architecture property List

}

}

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

string

2.4 GET /ep/architectures

Get all architectures

Search for all open architectures.

REQUEST

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
architecture-kind	string	Specifies the specific architecture kind to be queried. (e.g. 'Simulink', 'C-Code', 'TargetLink')

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

[{

Array of object:

uid string READ-ONLY

The unique identifier (UID) of this object.

architectureKind string READ-ONLY

The string representation of the concrete architecture.

name string READ-ONLY

The architecture name as specified by the model.

propList {

The Architecture property List

}

}]

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

string

2.5 PUT /ep/architectures

Update architectures

Long running task Perform an architecture update.

REQUEST

No request parameters

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed by using the GET '/EP/Progress/{progressId}' using the id received by this command.

RESPONSE MODEL - application/json

```
{
  jobID  string  READ-ONLY
                 The ID of a job.
}
```

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

2.6 PUT /ep/architectures/model-paths/{architecture-uid}

Update model paths for architectures

Update the paths for imported architectures. If the architecture-uid is not specified, the model paths will be updated for the master architecture.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*architecture-uid	string	The UID of the architecture to be updated. If empty, the path will be updated for the master architecture.

REQUEST BODY - application/json

```
{
  slModelFile  string  The path to the Simulink model(.mdl|.slx).
```

slInitScript	string	The path to the init script of the Simulink model.
addModelInfo	string	The path to additional model information.
tlModelFile	string	The path to the TargetLink model file (.mdl|.slx).
tlInitScript	string	The path to the init script of the TargetLink model.
environment	string	The path to XML file including information about environmental files like additional code.

}

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - text/plain

string

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

string

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

string

2.7 POST /ep/architectures/simulink

Import a Simulink™ architecture

Long running task Import a Simulink™ architecture. Settings are provided as an import info object.

REQUEST

REQUEST BODY - application/json

{

slModelFile*	string	The absolute or relativ path to the Simulink model. This file is mandatory and may not be null!
slInitScriptFile	string	The absolute or relativ path to the init script of the Simulink model. This file may be null. If the specified file path is invalid, an error is reported.
addModelInfoFile*	string	The absolute or relativ path to additional model information. The format is described by the AddModelInfo.dtd. This file may be null.
parameterHandling	enum	ALLOWED:OFF, EXPLICIT_PARAMETER Specifies how parameter variables are handled. Can be 'OFF' or 'EXPLICIT_PARAMETER'. 'OFF': Only regular inputs in the interface of subsystems are observed. 'EXPLICIT_PARAMETER': Parameter variables are regarded as additional inputs to subsystems. Default is 'OFF'. If not specified (null or invalid), the default value is used.
testMode	enum	ALLOWED:BLACK_BOX, GREY_BOX Specifies the test mode. Can be 'BLACK_BOX' or 'GREY_BOX'. If set to 'GREY_BOX', local variables are regarded as additional outputs of subsystems. For Black Box-Testing only the regular outputs in the interfaces of subsystems are observed. Default is 'GREY_BOX'. If not specified (null or invalid), the default value is used.
fixedStepSolver	boolean	Defines, if the fixed step solver is used or not. Can be true or false. true: The analyzed Simulink model will be set to the fixed-step solver type automatically. The usage of the EmbeddedPlatform requires a fixed-step solver. If the model is open and the fixed-step solver is not already set, this might lead to a modified model. false: The analyzed Simulink model will not be set to the fixed-step solver automatically. If the fixed-step solver is not already set in the model, the method return with state of a non fixed-step solver. In order to proceed the user has to set the fixed-step solver manually in the Simulink simulation settings. Default is false. Note: The option is ignored if the model is currently not in an open/loaded state. In this case it has no visible side-effect.

mappingFile	string	File name of the mapping XML file. Must be provided if architectures are already available in the profile. This file may be null. If the specified file path is invalid, an error is reported.

}

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed by using the GET '/ep/progress/{progress-id}' using the id received by this command.

RESPONSE MODEL - application/json

{

jobID	string	READ-ONLY The ID of a job.

}

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

string

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

string

2.8 POST /ep/architectures/targetlink

Import a TargetLink™ architecture

Long running task Import a TargetLink™ architecture. Settings are provided as an import info object.

REQUEST

REQUEST BODY - application/json

{

tlModelFile*	string	The absolute or relativ path to the TargetLink model file (.mdl|.slx).
tlInitScript	string	The absolute or relativ path to the script defining all parameters needed for initializing the TL-model. If not provided, the TL-model is assumed to be selfcontained.
slModelFile	string	The absolute or relativ path to the Simulink model file (.mdl|.slx). Note: If a TargetLink model is given, the SL-model is assumed to be equivalent to the TL-model.
slInitScript	string	The absolute or relativ path to the script defining all parameters needed for initializing the SL-model. If not provided, the SL-model is assumed to be selfcontained.
environment	string	The absolute or relativ path to XML file including information about environmental files like additional code. (see specifications in CodeGeneration.dtd).
useExistingCode	boolean	'true': CodeGeneration is not explicitly performed. Instead it is assumed that the required code files are already generated and that the corresponding DataDictionary of the model includes information about the CodeGeneration. 'false': CodeGeneration is explicitly performed during the analysis. The resulting code is used for further steps. Default is 'false'.
activateModelLinking	boolean	'true': A linking between the source code and the TargetLink model will be established. This setting may lead to a modified TargetLink model. To accomplish this link relation, TargetLink option 'ExtendedBlockComments' needs to be enabled with a subsequent TargetLink code generation. 'false': The source code model linking is not explictitly set by EmbeddedPlatform. Default is 'false'.
closedLoopModel	boolean	'true': The environment of the SUT is also analyzed during the model extraction. Important for analyzing closed-loop models. 'false': Only the SUT is considered during the model extraction. Default is 'false'.
fixedStepSolver	boolean	'true': The analyzed models (TargetLink model and Simulink model) will be set to the fixed-step solver type automatically. The usage of the EmbeddedPlatform requires a fixed-step solver. If the model is open and the fixed-step solver is not already set, this

might lead to a modified model.
'false': The analyzed models (TargetLink model and Simulink model) will not be set to the fixed-step solver automatically. If the fixed-step solver is not already set in the model, an exception is thrown. In order to proceed the user has to set the fixed-step solver manually in the simulation settings.
Note: The option is ignored if the model is currently not in an open/loaded state. In this case it has no visible side-effect.
Default is 'false'.

tlSubsystem	string	Name of the Subsystem representing the TL toplevel subsystem for the analysis. Note: Argument is obligatory if there is more than one toplevel system in the model.
calibrationHandling	enum	ALLOWED:OFF, EXPLICIT_PARAMETER, LIMITED_BLOCKSET 'OFF': Only regular inputs in the interface of subsystems are observed. 'EXPLICIT_PARAMETER': Calibration variables are regarded as additional inputs to subsystems. Their value is set once during the initial phase of the simulation and is held constant thereafter. 'LIMITED_BLOCKSET': Calibration variables are regarded as additional inputs to subsystems. Their value is set once during the initial phase of the simulation and is held constant thereafter. Default is 'EXPLICIT_PARAMETER'. Note: The supported Set of of calibrateable TL-Blocks is different from 'EXPLICIT_PARAMETER'.
testMode	enum	ALLOWED:BLACK_BOX, GREY_BOX 'GREY_BOX': If set to 'GREY_BOX', display variables are regarded as additional outputs of subsystems. 'BLACK_BOX': For BlackBox-Testing only the regular outputs in the interfaces of subsystems are observed. Default is 'GREY_BOX'.
mappingFileForTLArch	string	The absolute or relativ path to the mapping file for TargetLink architecture mapping. Must be provided if architectures are already available in the profile. The file is used to map existing architectures to the new imported TargetLink architecture.
mappingFileForCCodeArch	string	The absolute or relativ path to the mapping file for CCode architecture mapping. Must be provided if architectures are already available in the profile. The file is used to map existing architectures to the new imported CCode architecture.
mappingFileForSLArch	string	The absolute or relativ path to the mapping file for Simulink architecture mapping. Must be provided if a Simulink architecture is additionally imported and architectures are already available in the profile. The file is used to map existing architectures to the new imported Simulink architecture.

}

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed by using the GET '/ep/progress/{progress-id}' using the id received by this command.

RESPONSE MODEL - application/json

{
  jobID string READ-ONLY
  The ID of a job.
}

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

string

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

string

STATUS CODE - 406: Not acceptable

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

string

3. BACK-TO-BACK TEST REPORTS

3.1 GET /ep/b2b-reports

Get all B2B test reports

Retrieve all the Back-to-Back test reports from the profile.

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
[{
Array of object:
   uid          string  READ-ONLY
                        The unique identifier (UID) of this object.
   reportName   string  Name of the report.
   reportType   string  Type of the report.
}]
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

3.2 POST /ep/b2b/{b2b-test-uid}/b2b-reports

Create a B2B test report on a B2B test

Creates a Back-to-Back test report on a Back-to-Back test by providing its UID.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*b2b-test-uid	string	The Back-to-Back test UID for which the Back-to-Back test report is created.

RESPONSE

STATUS CODE - 201: Created

RESPONSE MODEL - application/json

```
{
   uid          string  READ-ONLY
                        The unique identifier (UID) of this object.
   reportName   string  Name of the report.
```

```
    reportType  string  Type of the report.
}
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

---

4. BACK-TO-BACK TESTS

Create and retrieve Back-to-Back tests.

4.1 POST /ep/folders/b2b

Create a B2B test on a list of folders

Long running task Generates a Back-to-Back test on a given list of folder UIDs for the specified reference and comparison execution kinds. Optionally, the execution can be forced to simulate all contained requirements-based test cases/stimuli-vectors. Optionally, only stimuli vectors can be set to be executed if force execution is activated. Otherwise if there are requirements-based test cases with comparison verdicts, the verdicts will be deleted in order to run the Back-to-Back test.

REQUEST

REQUEST BODY - application/json

{

refMode*	string	Reference execution mode (e.g. 'SL MIL', 'TL MIL', 'TL MIL (EV)', 'SIL', 'PIL', 'SL SIL')
compMode*	string	Comparison execution mode (e.g. 'SL MIL', 'TL MIL', 'TL MIL (EV)', 'SIL', 'PIL', 'SL SIL')
forceExecute	boolean	(Optional) If true, simulates all contained test cases/stimuli-vectors, replacing existing execution records. Default is false.
simulateStimuliVectorsOnly	boolean	(Optional) This option can be used only when forceExecution is set to true. If true, simulates only stimuli-vectors. If false, and there are Requirements-based Test Cases with comparison verdicts, the verdicts will be deleted in order to run the Back-To-Back Test. Default value is false.
UIDs*	[string]	UIDs list (e.g. scopes, folders)

}

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed at /ep/progress/{progressId}.

RESPONSE MODEL - application/json

{

jobID	string	READ-ONLY The ID of a job.

}

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

string

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

string

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

string

4.2 POST /ep/scopes/b2b

Create a B2B test on a list of scopes

Long running task Generates a Back-to-Back test on a given list of scope UIDs for the specified reference and comparison execution kinds. Optionally, the execution can be forced to simulate all contained requirements-based test cases/stimuli-vectors. Optionally, only stimuli vectors can be set to be executed if force execution is activated. Otherwise if there are requirements-based test cases with comparison verdicts, the verdicts will be deleted in order to run the Back-to-Back test.

REQUEST

REQUEST BODY - application/json

```
{
   refMode*                    string     Reference execution mode (e.g. 'SL MIL', 'TL MIL', 'TL MIL (EV)', 'SIL', 'PIL', 'SL SIL')
   compMode*                   string     Comparison execution mode (e.g. 'SL MIL', 'TL MIL', 'TL MIL (EV)', 'SIL', 'PIL', 'SL SIL')
   forceExecute                boolean    (Optional) If true, simulates all contained test cases/stimuli-vectors, replacing
                                          existing execution records. Default is false.
   simulateStimuliVectorsOnly  boolean    (Optional) This option can be used only when forceExecution is set to true. If true,
                                          simulates only stimuli-vectors. If false, and there are Requirements-based Test
                                          Cases with comparison verdicts, the verdicts will be deleted in order to run the
                                          Back-To-Back Test. Default value is false.
   UIDs*                       [string]   UIDs list (e.g. scopes, folders)
}
```

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed at /ep/progress/{progressId}.

RESPONSE MODEL - application/json

```
{
   jobID  string  READ-ONLY
                  The ID of a job.
}
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

4.3 POST /ep/folders/{folder-uid}/b2b

Create a B2B test on a folder

Long running task Generates a Back-to-Back test on a given folder UID for the specified reference and comparison execution kinds. Optionally, the execution can be forced to simulate all contained requirements-based test cases/stimuli-

vectors. Optionally, only stimuli vectors can be set to be executed if force execution is activated. Otherwise if there are requirements-based test cases with comparison verdicts, the verdicts will be deleted in order to run the Back-to-Back test.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*folder-uid	string	The folder UID for which the Back-to-Back test is generated.

REQUEST BODY - application/json

```
{
  refMode*                  string   Reference execution mode (e.g. 'SL MIL', 'TL MIL', 'TL MIL (EV)', 'SIL', 'PIL', 'SL SIL')
  compMode*                 string   Comparison execution mode (e.g. 'SL MIL', 'TL MIL', 'TL MIL (EV)', 'SIL', 'PIL', 'SL SIL')
  forceExecute              boolean  (Optional) If true, simulates all contained test cases/stimuli-vectors, replacing
                                     existing execution records. Default is false.
  simulateStimuliVectorsOnly boolean (Optional) This option can be used only when forceExecution is set to true. If true,
                                     simulates only stimuli-vectors. If false, and there are Requirements-based Test Cases
                                     with comparison verdicts, the verdicts will be deleted in order to run the Back-To-Back
                                     Test. Default value is false.
}
```

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed at /ep/progress/{progressId}.

RESPONSE MODEL - application/json

```
{
  jobID  string  READ-ONLY
                 The ID of a job.
}
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

4.4 GET /ep/b2b/{b2b-uid}

Get a B2B test

Get a Back-to-Back test with the provided UID.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*b2b-uid	string	The UID of the Back-to-Back test to be returned.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
{
  uid                  string    READ-ONLY
                                 The unique identifier (UID) of this object.
  name                 string    The name of the Back-To-Back Test.
  compMode             string    The comparison execution config type.
  referenceFolderUIDs  [string]  Reference folder UIDs
  comparisonFolderUID  string    Comparison folder UID
  executionDate        string    Execution Date
  verdictStatus        enum      ALLOWED:PASSED, FAILED, ERROR, FAILED_ACCEPTED
                                 PASSED, FAILED, ERROR, FAILED_ACCEPTED
  verdictState         enum      ALLOWED:VALID, OUTDATED_TOLERANCE_UPDATE,
                                 OUTDATED_MISSING_EXECUTIONS
                                 Verdict State
  failed               integer   Number of failed comparisons.
  failedAccepted       integer   Number of failed accepted comparisons.
  passed               integer   Number of passed comparisons.
  error                integer   Number of comparisons with error.
  total                integer   Total number of comparisons.
  comparisons [{
  Array of object:  All comparisons.
    uid                            string  READ-ONLY
                                           The unique identifier (UID) of this object.
    name                           string  The name of Test Case / Stimuli Vector used in Comparison.
    verdictStatus                  enum    ALLOWED:PASSED, FAILED, ERROR, FAILED_ACCEPTED
                                           PASSED, FAILED, ERROR, FAILED_ACCEPTED
    referenceExecutionRecordUID    string  UID of reference execution record.
    comparisonExecutionRecordUID   string  UID of compared to execution record.
    comment                        string  Added comment for Comparison.
  }]
  refMode              string    The reference execution config type.
  stimuliFolderUIDs    [string]  Folder UIDs for which Back-To-Back Test was generated.
  stimuliScopeUIDs     [string]  Scope UIDs for which Back-To-Back Test was generated.
}
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

4.5 PATCH /ep/b2b/{b2b-uid}

Change a verdict status for a comparison

Changes verdict status for a comparison. If accept is true, the comparison verdict status is changed from 'failed' to 'failed (accepted)'. If accept is false, the comparison verdict status is changed from 'failed (accepted)'to 'failed'.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*b2b-uid	string	The Back-to-Back test UID for which to change the comparison verdict.

REQUEST BODY - application/json

```
{
  comparisonUID*  string   UID of the Comparison
  accept*         boolean  If accept is true, the comparison verdict status is changed from 'failed' to 'failed (accepted)'. If accept is false, the comparison verdict status is changed from 'failed (accepted)'to 'failed'
}
```

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

4.6 POST /ep/scopes/{scope-uid}/b2b

Create a B2B test on a scope

Long running task Generates a Back-to-Back test on a given scope UID for the specified reference and comparison execution kinds. Optionally, the execution can be forced to simulate all contained requirements-based test cases/stimuli-vectors. Optionally, only stimuli vectors can be set to be executed if force execution is activated. Otherwise if there are requirements-based test cases with comparison verdicts, the verdicts will be deleted in order to run the Back-to-Back test.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*scope-uid	string	The scope UID for which the Back-to-Back test is generated.

REQUEST BODY - application/json

```
{
  refMode*                  string  Reference execution mode (e.g. 'SL MIL', 'TL MIL', 'TL MIL (EV)', 'SIL', 'PIL', 'SL SIL')
  compMode*                 string  Comparison execution mode (e.g. 'SL MIL', 'TL MIL', 'TL MIL (EV)', 'SIL', 'PIL', 'SL SIL')
```

forceExecute	boolean	(Optional) If true, simulates all contained test cases/stimuli-vectors, replacing existing execution records. Default is false.
simulateStimuliVectorsOnly	boolean	(Optional) This option can be used only when forceExecution is set to true. If true, simulates only stimuli-vectors. If false, and there are Requirements-based Test Cases with comparison verdicts, the verdicts will be deleted in order to run the Back-To-Back Test. Default value is false.

}

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed at /ep/progress/{progressId}.

RESPONSE MODEL - application/json

{

jobID	string	READ-ONLY The ID of a job.

}

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

string

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

string

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

string

4.7 GET /ep/b2b

Get all B2B tests

Get all Back-to-Back tests from active profile.

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

[{

Array of object:

uid	string	READ-ONLY The unique identifier (UID) of this object.
name	string	The name of the Back-To-Back Test.
compMode	string	The comparison execution config type.
referenceFolderUIDs	[string]	Reference folder UIDs

```
  comparisonFolderUID  string    Comparison folder UID
  executionDate        string    Execution Date
  verdictStatus        enum      ALLOWED:PASSED, FAILED, ERROR, FAILED_ACCEPTED
                                 PASSED, FAILED, ERROR, FAILED_ACCEPTED
  verdictState         enum      ALLOWED:VALID, OUTDATED_TOLERANCE_UPDATE,
                                 OUTDATED_MISSING_EXECUTIONS
                                 Verdict State
  failed               integer   Number of failed comparisons.
  failedAccepted       integer   Number of failed accepted comparisons.
  passed               integer   Number of passed comparisons.
  error                integer   Number of comparisons with error.
  total                integer   Total number of comparisons.
  comparisons [{
  Array of object:  All comparisons.
    uid                              string  READ-ONLY
                                             The unique identifier (UID) of this object.
    name                             string  The name of Test Case / Stimuli Vector used in Comparison.
    verdictStatus                    enum    ALLOWED:PASSED, FAILED, ERROR,
                                             FAILED_ACCEPTED
                                             PASSED, FAILED, ERROR, FAILED_ACCEPTED
    referenceExecutionRecordUID      string  UID of reference execution record.
    comparisonExecutionRecordUID     string  UID of compared to execution record.
    comment                          string  Added comment for Comparison.
  }]
  refMode              string    The reference execution config type.
  stimuliFolderUIDs    [string]  Folder UIDs for which Back-To-Back Test was generated.
  stimuliScopeUIDs     [string]  Scope UIDs for which Back-To-Back Test was generated.
}]
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

5. CODE ANALYSIS REPORTS B2B

Handle Back-To-Back code analysis reports.

5.1 GET /ep/code-analysis-reports-b2b

Get all reports

Retrieve all B2B code analysis reports from profile.

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
[{
Array of object:
  uid          string  READ-ONLY
                       The unique identifier (UID) of this object.
  reportName   string  Name of the report.
  reportType   string  Type of the report.
}]
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

5.2 POST /ep/scopes/{scope-uid}/code-analysis-reports-b2b

Create a report on a scope

Create a B2B code analysis report on given scope.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*scope-uid	string	The scope UID for which to create the B2B code analysis report.

RESPONSE

STATUS CODE - 201: Created

RESPONSE MODEL - application/json

```
{
  uid          string  READ-ONLY
                       The unique identifier (UID) of this object.
```

```
    reportName  string  Name of the report.
    reportType  string  Type of the report.
  }
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

5.3 POST /ep/folders/code-analysis-reports-b2b

Create a report on a list of folders

Create B2B Code Analysis Report on given folders.

REQUEST

REQUEST BODY - application/json

```
{
  UIDs*  [string]  A list with unique identifiers.
}
```

RESPONSE

STATUS CODE - 201: Created

RESPONSE MODEL - application/json

```
{
  uid         string  READ-ONLY
                      The unique identifier (UID) of this object.
  reportName  string  Name of the report.
  reportType  string  Type of the report.
}
```

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

5.4 POST /ep/folders/{folder-uid}/code-analysis-reports-b2b

Create a report on a folder

Create a B2B code analysis report on given folder.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*folder-uid	string	The folder UID for which to create the B2B code analysis report.

RESPONSE

STATUS CODE - 201: Created

RESPONSE MODEL - application/json

```
{
  uid         string  READ-ONLY
                      The unique identifier (UID) of this object.
  reportName  string  Name of the report.
  reportType  string  Type of the report.
}
```

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

6. CODE ANALYSIS REPORTS RBT

Creates requirement-based code analysis reports.

6.1 POST /ep/requirements-sources/code-analysis-reports-rbt

Create a report on requirements sources

Create an RBT code analysis report on given requirements sources.

REQUEST

REQUEST BODY - application/json

```
{
  UIDs* [string]  A list with unique identifiers.
}
```

RESPONSE

STATUS CODE - 201: Created

RESPONSE MODEL - application/json

```
{
  uid         string  READ-ONLY
                      The unique identifier (UID) of this object.
  reportName  string  Name of the report.
  reportType  string  Type of the report.
}
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

6.2 POST /ep/requirements-sources/{requirements-source-uid}/code-analysis-reports-rbt

Create a report on a requirements source

Create an RBT code analysis report on a given requirements source UID.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*requirements-source-uid	string	The requirements source UID for which to create RBT code analysis report.

RESPONSE

STATUS CODE - 201: Created

RESPONSE MODEL - application/json

```
{
  uid         string  READ-ONLY
                      The unique identifier (UID) of this object.
  reportName  string  Name of the report.
  reportType  string  Type of the report.
}
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

6.3 GET /ep/code-analysis-reports-rbt

Get all reports

Retrieve all RBT code analysis reports from the profile.

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
[{
Array of object:
  uid         string  READ-ONLY
                      The unique identifier (UID) of this object.
  reportName  string  Name of the report.
  reportType  string  Type of the report.
}]
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

6.4 POST /ep/scopes/{scope-uid}/code-analysis-reports-rbt

Create a report on scope

Create an RBT code analysis report on a given scope.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*scope-uid	string	The scope UID for which to create RBT code analysis report.

RESPONSE

STATUS CODE - 201: Created

RESPONSE MODEL - application/json

```
{
  uid         string  READ-ONLY
                      The unique identifier (UID) of this object.
  reportName  string  Name of the report.
  reportType  string  Type of the report.
}
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

6.5 POST /ep/folders/code-analysis-reports-rbt

Create a report on a list of folders

Create an RBT code analysis report on given folders.

REQUEST

REQUEST BODY - application/json

```
{
  UIDs*  [string]  A list with unique identifiers.
}
```

RESPONSE

STATUS CODE - 201: Created

RESPONSE MODEL - application/json

```
{
  uid         string  READ-ONLY
                      The unique identifier (UID) of this object.
  reportName  string  Name of the report.
  reportType  string  Type of the report.
}
```

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

string

6.6 POST /ep/folders/{folder-uid}/code-analysis-reports-rbt

Create a report on a folder

Create an RBT code analysis report on a given folder.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*folder-uid	string	The folder UID for which to create an RBT code analysis report.

RESPONSE

STATUS CODE - 201: Created

RESPONSE MODEL - application/json

```
{
  uid          string  READ-ONLY
                       The unique identifier (UID) of this object.
  reportName   string  Name of the report.
  reportType   string  Type of the report.
}
```

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

string

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

string

---

7. COVERAGE GENERATION

For a C-Code function, create stimuli vectors which cover the code function, or mark coverage properties that are unreachable.

7.1 GET /ep/coverage-generation

Get the configuration

Get the configuration with all possible settings for a stimuli vector generation run with the default values set.

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
{
  isSubscopesGoalsConsidered*    boolean   Whether or not goals from sub scopes should be considered.
  targetDefinitions* [{
  Array of object:  The target definitions to use for this run.
    label*    string   The label of the target definition.
    enabled*  boolean  Whether or not this target definition should be considered.
  }]
  folderName                     string    Name of the folder to store Stimuli Vectors in. If this folder exists, will use
                                           that folder. Else, will create a new folder with the given name.
  checkUnreachableProperties*    boolean   Whether or not unreachable properties should be (re-)checked.
  pllString                      string    PLL String for specifc goals to be reached. Default is '::' which matches all
                                           goals. Use e.g. 'STM;D;CDC' to find stimuli vectors for statement, decision,
                                           and condition/decision coverage. Individual PLLs can be addressed by
                                           their specific label (e.g. 'D:4:1'). Multiple PLLs can be concatenated using
                                           semicolon, e.g. 'D:4:1;C:2'. See the user guide for more information about
                                           the property location labels.If this is null or empty, only the selected
                                           targetDefinitions will be used.
  engineSettings* {
  The engine settings to use for this run.
    timeoutSeconds*                integer   Global timeout (seconds) for the execution
    handlingRateThreshold*         integer   After each scope is analyzed, the 'handled rate' of the entry scope
                                             (potentially including goals from subscopes) is checked against
                                             this threshold and the stimuli vector generation is stopped when it
                                             is reached. Allowed range are integers from [1, 100] (percent of
                                             handled goals). Default: 100
    analyseSubScopesHierarchically* boolean  Enables / disables recursive analysis of subscopes.
    engineAtg* {
    The ATG engine
      name*                        string    The name of the engine (heuristic).
      use*                         boolean   Whether or not this engine should be used in the run.
      searchDepthSteps*            integer   The search depth (number of SUT iterations)
      executionMode*               enum      ALLOWED:TOP_DOWN, BOTTOM_UP
                                             The search direction, bottom up or top down
      timeoutSecondsPerSubsystem*  integer   Timeout (seconds) per scope
    }
    engineCv* {
    The CV engine
      name*                        string    The name of the engine (heuristic).
      use*                         boolean   Whether or not this engine should be used in the run.
      searchDepthSteps*            integer   The search depth (number of SUT iterations)
      timeoutSecondsPerSubsystem*  integer   Timeout (seconds) per scope
```

```
      timeoutSecondsPerProperty*   integer  Timeout (seconds) per coverage property
      memoryLimitMb*               integer  The maximum amount of system memory to use (MB)
      loopUnroll*                  integer  The number of internal loop unwindings for potentially unbounded
                                            loops within each SUT iteration.
      coreEngines* [{
      Array of object:  The core engines to use
        name*  string   The name of the core engine.
        use*   boolean  Whether or not to use this core engine in the execution.
      }]
      assumptionCheckEnabled*      boolean  Whether or not core engines are allowed to explicitly checkthe
                                            satisfiability of the selected assumptions.
      searchFocus*                 enum     ALLOWED:BALANCED, REACHABLE, UNREACHABLE
                                            The search focus of the core engines.
      parallelExecutionMode*       enum     ALLOWED:BALANCED, ENGINES, GOALS
                                            The mode used for the parallel engine execution.If maximum number
                                            of threads used is 1, the value of this parameter is not used (instead
                                            the default value BALANCED will be used).
      maximumNumberOfThreads*      integer  The maximum number of threads available for parallel engine
                                            execution for core engines. Valid values are between 1 and available
                                            number of cores. It is possible to set this parameter to a value of -1,
                                            which will compute the number of threads automatically as half of the
                                            available number of cores.
    }
    allowDenormalizedFloats*       boolean  Whether or not the engine may produce denormalized floats.
  }
  scope* {
  The entry scope to use for this run.
    uid          string   The unique identifier (UID) of this object.
    name         string   The scope name.
    topLevel     boolean  TRUE if scope is a toplevel scope.
    kind         enum     ALLOWED:SUT, DUMMY, ENVIRONMENT, HIDDEN_INTERNAL, VIRTUAL
                          Scope kind.
    path         string   Scope path.
    architecture string   The corresponding architecture of the scope.
    sampleTime {
    The sample time of the scope.
      uid     string  The unique identifier (UID) of this object.
      seconds string  The sample time as a value given in seconds.
    }
  }
  assumptions [{
  Array of object:  The environmental assumptions to use for this run.
    id*    string   The Assumption UID.
    name*  string   The name of the Environmental Assumption.
    use*   boolean  Whether or not this assumption should be used in the execution.
  }]
  drivers [{
  Array of object:  The drivers to use for this run.
    id*    string   The Driver source UID.
    name*  string   The name of the Driver source
    use*   boolean  Whether or not this Driver source should be used in the execution.
  }]
}
```

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 406: Not Acceptable

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

7.2 POST /ep/coverage-generation

Execute coverage generation

Long running task Using the provided configuration, execute the coverage generation / stimuli vector generation.

REQUEST

REQUEST BODY - application/json

```
{
  isSubscopesGoalsConsidered*    boolean   Whether or not goals from sub scopes should be considered.
  targetDefinitions* [{
  Array of object:  The target definitions to use for this run.
    label*    string   The label of the target definition.
    enabled*  boolean  Whether or not this target definition should be considered.
  }]
  folderName                     string    Name of the folder to store Stimuli Vectors in. If this folder exists, will use
                                           that folder. Else, will create a new folder with the given name.
  checkUnreachableProperties*    boolean   Whether or not unreachable properties should be (re-)checked.
  pllString                      string    PLL String for specifc goals to be reached. Default is '::' which matches all
                                           goals. Use e.g. 'STM;D;CDC' to find stimuli vectors for statement, decision,
                                           and condition/decision coverage. Individual PLLs can be addressed by their
                                           specific label (e.g. 'D:4:1'). Multiple PLLs can be concatenated using
                                           semicolon, e.g. 'D:4:1;C:2'. See the user guide for more information about the
                                           property location labels.If this is null or empty, only the selected
                                           targetDefinitions will be used.
  engineSettings* {
  The engine settings to use for this run.
    timeoutSeconds*                  integer   Global timeout (seconds) for the execution
    handlingRateThreshold*           integer   After each scope is analyzed, the 'handled rate' of the entry scope
                                               (potentially including goals from subscopes) is checked against this
                                               threshold and the stimuli vector generation is stopped when it is
                                               reached. Allowed range are integers from [1, 100] (percent of handled
                                               goals). Default: 100
    analyseSubScopesHierarchically*  boolean   Enables / disables recursive analysis of subscopes.
    engineAtg* {
    The ATG engine
      name*                          string    The name of the engine (heuristic).
      use*                           boolean   Whether or not this engine should be used in the run.
      searchDepthSteps*              integer   The search depth (number of SUT iterations)
      executionMode*                 enum      ALLOWED:TOP_DOWN, BOTTOM_UP
                                               The search direction, bottom up or top down
      timeoutSecondsPerSubsystem*    integer   Timeout (seconds) per scope
    }
    engineCv* {
    The CV engine
      name*                          string    The name of the engine (heuristic).
      use*                           boolean   Whether or not this engine should be used in the run.
```

```
    searchDepthSteps*            integer  The search depth (number of SUT iterations)
    timeoutSecondsPerSubsystem*  integer  Timeout (seconds) per scope
    timeoutSecondsPerProperty*   integer  Timeout (seconds) per coverage property
    memoryLimitMb*               integer  The maximum amount of system memory to use (MB)
    loopUnroll*                  integer  The number of internal loop unwindings for potentially unbounded loops
                                          within each SUT iteration.
    coreEngines* [{
    Array of object:  The core engines to use
      name*  string   The name of the core engine.
      use*   boolean  Whether or not to use this core engine in the execution.
    }]
    assumptionCheckEnabled*      boolean  Whether or not core engines are allowed to explicitly checkthe
                                          satisfiability of the selected assumptions.
    searchFocus*                 enum     ALLOWED:BALANCED, REACHABLE, UNREACHABLE
                                          The search focus of the core engines.
    parallelExecutionMode*       enum     ALLOWED:BALANCED, ENGINES, GOALS
                                          The mode used for the parallel engine execution.If maximum number of
                                          threads used is 1, the value of this parameter is not used (instead the
                                          default value BALANCED will be used).
    maximumNumberOfThreads*      integer  The maximum number of threads available for parallel engine execution
                                          for core engines. Valid values are between 1 and available number of
                                          cores. It is possible to set this parameter to a value of -1, which will
                                          compute the number of threads automatically as half of the available
                                          number of cores.
  }
  allowDenormalizedFloats*       boolean  Whether or not the engine may produce denormalized floats.
}
scope* {
The entry scope to use for this run.
  uid          string   The unique identifier (UID) of this object.
  name         string   The scope name.
  topLevel     boolean  TRUE if scope is a toplevel scope.
  kind         enum     ALLOWED:SUT, DUMMY, ENVIRONMENT, HIDDEN_INTERNAL, VIRTUAL
                        Scope kind.
  path         string   Scope path.
  architecture string   The corresponding architecture of the scope.
  sampleTime {
  The sample time of the scope.
    uid      string  The unique identifier (UID) of this object.
    seconds  string  The sample time as a value given in seconds.
  }
}
assumptions [{
Array of object:  The environmental assumptions to use for this run.
  id*    string   The Assumption UID.
  name*  string   The name of the Environmental Assumption.
  use*   boolean  Whether or not this assumption should be used in the execution.
}]
drivers [{
Array of object:  The drivers to use for this run.
  id*    string   The Driver source UID.
  name*  string   The name of the Driver source
  use*   boolean  Whether or not this Driver source should be used in the execution.
}]
}
```

RESPONSE

STATUS CODE - 202: Accepted

RESPONSE MODEL - application/json

```
{
  jobID  string  READ-ONLY
                  The ID of a job.
}
```

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 406: Not Acceptable

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

8. DOMAIN CHECKS

Handle domain checks.

8.1 GET /ep/scopes/{scope-uid}/domain-checks-results

Get domain checks results

Get the domain checks results for a scope. Using the use case option will display the results for SVs and RBTest Cases if B2B is used or only for RBTest Cases if RBT option is used. Also, the results can be requested either only for the invalid goal types and for the valid ones, or for all goal types.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*scope-uid	string	The UID of the scope for which to get the domain checks results.

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
useCase	string	The use case for which to retrieve the domain check results. Possible values: B2B, RBT (not case-sensitive).Can be empty, in which case the results are shown for all use cases.
goalType	string	The goal type for which to retrieve the domain check results. Possible values: VALID, INVALID (not case-sensitive).Can be empty, in which case the results are shown for all goal types.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

{

- totalCountValid — string — READ-ONLY
  The total number of valid range goals.
- coveredCountValid — string — READ-ONLY
  The number of covered valid range goals.
- unreachableCountValid — string — READ-ONLY
  The number of unreachable valid range goals.
- errorCountValid — string — READ-ONLY
  The number of errorneous valid range goals.
- handledCountValid — string — READ-ONLY
  The number of handled valid range goals.
- unhandledCountValid — string — READ-ONLY
  The number of unhandled valid range goals.
- coveredPercValid — string — READ-ONLY
  The covered percentage for valid range goals.
- unreachablePercValid — string — READ-ONLY
  The unreachable percentage for valid range goals.
- errorPercValid — string — READ-ONLY
  The error percentage for valid range goals.
- handledPercValid — string — READ-ONLY
  The handled percentage for valid range goals.
- unhandledPercValid — string — READ-ONLY
  The unhandled percentage for valid range goals.
- totalCountInvalid — string — READ-ONLY

The total number of invalid range goals.

coveredCountInvalid string READ-ONLY
The number of covered invalid range goals.

unreachableCountInvalid string READ-ONLY
The number of unreachable invalid range goals.

errorCountInvalid string READ-ONLY
The number of errorneous invalid range goals.

handledCountInvalid string READ-ONLY
The number of handled invalid range goals.

unhandledCountInvalid string READ-ONLY
The number of unhandled invalid range goals.

coveredPercInvalid string READ-ONLY
The covered percentage for invalid range goals.

unreachablePercInvalid string READ-ONLY
The unreachable percentage for invalid range goals.

errorPercInvalid string READ-ONLY
The error percentage for invalid range goals.

handledPercInvalid string READ-ONLY
The handled percentage for invalid range goals.

unhandledPercInvalid string READ-ONLY
The unhandled percentage for invalid range goals.

}

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

string

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

string

8.2 POST /ep/domain-checks-ranges

Create domain checks ranges

Create the domain checks ranges for a scope and a list of signals. If the list of signals is not given, ranges for all signals of the scope will be created by default. The ranges can be created with some convenience functions applied (partitioned by a percent, with boundaries checks included or with invalid ranges checks included). Please note ALL domain checks ranges created via this service will overwrite any existing ranges for the given list of signals or for all signals (if no signal is specified).

REQUEST

REQUEST BODY - application/json

{

scopeUid* string The scope for which to create the domain checks ranges.

signalUids [string] The list of signals for which to create the domain checks ranges. If no signal is provided, ranges for all signals from the scope are created by default.

applyBoundaryChecks boolean Used for applying the boundary checks when creating the range.Can only be used for applying boundary checks on a defined range, so it must be used only together with one of the other options (either apply invalid ranges checks or partition ranges).

applyInvalidRangesChecks boolean Used for applying the invalid ranges checks when creating the range.

percentage integer Percentage used for partioning the range interval when creating the range.

}

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

8.3 POST /ep/domain-checks-export

Export domain check ranges

Export domain check ranges of the given scopeUid.

REQUEST

REQUEST BODY - application/json

```
{
   scopeUid*  string  The scopeUid for which to export/import domain check ranges.
   filePath*  string  The file path used for exporting/importing domain check ranges. The directory path specified in the filePath must
                      already exist.
}
```

RESPONSE

STATUS CODE - 202: Accepted

RESPONSE MODEL - application/json

```
{
   jobID  string  READ-ONLY
                  The ID of a job.
}
```

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

string

8.4 POST /ep/domain-checks

Import domain check ranges

Import domain check ranges on the given scopeUid.

REQUEST

REQUEST BODY - application/json

{

scopeUid*	string	The scopeUid for which to export/import domain check ranges.
filePath*	string	The file path used for exporting/importing domain check ranges. The directory path specified in the filePath must already exist.

}

RESPONSE

STATUS CODE - 202: Accepted

RESPONSE MODEL - application/json

{

jobID	string	READ-ONLY The ID of a job.

}

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

string

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

string

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

string

9. EXECUTION RECORDS

Handle execution records.

9.1 GET /ep/folders/{folder-uid}/execution-records

Get all execution records for a folder

Get all the execution records for a given folder UID.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*folder-uid	string	The folder UID from which to retrieve all execution records.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

[{
Array of object:

uid	string	READ-ONLY The unique identifier (UID) of this object.
executionConfig*	string	The execution config name
name*	string	The execution record name
status	enum	ALLOWED:OK, WARNING, ERROR The status of execution record.Possible options: OK, WARNING, or ERROR.
folderName*	string	The folder name on which this execution record can be found.
length*	integer	The length of execution record source
scopeName*	string	The scope name of execution record
sourceName*	string	The name of execution record source

}]

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

9.2 PUT /ep/folders/{folder-uid}/execution-records

Move a list of execution records to a folder

Moves the list of execution records to the requested user-defined execution record folder.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION

NAME	TYPE	DESCRIPTION
*folder-uid	string	The UID of user-defined execution record folder.

REQUEST BODY - application/json

```
{
  UIDs* [string]  UIDs of execution records to be moved.
}
```

RESPONSE

STATUS CODE - 200: OK

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

9.3 GET /ep/scopes/{scope-uid}/execution-records

Get all execution records for a scope

Get all the execution records for the given scope UID.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*scope-uid	string	The scope UID from which to retrieve all execution records.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
[{
```

Array of object:

uid	string	READ-ONLY The unique identifier (UID) of this object.
executionConfig*	string	The execution config name
name*	string	The execution record name
status	enum	ALLOWED:OK, WARNING, ERROR The status of execution record.Possible options: OK, WARNING, or ERROR.
folderName*	string	The folder name on which this execution record can be found.
length*	integer	The length of execution record source
scopeName*	string	The scope name of execution record
sourceName*	string	The name of execution record source

}]

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

9.4 GET /ep/execution-records

Get all execution records

Get all execution records available in the profile.

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

[{
Array of object:

uid	string	**READ-ONLY** The unique identifier (UID) of this object.
executionConfig*	string	The execution config name
name*	string	The execution record name
status	enum	**ALLOWED:**OK, WARNING, ERROR The status of execution record.Possible options: OK, WARNING, or ERROR.
folderName*	string	The folder name on which this execution record can be found.
length*	integer	The length of execution record source
scopeName*	string	The scope name of execution record
sourceName*	string	The name of execution record source

}]

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

9.5 POST /ep/execution-records

Import execution records

Import multiple execution records by providing the path for each file.

REQUEST

REQUEST BODY - application/json

```
{
  paths*                [string]  The path to all exection record files  you'd like to import.
  kind*                 string    The simulation kind that was used for creating the execution  ExecutionRecordImportInfo
                                  records from the given files. Possible values: It can be one of the default execution
                                  configurations (TL MIL, SL MIL, SIL, PIL) or it can be an external one. user defined folder
                                  option is not specified, the simulation kind willdefine the default folder where the execution
                                  records will be imported.
  folderName            string    User defined execution records folder name.If used, folder UID must not be specified.
  folderUID             string    Existing user defined folder uid to import records. <b color="red">If used, folderName can
                                  not be used at the same time </b>
  format                string    The format of the execution record file you want to import into.  Possible values: MDF, MF4
  referenceExternalFile boolean   whether to only reference the external file rather than importing the execution record in
                                  theprofile. This would be recommended for very big execution records.
}
```

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed at /EP/progress/{progressId}.

RESPONSE MODEL - application/json

```
{
  jobID  string  READ-ONLY
                 The ID of a job.
}
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not Found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

9.6 GET /ep/execution-records/{execution-record-uid}

Get an execution record

Get the requested execution record by UID.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*execution-record-uid	string	Execution record UID

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
[{
```
Array of object:

uid	string	READ-ONLY The unique identifier (UID) of this object.
executionConfig*	string	The execution config name
name*	string	The execution record name
status	enum	ALLOWED:OK, WARNING, ERROR The status of execution record.Possible options: OK, WARNING, or ERROR.
folderName*	string	The folder name on which this execution record can be found.
length*	integer	The length of execution record source
scopeName*	string	The scope name of execution record
sourceName*	string	The name of execution record source

```
}]
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

9.7 DELETE /ep/execution-records/{execution-record-uid}

Delete an execution record

Deletes the specified execution record.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*execution-record-uid	string	The UID of the execution record to be deleted.

RESPONSE

STATUS CODE - 200: OK

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 406: Not acceptable

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

9.8 POST /ep/execution-records-export

Export execution records

LONG RUNNING TASK Export single or multiple execution record(s) by providing the list of the execution records which will be exported, the export directory, the export format and a list of additional options for export.

REQUEST

REQUEST BODY - application/json

```
{
  UIDs*             [string]  List with the UIDs of the elements which will be exported
  exportDirectory*  string    Directory where to export the elements
  exportFormat      string    The format of the exported execution records. It can be: "MDF" or "Excel". (Default value: "MDF"). Please keep the specified format when you want to introduce a value.
}
```

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed at /ep/progress/{progress-id}.

RESPONSE MODEL - application/json

```
{
  jobID  string  READ-ONLY
                 The ID of a job.
}
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

10. FOLDERS

Handle folders.

10.1 GET /ep/folders

Get a list of folders

Get a list of folders by name and/or kind.

REQUEST

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
name	string	Enter the name of the folder you would like to search for. If null, then all folders will be returned.
kind	enum ALLOWED: RB_TEST_CASE, EXECUTION_RECORD, STIMULI_VECTOR	Enter the folder kind. If null, then all folder kinds will be returned.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
[{
```

Array of object:

uid	string	READ-ONLY The unique identifier (UID) of this object.
name*	string	The name of the folder.
kind*	string	The folder kind.
isDefault*	boolean	TRUE if its a dafault folder.

```
}]
```

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

10.2 POST /ep/folders

Create a folder

Add a folder by providing a FolderKind. Note that a new UID will be assigned.

REQUEST

REQUEST BODY - application/json

```
{
  folderKind  string  The folder kind. Possible: "RB_TEST_CASE", "EXECUTION_RECORD", "STIMULI_VECTOR"
}
```

RESPONSE

STATUS CODE - 201: Created

RESPONSE MODEL - application/json

```
{
  uid         string   READ-ONLY
                       The unique identifier (UID) of this object.
  name*       string   The name of the folder.
  kind*       string   The folder kind.
  isDefault*  boolean  TRUE if its a dafault folder.
}
```

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

10.3 DELETE /ep/folders/{folder-uid}

Delete a folder

Delete a folder by providing its UID.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*folder-uid	string	Enter the UID of the folder you would like to delete.

RESPONSE

STATUS CODE - 200: OK

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

11. FORMAL SPECIFICATION REPORTS

Formal Specification Reports

11.1 GET /ep/formal-specification-reports

Get all reports

Get all formal specification reports from the profile.

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
[{
Array of object:
  uid          string  READ-ONLY
                       The unique identifier (UID) of this object.
  reportName   string  Name of the report.
  reportType   string  Type of the report.
}]
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

11.2 POST /ep/formal-specification-reports

Create a formal specification report

Create a formal specification report on a list of formal specification UID's.

REQUEST

REQUEST BODY - application/json

```
{
  UIDs*  [string]  A list with unique identifiers.
}
```

RESPONSE

STATUS CODE - 201: Created

RESPONSE MODEL - application/json

```
{
  uid          string  READ-ONLY
                       The unique identifier (UID) of this object.
  reportName   string  Name of the report.
```

```
    reportType  string  Type of the report.
}
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

---

12. FORMAL SPECIFICATIONS

Handle Formal Specifications.

12.1 GET /ep/formal-requirements

Get all formal requirements

Get all formal requirements from the profile, or from the specified scope-uid.

REQUEST

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
scope-uid	string	The UID of scope from which to get the formal requirements.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
[{
```

Array of object:

uid	string	READ-ONLY The unique identifier (UID) of this object.
name*	string	The name of the formal requirement.
description*	string	The description of the formal requirement.
scopeUID*	string	The unique identifier (UID) of the scope this object belongs to.
errors	[string]	List of errors

```
}]
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

12.2 GET /ep/formal-requirements/{formal-requirement-uid}/environmental-assumptions

Get all environmental assumptions from a formal requirement

Use this command to retrieve the environmental assumptions from a formal requirement.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*formal-requirement-uid	string	The uid of the formal requirement for which all environmental assumptions should be returned.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

[{
Array of object:

uid	string	READ-ONLY The unique identifier (UID) of this object.
name*	string	The name of the environmental assumption.
description*	string	The description of the environmental assumption.
scopeUID*	string	The unique identifier (UID) of the scope this object belongs to.
errors	[string]	List of errors

}]

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

string

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

string

12.3 GET /ep/environmental-assumptions

Get all environmental assumptions

Get all environmental assumptions present on active profile, or from the specified scope-uid.

REQUEST

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
scope-uid	string	The UID of scope from which to get the environmental assumptions.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

[{
Array of object:

uid	string	READ-ONLY The unique identifier (UID) of this object.
name*	string	The name of the environmental assumption.
description*	string	The description of the environmental assumption.
scopeUID*	string	The unique identifier (UID) of the scope this object belongs to.
errors	[string]	List of errors

}]

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

12.4 POST /ep/specifications-export

Export a SPEC file

Exports the given formal requirements belonging to the same scope to the specified SPEC file.

REQUEST

REQUEST BODY - application/json

```
{
  specFile*              string     The file name of the target SPEC file. Should have .spec extension.
  archUid                string     The architecture UID to define the interface names. If specified, the SPEC file will use the
                                    interface and expression names based on the interfaces and signals defined in the given
                                    architecture.If not specified, the visible architecture is used for the name space.
  formalRequirementUids* [string]   A list of uids of formal requirement to export. Note: The formal requirements must reside
                                    on the same scope!
}
```

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

12.5 POST /ep/specifications-import

Import a SPEC file

Imports formal specifications from the specified SPEC file.
Long running task Specify artifact existing policy, one of EXTEND_NAME, OVERWRITE, or SKIP. By default it will be used 'EXTEND_NAME'.

REQUEST

REQUEST BODY - application/json

```
{
  specPath*    string  The path of the given SPEC file. Should have .spec extension.
  scopeId      string  The scopeId to use, when scope definition in SPEC file is invalid. This can happed for two reasons, the first is that
                       in the SPEC file, the component(scope) name is invalid and can not be found in the opened profile, or when the
                       given component(scope) name in the SPEC file is not unique within current profile.e.g. a TL architecture scope
                       name is unique to the one in the generated CCode.
```

optionParam enum ALLOWED:EXTEND_NAME, OVERWRITE, SKIP
The options of importing a SPEC file, when the artifacts already exists.Possible options: EXTEND_NAME, OVERWRITE, or SKIP.

}

RESPONSE

STATUS CODE - 202: Accepted

RESPONSE MODEL - application/json

{
jobID string READ-ONLY
The ID of a job.
}

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

string

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

string

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

string

13. INPUT RESTRICTIONS

13.1 POST /ep/input-restrictions-import

Import input restrictions

Import input restrictions from a file

REQUEST

REQUEST BODY - application/json

```
{
  filePath* string The file containing input restrictions.
}
```

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

13.2 POST /ep/input-restrictions-export

Export input restrictions

Export input restrictions to a file

REQUEST

REQUEST BODY - application/json

```
{
  filePath* string The file containing input restrictions.
}
```

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

14. INTERFACE REPORTS

Handle interface reports.

14.1 POST /ep/scopes/{scope-uid}/interface-reports

Create a report on a scope

Create an interface report on given scope. The interface report will use the interface of the architecture of the provided scope.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*scope-uid	string	The scope on which the interface report should be created. The interface report will use the interface of the architecture of the provide scope.

RESPONSE

STATUS CODE - 201: Created

RESPONSE MODEL - application/json

```
{
  uid         string  READ-ONLY
                      The unique identifier (UID) of this object.
  reportName  string  Name of the report.
  reportType  string  Type of the report.
}
```

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

14.2 GET /ep/interface-reports

Get all reports

Retrieve all interface reports from the profile.

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
[{
```
Array of object:

uid	string	READ-ONLY The unique identifier (UID) of this object.
reportName	string	Name of the report.
reportType	string	Type of the report.

```
}]
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

15. MESSAGES

Handle messages and message markers.

15.1 GET /ep/message-markers/{marker-date}/messages

Get a list of messages from a message marker

Search for messages created after a given message marker.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*marker-date	string	The message marker after which the messages should be queried from the Database.

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
search-string	string	Enter the beginning of the messages you would like to search for. If null, then all messages will be returned.
severity	enum ALLOWED: INFO, WARNING, ERROR, CRITICAL	Choose any severity you would like to search for. If null, then all severities will be returned.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
[{
```

Array of object:

uid	string	READ-ONLY The unique identifier (UID) of this object.
date	string	READ-ONLY The creation-date of the message.
message*	string	The message itself.
hint	string	An additional hint.
severity*	enum	ALLOWED:INFO, WARNING, ERROR, CRITICAL The severity of the message.

```
}]
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

15.2 POST /ep/message-markers

Create a message marker

Use this command to create a new message marker at the current time. The response will contain the created message marker time stamp as java Timestamp

REQUEST

No request parameters

RESPONSE

STATUS CODE - 201: Created

RESPONSE MODEL - application/json

```
{
  date  string  READ-ONLY
                The date when the marker was set.
}
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

15.3 GET /ep/messages/{message-uid}

Get a message

Get the message with the provided UID.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*message-uid	string	The UID of the message to be returned.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
{
  uid       string  READ-ONLY
                    The unique identifier (UID) of this object.
  date      string  READ-ONLY
                    The creation-date of the message.
  message*  string  The message itself.
  hint      string  An additional hint.
  severity* enum    ALLOWED:INFO, WARNING, ERROR, CRITICAL
                    The severity of the message.
}
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

string

15.4 DELETE /ep/messages/{message-uid}

Delete a message

Delete a message by providing its UID.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*message-uid	string	Enter the UID of the message you would like to delete.

RESPONSE

STATUS CODE - 200: OK

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

string

15.5 GET /ep/messages

Get a list of messages

Search for past messages up until a certain amount you can set yourself.

REQUEST

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
search-string	string	Enter the beginning of the messages you would like to search for. If null, then all messages will be returned.
severity	enum **ALLOWED:** INFO, WARNING, ERROR, CRITICAL	Choose any severity you would like to search for. If null, then all severities will be returned.
max-messages	int32	The maximum number of messages returned. Cannot be > 1000. If > 1000, negative, or null, at most 1000 messages will be returned.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

[{
Array of object:

uid string READ-ONLY

```
                        The unique identifier (UID) of this object.
  date        string    READ-ONLY
                        The creation-date of the message.
  message*    string    The message itself.
  hint        string    An additional hint.
  severity*   enum      ALLOWED:INFO, WARNING, ERROR, CRITICAL
                        The severity of the message.
}]
```

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

15.6 POST /ep/messages

Create a message

Add a message by providing a Message. Note that a new UID will be assigned.

REQUEST

REQUEST BODY - application/json

```
{
  uid         string    READ-ONLY
                        The unique identifier (UID) of this object.
  date        string    READ-ONLY
                        The creation-date of the message.
  message*    string    The message itself.
  hint        string    An additional hint.
  severity*   enum      ALLOWED:INFO, WARNING, ERROR, CRITICAL
                        The severity of the message.
}
```

RESPONSE

STATUS CODE - 201: Created

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

15.7 DELETE /ep/messages

Delete a list of messages

Search for past messages up until a certain amount you can set yourself and delete them.

REQUEST

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
search-string	string	Enter the beginning of the messages you would like to search for. If null, then all messages will be deleted.
severity	enum ALLOWED: INFO, WARNING, ERROR, CRITICAL	Choose any severity you would like to search for. If null, then all severities will be deleted.
max-messages	int32	The max number of messages deleted. If negative or null, all messages will be deleted.

RESPONSE

STATUS CODE - 200: OK

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

16. MODEL COVERAGE REPORTS

Creates RBT or B2B model coverage reports.

16.1 GET /ep/model-coverage-reports

Get a list of reports

Retrieve all model coverage reports of the specified testing use-case from the profile.

REQUEST

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
coverage-type	enum ALLOWED: RBT, B2B	The model coverage testing use-case.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

[{
Array of object:

uid	string	READ-ONLY The unique identifier (UID) of this object.
reportName	string	Name of the report.
reportType	string	Type of the report.

}]

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

string

16.2 POST /ep/scopes/{scope-uid}/model-coverage-reports

Create a report for a scope

Long running task Create RBT or B2B model coverage report on given scope.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*scope-uid	string	The scope UID for which to create the model coverage report.

REQUEST BODY - application/json

{

```
  type*                  enum     ALLOWED:RBT, B2B
                                  Specifies the testing use-case for which the model coverage report should be created.<br>Must
                                  be 'B2B' or 'RBT' for Back-to-back testing or Requirement based test, respectively.
  simulationKind*        string   Specifies the simulation mode: 'SL MIL' or 'TL MIL'
  useShortCircuitLogic   boolean  Specifies if short circuit logic should be used for Simulink blocks. <br>The paramter is optional.
                                  If not provided, the current setting from EP is used.
}
```

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed by using the GET '/ep/progress/{progress-id}' using the id received by this command.

RESPONSE MODEL - application/json

```
{
  jobID  string  READ-ONLY
                 The ID of a job.
}
```

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

16.3 POST /ep/folders/{folder-uid}/model-coverage-reports

Create a report for a folder

Long running task Create RBT or B2B model coverage report for given folder.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*folder-uid	string	The folder UID for which to create the model coverage report.

REQUEST BODY - application/json

```
{
  type*                  enum     ALLOWED:RBT, B2B
                                  Specifies the testing use-case for which the model coverage report should be
                                  created.<br>Must be 'B2B' or 'RBT' for Back-to-back testing or Requirement based test,
                                  respectively.
  simulationKind*        string   Specifies the simulation mode: 'SL MIL' or 'TL MIL'
  useShortCircuitLogic   boolean  Specifies if short circuit logic should be used for Simulink blocks. <br>The paramter is
```

optional. If not provided, the current setting from EP is used.

}

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed by using the GET '/ep/progress/{progress-id}' using the id received by this command.

RESPONSE MODEL - application/json

{
  jobID string READ-ONLY
  The ID of a job.
}

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

string

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

string

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

string

16.4 POST /ep/folders/model-coverage-reports

Create a report for a list of folders

Long running task Create RBT or B2B model coverage report for a list of folders.
In the RBT use-case, RBTestCases from the folders will be used to generate the report.
In the B2B use-case, RBTestCases and Stimuli-Vectors from the folders will be used to generate the report.

REQUEST

REQUEST BODY - application/json

{

type*	enum	ALLOWED:RBT, B2B Specifies the testing use-case for which the model coverage report should be created. Must be 'B2B' or 'RBT' for Back-to-back testing or Requirement based test, respectively.
simulationKind*	string	Specifies the simulation mode: 'SL MIL' or 'TL MIL'
folderUIDs*	[string]	The comma separated list of folder UIDs for which the model coverage report shall be created.
useShortCircuitLogic	boolean	Specifies if short circuit logic should be used for Simulink blocks. The paramter is optional. If not provided, the current setting from EP is used.

}

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed by using the GET '/ep/progress/{progress-id}' using the id received by this command.

RESPONSE MODEL - application/json

```
{
  jobID  string  READ-ONLY
                  The ID of a job.
}
```

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

17. PREFERENCES

Set and retrieve preferences.

17.1 PUT /ep/preferences

Set a list of preferences

Set new values for a list of given preferences. The preference name and new value must be provided for each of them.

REQUEST

REQUEST BODY - application/json

```
[{
```

Array of object:

```
  preferenceName*   string  The name of the preference.
  preferenceValue*  string  The value of the preference.
}]
```

RESPONSE

STATUS CODE - 200: Preferences set

RESPONSE MODEL - application/json

```
{
  messages [string]
}
```

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

17.2 GET /ep/preferences/{preference-name}

Get a preference

Get the preference with a given name. If the retrieved value is empty, the preference might have its default value.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*preference-name	string	Enter the name of the preference that you want retrieved.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
{
```

```
    preferenceName*   string  The name of the preference.
    preferenceValue*  string  The value of the preference.
  }
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

---

18. PROFILES

Create and handle EP Profiles.

18.1 GET /ep/profiles

Get the active profile

Use this command to get the currently active profile.

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
{
  uid   string   READ-ONLY
                 The unique identifier (UID) of this object.
  metadata* {
  The metadata containing all relevant info about the profile.
  }
  profilePath {
    path  string  The location where the profile is stored.
  }
}
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

18.2 PUT /ep/profiles

Save the profile

Use this command to save your active profile at a given location. Keep in mind to use only legal profile paths.

REQUEST

REQUEST BODY - application/json

```
{
  path  string  The location where the profile is stored.
}
```

RESPONSE

STATUS CODE - 201: Created

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

18.3 POST /ep/profiles

Create a profile

Use this command to create a new empty profile. It won't contain a path, since it's not stored anywhere yet.

REQUEST

No request parameters

RESPONSE

STATUS CODE - 201: Created

RESPONSE MODEL - application/json

```
{
  uid   string   READ-ONLY
                 The unique identifier (UID) of this object.
  metadata* {
  The metadata containing all relevant info about the profile.
  }
  profilePath {
    path  string  The location where the profile is stored.
  }
}
```

STATUS CODE - 428: Precondition Required

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

18.4 GET /ep/profiles/{profile-path}

Open a profile

Use this command to open an existing profile. Specifiy the path to the profile with a name of your choice. Keep in mind to only use legal profile paths of already existing profiles.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*profile-path	string	The path to the existing profile.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
{
  uid   string   READ-ONLY
                 The unique identifier (UID) of this object.
  metadata* {
  The metadata containing all relevant info about the profile.
  }
  profilePath {
    path  string  The location where the profile is stored.
  }
}
```

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 428: Precondition Required

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

19. PROGRESS

Get the current status of long-running operations.

19.1 GET /ep/progress/{progress-id}

Get the status of an operation

Get the status of the operation with the given progress id. If the operation is on-going, the current progress will be returned. If the operation is complete, the resulted object will be returned if there is one. An error will be returned if it occured during the long-running operation.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*progress-id	string	The progress id. Can be retrieved by starting a long-running operation.

RESPONSE

STATUS CODE - 200: Operation is complete. No resulting object is returned.

RESPONSE MODEL - application/json

```
{
  message  string
  progress integer
  result {
  }
}
```

STATUS CODE - 201: Operation is complete. Resulting object is returned as JSON.

RESPONSE MODEL - application/json

```
{
  message  string
  progress integer
  result {
  }
}
```

STATUS CODE - 202: Operation is currently in progress

RESPONSE MODEL - application/json

```
{
  message  string
  progress integer
  result {
  }
}
```

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

---

20. REGRESSION TEST REPORTS

Creates a Regression Test Report for a given Regression Test. Retrieves all existing Regression Test reports from the profile.

20.1 GET /ep/regression-test-reports

Get all reports

Retrieve all the Regression Test reports from the profile.

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
[{
Array of object:
  uid         string  READ-ONLY
                      The unique identifier (UID) of this object.
  reportName  string  Name of the report.
  reportType  string  Type of the report.
}]
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

20.2 POST /ep/regression-tests/{regression-test-uid}/regression-test-reports

Create a report for a regression test

Creates a Regression Test Report on a regression test.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*regression-test-uid	string	The Regression test UID for which the Regression Report is created.

RESPONSE

STATUS CODE - 201: Created

RESPONSE MODEL - application/json

```
{
  uid         string  READ-ONLY
                      The unique identifier (UID) of this object.
  reportName  string  Name of the report.
  reportType  string  Type of the report.
}
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

21. REGRESSION TESTS

Creates regression tests.

21.1 GET /ep/regression-tests/{regression-test-uid}

Get a test

Get the Regression Test with the provided UID.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*regression-test-uid	string	The UID of the Regression Test to be returned.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
{
  uid                 string    READ-ONLY
                                The unique identifier (UID) of this object.
  name                string    The name of the RegresionTest Test.
  compMode            string    The comparison execution config type.
  referenceFolderUIDs [string]  Reference folder UIDs
  comparisonFolderUID string    Comparison folder UID
  executionDate       string    Execution Date
  verdictStatus       enum      ALLOWED:PASSED, FAILED, ERROR, FAILED_ACCEPTED
                                PASSED, FAILED, ERROR, FAILED_ACCEPTED
  verdictState        enum      ALLOWED:VALID, OUTDATED_TOLERANCE_UPDATE,
                                OUTDATED_MISSING_EXECUTIONS
                                Verdict State
  failed              integer   Number of failed comparisons.
  failedAccepted      integer   Number of failed accepted comparisons.
  passed              integer   Number of passed comparisons.
  error               integer   Number of comparisons with error.
  total               integer   Total number of comparisons.
  comparisons [{
  Array of object:  All comparisons.
    uid                           string  READ-ONLY
                                          The unique identifier (UID) of this object.
    name                          string  The name of Test Case / Stimuli Vector used in Comparison.
    verdictStatus                 enum    ALLOWED:PASSED, FAILED, ERROR, FAILED_ACCEPTED
                                          PASSED, FAILED, ERROR, FAILED_ACCEPTED
    referenceExecutionRecordUID   string  UID of reference execution record.
    comparisonExecutionRecordUID  string  UID of compared to execution record.
    comment                       string  Added comment for Comparison.
  }]
}
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

21.2 PATCH /ep/regression-tests/{regression-test-uid}

Update verdict status for a comparison

Changes verdict status for a Comparison.If accept is true, the comparison verdict status is changed from 'failed' to 'failed (accepted)'. If accept is false, the comparison verdict status is changed from 'failed (accepted)'to 'failed'

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*regression-test-uid	string	The Regression Test UID for which to change the Comparison verdict.

REQUEST BODY - application/json

```
{
  comparisonUID*  string   UID of the Comparison
  accept*         boolean  If accept is true, the comparison verdict status is changed from 'failed' to 'failed (accepted)'. If accept is false, the comparison verdict status is changed from 'failed (accepted)'to 'failed'
}
```

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

21.3 POST /ep/folders/{folder-uid}/regression-tests

Generate a test on a folder

Long running task Generates a Regression Test on a given folder UID for the specified comparison execution kind. Optionally, the folder where to save the simulated execution records can be provided. If this property is not provided, the execution records are not saved.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION

NAME	TYPE	DESCRIPTION
*folder-uid	string	The folder UID for which the Regression Test is generated.

REQUEST BODY - application/json

```
{
  compMode*       string  Comparison execution mode (e.g. 'SL MIL', 'TL MIL', 'TL MIL (EV)', 'SIL', 'PIL', 'SL SIL')
  compFolderUID   string  (Optional) The folder where to save the simulated ExecutionRecords. If this property is not provided, the
                          ExecutionRecords are not saved.
}
```

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed at /ep/progress/{progressId}.

RESPONSE MODEL - application/json

```
{
  jobID  string  READ-ONLY
                 The ID of a job.
}
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

21.4 POST /ep/folders/regression-tests

Generate a test on a list of folders

Long running task Generates a Regression Test on a given list of folder UIDs for the specified comparison execution kind. Optionally, the folder where to save the simulated execution records can be provided. If this property is not provided, the execution records are not saved.

REQUEST

REQUEST BODY - application/json

```
{
  compMode*       string    Comparison execution mode (e.g. 'SL MIL', 'TL MIL', 'TL MIL (EV)', 'SIL', 'PIL', 'SL SIL')
  compFolderUID   string    (Optional) The folder where to save the simulated ExecutionRecords. If this property is not provided, the
                            ExecutionRecords are not saved.
  UIDs*           [string]  Folder UID list
}
```

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed at /ep/progress/{progressId}.

RESPONSE MODEL - application/json

{
  jobID `string` READ-ONLY
  The ID of a job.
}

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

`string`

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

`string`

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

`string`

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

`string`

21.5 GET /ep/regression-tests

Get all tests

Get all Regression Tests from active profile.

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

[{
Array of object:

uid	string	READ-ONLY The unique identifier (UID) of this object.
name	string	The name of the RegresionTest Test.
compMode	string	The comparison execution config type.
referenceFolderUIDs	[string]	Reference folder UIDs
comparisonFolderUID	string	Comparison folder UID
executionDate	string	Execution Date
verdictStatus	enum	ALLOWED:PASSED, FAILED, ERROR, FAILED_ACCEPTED PASSED, FAILED, ERROR, FAILED_ACCEPTED
verdictState	enum	ALLOWED:VALID, OUTDATED_TOLERANCE_UPDATE, OUTDATED_MISSING_EXECUTIONS Verdict State

```
  failed                integer  Number of failed comparisons.
  failedAccepted        integer  Number of failed accepted comparisons.
  passed                integer  Number of passed comparisons.
  error                 integer  Number of comparisons with error.
  total                 integer  Total number of comparisons.
  comparisons [{
  Array of object:  All comparisons.
    uid                           string  READ-ONLY
                                          The unique identifier (UID) of this object.
    name                          string  The name of Test Case / Stimuli Vector used in Comparison.
    verdictStatus                 enum    ALLOWED:PASSED, FAILED, ERROR, FAILED_ACCEPTED
                                          PASSED, FAILED, ERROR, FAILED_ACCEPTED
    referenceExecutionRecordUID   string  UID of reference execution record.
    comparisonExecutionRecordUID  string  UID of compared to execution record.
    comment                       string  Added comment for Comparison.
  }]
}]
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

22. REPORTS

Retrieve and export Reports.

22.1 GET /ep/reports/{report-uid}

Get a report

Get the report with the provided UID.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*report-uid	string	The UID of the report to be returned.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
{
  uid         string  READ-ONLY
                      The unique identifier (UID) of this object.
  reportName  string  Name of the report.
  reportType  string  Type of the report.
}
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

22.2 POST /ep/reports/{report-uid}

Export a report

Use this command export a report to a given location. The exported report corresponds to the given UID inside the command. New name can be set for file. If file exists, will be overwritten. Keep in mind to use only legal profile paths.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*report-uid	string	The UID of the report to be returned.

REQUEST BODY - application/json

```
{
  exportPath*  string  Path to export report
```

```
  newName      string  (Optional) New report name.
}
```

RESPONSE

STATUS CODE - 201: Exported Succesfully. Returns exported report location.

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

23. REQUIREMENT-BASED TEST CASES

Handle requirement-based Test Cases.

23.1 GET /ep/test-cases-rbt/{testcase-uid}

Get a test case

Get a requirement-based Test Case by providing its UID.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*testcase-uid	string	The UID of the requirement-based Test Case to be returned.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
{
  uid              string    READ-ONLY
                             The unique identifier (UID) of this object.
  name             string    The name of the RBTestCase.
  description      string    An optional description of the RBTestCase
  kind             string    The datatype or kind of the RBTestCase. Usually "tc" or "csv".
  length           integer   The length of the vector.
  draft            boolean   States wheter or not the RBTestCase is in Draft-Mode.
  lastModifiedDate string    The date of the last modification to the RBTestCase
  folderUID        string    The unique identifier of the folder the RBTestCase belongs to.
  scopeUID         string    The unique identifier of the scope the RBTestCase belongs to.
  requirementUIDs  [string]  The unique identifiers of the requirements belonging to the RBTestCase.
}
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

23.2 DELETE /ep/test-cases-rbt/{testcase-uid}

Delete a test case

Delete a requirement-based Test Case by providing its UID.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION

NAME	TYPE	DESCRIPTION
*testcase-uid	string	The UID of the requirement-based Test Case to be deleted.

RESPONSE

STATUS CODE - 200: OK

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

23.3 GET /ep/test-cases-rbt

Get all test cases

Get all requirement-based Test Cases.

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
[{
```

Array of object:

uid	string	READ-ONLY The unique identifier (UID) of this object.
name	string	The name of the RBTestCase.
description	string	An optional description of the RBTestCase
kind	string	The datatype or kind of the RBTestCase. Usually "tc" or "csv".
length	integer	The length of the vector.
draft	boolean	States wheter or not the RBTestCase is in Draft-Mode.
lastModifiedDate	string	The date of the last modification to the RBTestCase
folderUID	string	The unique identifier of the folder the RBTestCase belongs to.
scopeUID	string	The unique identifier of the scope the RBTestCase belongs to.
requirementUIDs	[string]	The unique identifiers of the requirements belonging to the RBTestCase.

```
}]
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

23.4 PUT /ep/test-cases-rbt

Import test cases

LONG RUNNING TASK Import multiple requirement-based Test Cases from external files by providing their Path and an Overwrite Policy.

REQUEST

REQUEST BODY - application/json

{

paths*	[string]	The paths to all profiles you would like to import.
folderUID	string	The UID of the folder you want to import into. Test-cases will be imported in the default folder if null.
overwritePolicy*	string	Decides what happens in case of duplicate names. Can be "EXTEND_NAME", "OVERWRITE" or "SKIP". Uppercase is required.
draft	boolean	Sets the Draft-Mode of the test cases. By default its value is false.
csvDelimiter	string	Relevant only for CSV export format. It can have one of the following values: "Semicolon", "Comma", "Colon", "Pipe". (Default value: "Semicolon").
importKind	string	The kind a file should be imported as. Either "tc" or "excel". "tc" by default

}

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed at /ep/progress/{progress-id}.

RESPONSE MODEL - application/json

{

jobID	string	READ-ONLY The ID of a job.

}

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

string

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

string

23.5 POST /ep/test-cases-rbt-export

Export test cases

Long running task Export single or multiple requirement-based Test Case(s) by providing the list of the test cases which will be exported, the export directory, the export format and a list of additional options for export.

REQUEST

REQUEST BODY - application/json

{

UIDs*	[string]	List with the UIDs of the elements which will be exported
exportDirectory*	string	Directory where to export the elements
exportFormat*	string	The format of the exported test cases. It can be: "Excel", "TC" or "CSV".Please keep the

specified format when you want to introduce a value.

```
  additionalOptions {
    csvDelimiter     string  Relevant only for CSV export format. It can have one of the following values: "Semicolon", "Comma", "Colon", "Pipe". (Default value: "Semicolon").
    singleFile       boolean Relevant only for CSV export format: false  - each vector will be exported in it's own file; true - all vectors will be exported in same file. (Default value: true)
    architectureUid  string  Relevant only for Excel export format. It specifies the UID of the architecture on which the interfaces of the vectors will be exported.
  }
}
```

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed at /ep/progress/{progress-id}.

RESPONSE MODEL - application/json

```
{
  jobID  string  READ-ONLY
                 The ID of a job.
}
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

23.6 GET /ep/folders/{folder-uid}/test-cases-rbt

Get a list of test cases from a folder

Get all requirement-based Test Cases from a certain Folder by providing its UID.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*folder-uid	string	The UID of the Folder containing the requirement based Test Cases.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

[{
Array of object:

uid	string	READ-ONLY The unique identifier (UID) of this object.
name	string	The name of the RBTestCase.
description	string	An optional description of the RBTestCase
kind	string	The datatype or kind of the RBTestCase. Usually "tc" or "csv".
length	integer	The length of the vector.
draft	boolean	States wheter or not the RBTestCase is in Draft-Mode.
lastModifiedDate	string	The date of the last modification to the RBTestCase
folderUID	string	The unique identifier of the folder the RBTestCase belongs to.
scopeUID	string	The unique identifier of the scope the RBTestCase belongs to.
requirementUIDs	[string]	The unique identifiers of the requirements belonging to the RBTestCase.

}]

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

string

23.7 GET /ep/scopes/{scope-uid}/test-cases-rbt

Get a list of test cases from a scope

Get all requirement-based Test Cases from a certain Scope by providing its UID.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*scope-uid	string	The UID of the Scope containing the requirement based Test Cases.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

[{
Array of object:

uid	string	READ-ONLY The unique identifier (UID) of this object.
name	string	The name of the RBTestCase.
description	string	An optional description of the RBTestCase
kind	string	The datatype or kind of the RBTestCase. Usually "tc" or "csv".
length	integer	The length of the vector.
draft	boolean	States wheter or not the RBTestCase is in Draft-Mode.
lastModifiedDate	string	The date of the last modification to the RBTestCase
folderUID	string	The unique identifier of the folder the RBTestCase belongs to.
scopeUID	string	The unique identifier of the scope the RBTestCase belongs to.
requirementUIDs	[string]	The unique identifiers of the requirements belonging to the RBTestCase.

}]

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

23.8 PUT /ep/requirements/test-cases-rbt

Unlink requirements from test cases

Use this command to unlink requirements from test cases. Only their uids must be specified.

REQUEST

REQUEST BODY - application/json

```
{
   requirementUIDs*  [string]  The uids of the requirements that need to be linked to test cases.
   testCaseUIDs*     [string]  The uids of the test cases that need to be linked to the requirements.
}
```

RESPONSE

STATUS CODE - 200: Requirements unlinked from test cases.

STATUS CODE - 400: Bad request.

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

23.9 POST /ep/requirements/test-cases-rbt

Link requirements to test cases

Use this command to link requirements to test cases. Only their uids must be specified.

REQUEST

REQUEST BODY - application/json

```
{
   requirementUIDs*  [string]  The uids of the requirements that need to be linked to test cases.
   testCaseUIDs*     [string]  The uids of the test cases that need to be linked to the requirements.
}
```

RESPONSE

STATUS CODE - 200: Requirements linked to test cases.

STATUS CODE - 400: Bad request.

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

23.10 GET /ep/requirements/{requirement-uid}/test-cases-rbt

Get a list of test cases linked to a requirement

Use this command to retrieve the test cases linked to a given requirement uid.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*requirement-uid	string	The uid of the requirement for which all linked test cases should be returned.

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
only-indirect-testcases	boolean	The flag used to specify whether to retrieve only test cases which are indirectly linked to a given requirement. This flag has an effect only on non-leaves requirements. If no value is specified, this flag will be set to false.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
[{
```

Array of object:

uid	string	READ-ONLY The unique identifier (UID) of this object.
name	string	The name of the RBTestCase.
description	string	An optional description of the RBTestCase
kind	string	The datatype or kind of the RBTestCase. Usually "tc" or "csv".
length	integer	The length of the vector.
draft	boolean	States wheter or not the RBTestCase is in Draft-Mode.
lastModifiedDate	string	The date of the last modification to the RBTestCase
folderUID	string	The unique identifier of the folder the RBTestCase belongs to.
scopeUID	string	The unique identifier of the scope the RBTestCase belongs to.
requirementUIDs	[string]	The unique identifiers of the requirements belonging to the RBTestCase.

```
}]
```

STATUS CODE - 400: Bad request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

---

24. REQUIREMENT-BASED TEST EXECUTION

Executes requirement-based Testing.

24.1 POST /ep/folders/{folder-uid}/test-execution-rbt

Execute all test cases from a folder

Long running task Executes a requirement-based Test on all Test Cases from a given folder for the specified execution kinds. Optionally, also the model coverage report can be generated or the test execution can be forced to not re-use any previous results.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*folder-uid	string	The folder UID for which the requirement-based Test is executed.

REQUEST BODY - application/json

```
{
  execConfigNames*              [string]  List of execution kinds
  generateModelCoverageReport   boolean   Specify (optional) if the model coverage report should be generated
  forceExecute                  boolean   Specify (optional) if the test execution should be forced (all previous results will be discarded).
}
```

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed at /ep/progress/{progressId}.

RESPONSE MODEL - application/json

```
{
  jobID  string  READ-ONLY
                 The ID of a job.
}
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

24.2 POST /ep/scopes/{scope-uid}/test-execution-rbt

Execute all test cases from a scope

Long running task Executes a requirement-based Test on all Test Cases from a given scope for the specified execution kinds. Optionally, also the model coverage report can be generated or the test execution can be forced to not re-use any previous results.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*scope-uid	string	The scope UID for which the requirement-based Test is executed.

REQUEST BODY - application/json

```
{
  execConfigNames*              [string]  List of execution kinds
  generateModelCoverageReport   boolean   Specify (optional) if the model coverage report should be generated
  forceExecute                  boolean   Specify (optional) if the test execution should be forced (all previous results will be discarded).
}
```

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed at /ep/progress/{progressId}.

RESPONSE MODEL - application/json

```
{
  jobID  string  READ-ONLY
                 The ID of a job.
}
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

24.3 POST /ep/folders/test-execution-rbt

Execute all test cases from a list of folders

Long running task Executes a requirement-based Test on all Test Cases from a given list of folders for the specified execution kinds. Optionally, also the model coverage report can be generated or the test execution can be forced to not re-use any previous results.

REQUEST

REQUEST BODY - application/json

```
{
  UIDs*                          [string]                  A list with unique identifiers.
  data {
    execConfigNames*             [string]  List of execution kinds
    generateModelCoverageReport  boolean   Specify (optional) if the model coverage report should be generated
    forceExecute                 boolean   Specify (optional) if the test execution should be forced (all previous
                                           results will be discarded).
  }
}
```

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed at /ep/progress/{progressId}.

RESPONSE MODEL - application/json

```
{
  jobID  string  READ-ONLY
                 The ID of a job.
}
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

24.4 POST /ep/scopes/test-execution-rbt

Execute all test cases from a list of scopes

Long running task Executes a requirement-based Test on all Test Cases from a given list of scopes for the specified execution kinds. Optionally, also the model coverage report can be generated or the test execution can be forced to not re-use any previous results.

REQUEST

REQUEST BODY - application/json

```
{
  UIDs*                          [string]                  A list with unique identifiers.
  data {
    execConfigNames*             [string]  List of execution kinds
    generateModelCoverageReport  boolean   Specify (optional) if the model coverage report should be generated
```

```
    forceExecute                boolean   Specify (optional) if the test execution should be forced (all previous
                                          results will be discarded).
  }
}
```

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed at /ep/progress/{progressId}.

RESPONSE MODEL - application/json

```
{
  jobID  string  READ-ONLY
                 The ID of a job.
}
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

24.5 POST /ep/test-cases-rbt/test-execution-rbt

Execute a list of test cases

Long running task Executes a requirement-based Test on a list of requirement-based Test Cases for the specified execution kinds. Optionally, also the test execution can be forced to not re-use any previous results.

REQUEST

REQUEST BODY - application/json

```
{
  UIDs*                  [string]                A list with unique identifiers.
  data {
    execConfigNames*  [string]   List of execution kinds
    forceExecute      boolean    Specify (optional) if the test execution should be forced (all previous results will be
                                 discarded).
  }
}
```

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed at /ep/progress/{progressId}.

RESPONSE MODEL - application/json

```
{
  jobID string READ-ONLY
        The ID of a job.
}
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

24.6 POST /ep/requirements-sources/test-execution-rbt

Execute all test cases linked to a list of requirements sources

Long running task Executes a requirement-based Test on all Test Cases linked to the requirements of the given requirements sources for the specified execution kinds. Optionally, also the model coverage report can be generated or the test execution can be forced to not re-use any previous results.

REQUEST

REQUEST BODY - application/json

```
{
  UIDs*                        [string]   A list with unique identifiers.
  data {
    execConfigNames*           [string]   List of execution kinds
    generateModelCoverageReport boolean   Specify (optional) if the model coverage report should be generated
    forceExecute               boolean    Specify (optional) if the test execution should be forced (all previous
                                          results will be discarded).
  }
}
```

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed at /ep/progress/{progressId}.

RESPONSE MODEL - application/json

```
{
  jobID string READ-ONLY
        The ID of a job.
}
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

string

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

string

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

string

24.7 POST /ep/requirements-sources/{requirements-source-uid}/test-execution-rbt

Execute all test cases linked to a requirements source

Long running task Executes a requirement-based Test on all Test Cases linked to the requirements of the given requirements source for the specified execution kinds. Optionally, also the model coverage report can be generated or the test execution can be forced to not re-use any previous results.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*requirements-source-uid	string	The requirements source UID for which the RBT is executed.

REQUEST BODY - application/json

```
{
  execConfigNames*              [string]  List of execution kinds
  generateModelCoverageReport   boolean   Specify (optional) if the model coverage report should be generated
  forceExecute                  boolean   Specify (optional) if the test execution should be forced (all previous results will
                                          be discarded).
}
```

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed at /ep/progress/{progressId}.

RESPONSE MODEL - application/json

```
{
  jobID  string  READ-ONLY
                 The ID of a job.
}
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

string

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

string

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

string

24.8 POST /ep/test-cases-rbt/{testcase-uid}/test-execution-rbt

Execute a test case

Long running task Executes a requirement-based Test on a requirement-based Test Case for the specified execution kinds. Optionally, also the test execution can be forced to not re-use any previous results.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*testcase-uid	string	The requirement-based Test Case UID for which the RBT is executed.

REQUEST BODY - application/json

```
{
  execConfigNames*  [string]  List of execution kinds
  forceExecute      boolean   Specify (optional) if the test execution should be forced (all previous results will be discarded).
}
```

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed at /ep/progress/{progressId}.

RESPONSE MODEL - application/json

```
{
  jobID  string  READ-ONLY
                 The ID of a job.
}
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

string

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

string

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

25. REQUIREMENT-BASED TEST EXECUTION REPORTS

Creates requirement-based Test Execution Reports.

25.1 POST /ep/scopes/test-execution-reports-rbt

Create a report on a list of scopes

Create a requirement-based Test Execution Report on a given list of scopes.

REQUEST

REQUEST BODY - application/json

```
{
  UIDs*           [string]  A list with unique identifiers.
  execConfigName* string    Execution kind
}
```

RESPONSE

STATUS CODE - 201: Created

RESPONSE MODEL - application/json

```
{
  uid         string  READ-ONLY
                      The unique identifier (UID) of this object.
  reportName  string  Name of the report.
  reportType  string  Type of the report.
}
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

25.2 POST /ep/folders/test-execution-reports-rbt

Create a report on a list of folders

Create a requirement-based Test Execution Report on a given list of folders.

REQUEST

REQUEST BODY - application/json

```
{
  UIDs*           [string]  A list with unique identifiers.
  execConfigName* string    Execution kind
}
```

RESPONSE

STATUS CODE - 201: Created

RESPONSE MODEL - application/json

```
{
  uid          string  READ-ONLY
                       The unique identifier (UID) of this object.
  reportName   string  Name of the report.
  reportType   string  Type of the report.
}
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

25.3 POST /ep/requirements-sources/{requirements-source-uid}/test-execution-reports-rbt

Create a report on a requirements source

Create a requirement-based Test Execution Report on a given requirements source.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*requirements-source-uid	string	The requirements source UID for which the Test Execution Report is created.

REQUEST BODY - application/json

```
{
  execConfigName*  string  Execution kind
}
```

RESPONSE

STATUS CODE - 201: Created

RESPONSE MODEL - application/json

```
{
  uid          string  READ-ONLY
                       The unique identifier (UID) of this object.
  reportName   string  Name of the report.
  reportType   string  Type of the report.
}
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

25.4 POST /ep/requirements-sources/test-execution-reports-rbt

Create a report on a list of requirements sources

Create a requirement-based Test Execution Report on a given list of requirements sources.

REQUEST

REQUEST BODY - application/json

```
{
  UIDs*            [string]  A list with unique identifiers.
  execConfigName*  string    Execution kind
}
```

RESPONSE

STATUS CODE - 201: Created

RESPONSE MODEL - application/json

```
{
  uid         string  READ-ONLY
                      The unique identifier (UID) of this object.
  reportName  string  Name of the report.
  reportType  string  Type of the report.
}
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

25.5 GET /ep/test-execution-reports-rbt

Get all reports

Retrieve all the requirement-based Test Execution reports from the profile.

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
[{
Array of object:
  uid          string  READ-ONLY
                       The unique identifier (UID) of this object.
  reportName   string  Name of the report.
  reportType   string  Type of the report.
}]
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

25.6 POST /ep/scopes/{scope-uid}/test-execution-reports-rbt

Create a report on a scope

Create a requirement-based Test Execution Report on a given scope.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*scope-uid	string	The Scope UID for which the Test Execution Report is created.

REQUEST BODY - application/json

```
{
  execConfigName*  string  Execution kind
}
```

RESPONSE

STATUS CODE - 201: Created

RESPONSE MODEL - application/json

```
{
  uid          string  READ-ONLY
                       The unique identifier (UID) of this object.
  reportName   string  Name of the report.
  reportType   string  Type of the report.
}
```

STATUS CODE - 400: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

25.7 POST /ep/folders/{folder-uid}/test-execution-reports-rbt

Create a report on a folder

Create a requirement-based Test Execution Report on a given folder.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*folder-uid	string	The folder UID for which the Test Execution Report is created.

REQUEST BODY - application/json

```
{
  execConfigName* string  Execution kind
}
```

RESPONSE

STATUS CODE - 201: Created

RESPONSE MODEL - application/json

```
{
  uid         string  READ-ONLY
                      The unique identifier (UID) of this object.
  reportName  string  Name of the report.
  reportType  string  Type of the report.
}
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

---

26. REQUIREMENTS

Retrieve Requirements, Linked Requirements, or Requirement Sources.

26.1 GET /ep/requirements/{requirement-source-id}

Get all requirements of a requirement source

Get the requirements of the given requirement source id.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*requirement-source-id	string	The UID of the requirement source.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
[{
```
Array of object:

```
  uid*                                  string      The universally unique identifier.
  identifier*                           string      The requirement identifier (e.g. a chapter
                                                    number within DOORS).
  isRemoved                             boolean     Value meaning whether this requirement has
                                                    been removed within the requirement
                                                    management tool.

  additionalAttributes {
```
Map containing all additional attributes.

```
  }
  scopeId*                              string      The scope id this requirement is directly
                                                    linked to.
  description*                          string      The requirement description.
  name*                                 string      The requirement name.
  dateOfLastUpdate*                     string      The requirement last update date.
  requirementSource* {
    kind*                                   string  The kind of this requirement source.
                                                    The kind identifies this source to be
                                                    from a specific requirement
                                                    management tool.

    settings* [{
```
Array of object: A list with requirement settings.For EXCEL requirement kind, the following keys can be: name_attr_value, desc_attr_value, additional_attr, modification_date, excel_file_path, projectName_attr (or excel_sheet_name), excel_id_attr, excel_start_row, excel_parent_id.For PTC requirement kind, the following keys can be: name_attr_value, desc_attr_value, additional_attr, modification_date, ptc_user, ptc_pw, projectName_attr, ptc_host, ptc_port, mapping_file_location.For DOORS requirement kind, the following keys can be: name_attr_value, desc_attr_value, additional_attr, modification_date, projectName_attr, doors_module_qualifier, doors_baseline_major, doors_baseline_minor, doors_baseline_suffix.

```
      key*    string  For EXCEL requirement kind, the following keys are mandatory: excel_file_path, projectName_attr,
                      excel_id_attr. The optional setting keys are: excel_start_row, excel_parent_id.For PTC requirement kind,
                      the following keys are mandatory: ptc_user, ptc_pw, projectName_attr, ptc_host, ptc_port,
                      mapping_file_location.For DOORS requirement kind, the following keys are mandatory: projectName_attr,
                      doors_module_qualifier. The optional setting keys are: doors_baseline_major, doors_baseline_minor,
                      doors_baseline_suffix.
      value*  string  Setting value
    }]
    definedAdditionalAttributes         [string]    A list with additional attributes.
    externalUUID*                       string      The unique ID identifying the external
                                                    requirement source.
```

name* string The requirement source name.
uid* string The universally unique identifier of this requirement source.
}
}]

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

string

26.2 GET /ep/scopes/{scope-id}/linked-requirements

Get all requirements for a scope

Get the linked requirements of this given scope id.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*scope-id	string	The UID of the scope id.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

[{
Array of object:

uid* string The universally unique identifier.
identifier* string The requirement identifier (e.g. a chapter number within DOORS).
isRemoved boolean Value meaning whether this requirement has been removed within the requirement management tool.
additionalAttributes {
Map containing all additional attributes.
}
scopeId* string The scope id this requirement is directly linked to.
description* string The requirement description.
name* string The requirement name.
dateOfLastUpdate* string The requirement last update date.
requirementSource* {
kind* string The kind of this requirement source. The kind identifies this source to be from a specific requirement management tool.
settings* [{
Array of object: A list with requirement settings.For EXCEL requirement kind, the following keys can be: name_attr_value, desc_attr_value, additional_attr, modification_date, excel_file_path, projectName_attr (or excel_sheet_name), excel_id_attr, excel_start_row, excel_parent_id.For PTC requirement kind, the following keys can be: name_attr_value, desc_attr_value, additional_attr, modification_date, ptc_user, ptc_pw, projectName_attr, ptc_host, ptc_port, mapping_file_location.For DOORS requirement kind, the following keys can be: name_attr_value, desc_attr_value, additional_attr, modification_date, projectName_attr, doors_module_qualifier, doors_baseline_major, doors_baseline_minor, doors_baseline_suffix.

key* string For EXCEL requirement kind, the following keys are mandatory: excel_file_path, projectName_attr, excel_id_attr. The optional setting keys are: excel_start_row, excel_parent_id.For PTC requirement kind, the following keys are mandatory: ptc_user, ptc_pw, projectName_attr, ptc_host, ptc_port, mapping_file_location.For DOORS requirement kind, the following keys are mandatory: projectName_attr, doors_module_qualifier. The optional setting keys are: doors_baseline_major, doors_baseline_minor, doors_baseline_suffix.

value* string Setting value

}]

definedAdditionalAttributes [string] A list with additional attributes.

externalUUID* string The unique ID identifying the external requirement source.

name* string The requirement source name.

uid* string The universally unique identifier of this requirement source.

}

}]

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

string

26.3 GET /ep/requirements-sources

Get all requirement sources

Get all requirement sources found on the opened profile.

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

[{

Array of object:

kind* string The kind of this requirement source. The kind identifies this source to be from a specific requirement management tool.

settings* [{

Array of object: A list with requirement settings.For EXCEL requirement kind, the following keys can be: name_attr_value, desc_attr_value, additional_attr, modification_date, excel_file_path, projectName_attr (or excel_sheet_name), excel_id_attr, excel_start_row, excel_parent_id.For PTC requirement kind, the following keys can be: name_attr_value, desc_attr_value, additional_attr, modification_date, ptc_user, ptc_pw, projectName_attr, ptc_host, ptc_port, mapping_file_location.For DOORS requirement kind, the following keys can be: name_attr_value, desc_attr_value, additional_attr, modification_date, projectName_attr, doors_module_qualifier, doors_baseline_major, doors_baseline_minor, doors_baseline_suffix.

key* string For EXCEL requirement kind, the following keys are mandatory: excel_file_path, projectName_attr, excel_id_attr. The optional setting keys are: excel_start_row, excel_parent_id.For PTC requirement kind, the following keys are mandatory: ptc_user, ptc_pw, projectName_attr, ptc_host, ptc_port, mapping_file_location.For DOORS requirement kind, the following keys are mandatory: projectName_attr, doors_module_qualifier. The optional setting keys are: doors_baseline_major, doors_baseline_minor, doors_baseline_suffix.

value* string Setting value

}]

definedAdditionalAttributes [string] A list with additional attributes.

externalUUID* string The unique ID identifying the external requirement source.

name* string The requirement source name.

uid*	string	The universally unique identifier of this requirement source.

}]

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

27. REQUIREMENTS IMPORT

Import Requirements.

27.1 GET /ep/requirements-import/doors

Get the DOORS™ configuration template

Get the configuration with default settings for importing different kinds of requirements.

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
{
  kind*                    enum          ALLOWED:EXCEL, PTC, DOORS
                                         The kind of imported requirements. Allowed types: EXCEL, PTC, or DOORS.
  nameAttribute            string        The name of imported requirements. Required for DOORS and EXCEL import.
  descriptionAttribute     string        The description of imported requirements. Required for DOORS and EXCEL import.
  additionalAttributes     [string]      A list with additional attributes.
  settings* [{
  Array of object:  A list with requirement settings.For EXCEL requirement kind, the following keys can be: name_attr_value, desc_attr_value, additional_attr, modification_date, excel_file_path, projectName_attr (or excel_sheet_name), excel_id_attr, excel_start_row, excel_parent_id.For PTC requirement kind, the following keys can be: name_attr_value, desc_attr_value, additional_attr, modification_date, ptc_user, ptc_pw, projectName_attr, ptc_host, ptc_port, mapping_file_location.For DOORS requirement kind, the following keys can be: name_attr_value, desc_attr_value, additional_attr, modification_date, projectName_attr, doors_module_qualifier, doors_baseline_major, doors_baseline_minor, doors_baseline_suffix.
    key*     string   For EXCEL requirement kind, the following keys are mandatory: excel_file_path, projectName_attr, excel_id_attr. The optional setting keys are: excel_start_row, excel_parent_id.For PTC requirement kind, the following keys are mandatory: ptc_user, ptc_pw, projectName_attr, ptc_host, ptc_port, mapping_file_location.For DOORS requirement kind, the following keys are mandatory: projectName_attr, doors_module_qualifier. The optional setting keys are: doors_baseline_major, doors_baseline_minor, doors_baseline_suffix.
    value*   string   Setting value
  }]
}
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

27.2 GET /ep/requirements-import/ptc

Get the PTC™ configuration template

Get the configuration with default settings for importing different kinds of requirements.

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

{

kind*	enum	**ALLOWED:** EXCEL, PTC, DOORS The kind of imported requirements. Allowed types: EXCEL, PTC, or DOORS.
nameAttribute	string	The name of imported requirements. Required for DOORS and EXCEL import.
descriptionAttribute	string	The description of imported requirements. Required for DOORS and EXCEL import.
additionalAttributes	[string]	A list with additional attributes.

settings* [{

Array of object: A list with requirement settings.For EXCEL requirement kind, the following keys can be: name_attr_value, desc_attr_value, additional_attr, modification_date, excel_file_path, projectName_attr (or excel_sheet_name), excel_id_attr, excel_start_row, excel_parent_id.For PTC requirement kind, the following keys can be: name_attr_value, desc_attr_value, additional_attr, modification_date, ptc_user, ptc_pw, projectName_attr, ptc_host, ptc_port, mapping_file_location.For DOORS requirement kind, the following keys can be: name_attr_value, desc_attr_value, additional_attr, modification_date, projectName_attr, doors_module_qualifier, doors_baseline_major, doors_baseline_minor, doors_baseline_suffix.

key*	string	For EXCEL requirement kind, the following keys are mandatory: excel_file_path, projectName_attr, excel_id_attr. The optional setting keys are: excel_start_row, excel_parent_id.For PTC requirement kind, the following keys are mandatory: ptc_user, ptc_pw, projectName_attr, ptc_host, ptc_port, mapping_file_location.For DOORS requirement kind, the following keys are mandatory: projectName_attr, doors_module_qualifier. The optional setting keys are: doors_baseline_major, doors_baseline_minor, doors_baseline_suffix.
value*	string	Setting value

}]

}

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

string

27.3 GET /ep/requirements-import/excel

Get the Excel™ configuration template

Get the configuration with default settings for importing different kinds of requirements.

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

{

kind*	enum	**ALLOWED:** EXCEL, PTC, DOORS The kind of imported requirements. Allowed types: EXCEL, PTC, or DOORS.
nameAttribute	string	The name of imported requirements. Required for DOORS and EXCEL import.
descriptionAttribute	string	The description of imported requirements. Required for DOORS and EXCEL import.
additionalAttributes	[string]	A list with additional attributes.

settings* [{

Array of object: A list with requirement settings.For EXCEL requirement kind, the following keys can be: name_attr_value, desc_attr_value, additional_attr, modification_date, excel_file_path, projectName_attr (or excel_sheet_name), excel_id_attr, excel_start_row, excel_parent_id.For PTC requirement kind, the following keys can be: name_attr_value, desc_attr_value, additional_attr, modification_date, ptc_user, ptc_pw, projectName_attr, ptc_host, ptc_port, mapping_file_location.For DOORS requirement kind, the following keys can be: name_attr_value, desc_attr_value, additional_attr, modification_date, projectName_attr, doors_module_qualifier, doors_baseline_major, doors_baseline_minor, doors_baseline_suffix.

key*	string	For EXCEL requirement kind, the following keys are mandatory: excel_file_path, projectName_attr, excel_id_attr. The optional setting keys are: excel_start_row, excel_parent_id.For PTC requirement kind, the following keys are mandatory: ptc_user, ptc_pw, projectName_attr, ptc_host, ptc_port, mapping_file_location.For DOORS requirement

kind, the following keys are mandatory: projectName_attr, doors_module_qualifier. The optional setting keys are: doors_baseline_major, doors_baseline_minor, doors_baseline_suffix.

value* string Setting value

}]

}

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

string

27.4 POST /ep/requirements-import

Import requirements

Import requirements by specifying the kind of requirements.

REQUEST

REQUEST BODY - application/json

{

kind* enum ALLOWED:EXCEL, PTC, DOORS
The kind of imported requirements. Allowed types: EXCEL, PTC, or DOORS.

nameAttribute string The name of imported requirements. Required for DOORS and EXCEL import.

descriptionAttribute string The description of imported requirements. Required for DOORS and EXCEL import.

additionalAttributes [string] A list with additional attributes.

settings* [{
Array of object: A list with requirement settings.For EXCEL requirement kind, the following keys can be: name_attr_value, desc_attr_value, additional_attr, modification_date, excel_file_path, projectName_attr (or excel_sheet_name), excel_id_attr, excel_start_row, excel_parent_id.For PTC requirement kind, the following keys can be: name_attr_value, desc_attr_value, additional_attr, modification_date, ptc_user, ptc_pw, projectName_attr, ptc_host, ptc_port, mapping_file_location.For DOORS requirement kind, the following keys can be: name_attr_value, desc_attr_value, additional_attr, modification_date, projectName_attr, doors_module_qualifier, doors_baseline_major, doors_baseline_minor, doors_baseline_suffix.

key* string For EXCEL requirement kind, the following keys are mandatory: excel_file_path, projectName_attr, excel_id_attr. The optional setting keys are: excel_start_row, excel_parent_id.For PTC requirement kind, the following keys are mandatory: ptc_user, ptc_pw, projectName_attr, ptc_host, ptc_port, mapping_file_location.For DOORS requirement kind, the following keys are mandatory: projectName_attr, doors_module_qualifier. The optional setting keys are: doors_baseline_major, doors_baseline_minor, doors_baseline_suffix.

value* string Setting value

}]

}

RESPONSE

STATUS CODE - 201: Requirements imported.

STATUS CODE - 400: Import failed.

RESPONSE MODEL - text/plain

string

STATUS CODE - 404: Required file not found.

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

string

---

28. SCOPES

Handle Scopes.

28.1 GET /ep/architectures/{architecture-uid}/scopes

Get all scopes from an architecture

Get the scopes from an architecture by a query which filters by path and top-level.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*architecture-uid	string	The UID of the architecture to get the scope from.

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
path	string	The path of the scope you would like to search for. If null, then all scopes from the architecture will be returned.
top-level	boolean	Specifies, if only top level scopes shall be returned. TRUE will return only top level scopes. FALSE will return all scopes, including the top level. If not specified, then the default value is FALSE.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
[{
Array of object:  The entry scope to use for this run.
   uid           string   The unique identifier (UID) of this object.
   name          string   The scope name.
   topLevel      boolean  TRUE if scope is a toplevel scope.
   kind          enum     ALLOWED:SUT, DUMMY, ENVIRONMENT, HIDDEN_INTERNAL, VIRTUAL
                          Scope kind.
   path          string   Scope path.
   architecture  string   The corresponding architecture of the scope.
   sampleTime {
   The sample time of the scope.
      uid      string  The unique identifier (UID) of this object.
      seconds  string  The sample time as a value given in seconds.
   }
}]
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

28.2 GET /ep/scopes/{scope-uid}

Get a scope

Get a specific scope by providing its UID.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*scope-uid	string	The UID of the scope to be returned.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
{
The entry scope to use for this run.
  uid           string   The unique identifier (UID) of this object.
  name          string   The scope name.
  topLevel      boolean  TRUE if scope is a toplevel scope.
  kind          enum     ALLOWED:SUT, DUMMY, ENVIRONMENT, HIDDEN_INTERNAL, VIRTUAL
                         Scope kind.
  path          string   Scope path.
  architecture  string   The corresponding architecture of the scope.
  sampleTime {
  The sample time of the scope.
    uid      string  The unique identifier (UID) of this object.
    seconds  string  The sample time as a value given in seconds.
  }
}
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

28.3 GET /ep/scopes

Get a list of scopes

Get a list of scopes by a query which filters by path and top-level.

REQUEST

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
path	string	Enter the path of the scope you would like to search for. If null, then all scopes will be returned.

NAME	TYPE	DESCRIPTION
top-level	boolean	Specifies, if only top level scopes shall be returned. TRUE will return only top level scopes. FALSE will return all scopes, including the top level. If not specified, then the default value is FALSE.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
[{
```
Array of object: The entry scope to use for this run.

uid	string	The unique identifier (UID) of this object.
name	string	The scope name.
topLevel	boolean	TRUE if scope is a toplevel scope.
kind	enum	ALLOWED:SUT, DUMMY, ENVIRONMENT, HIDDEN_INTERNAL, VIRTUAL Scope kind.
path	string	Scope path.
architecture	string	The corresponding architecture of the scope.

```
  sampleTime {
```
The sample time of the scope.

uid	string	The unique identifier (UID) of this object.
seconds	string	The sample time as a value given in seconds.

```
  }
}]
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

29. SIGNALS

Handle signals' operations.

29.1 GET /ep/scopes/{scope-uid}/signals

Get all signals from a scope

Get all signals from the given scope

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*scope-uid	string	The scope for which to get the signals.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

[{
Array of object:

uid string READ-ONLY
The unique identifier (UID) of this object.

identifier string READ-ONLY
The signal identifier.

}]

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

string

30. STIMULI VECTORS

Handle stimuli vectors.

30.1 GET /ep/stimuli-vectors

Get all stimuli vectors

Get all stimuli vectors.

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
[{
```

Array of object:

uid	string	READ-ONLY The unique identifier (UID) of this object.
name	string	The name of the StimuliVector.
length	integer	The length of the vector.
folderUID	string	The unique identifier of the folder the StimuliVector belongs to.
scopeUID	string	The unique identifier of the scope the StimuliVector belongs to.

```
}]
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

30.2 PUT /ep/stimuli-vectors

Import stimuli vectors

Long running task Import multiple stimuli vectors from external files into a Folder by providing their path and the Folders UID. Can either overwrite or skip stimuli vectors that are already imported.

REQUEST

REQUEST BODY - application/json

```
{
```

paths*	[string]	The paths to all stimuli vectors you'd like to import.
format	string	The format of the selected files. Currently supported: "excel", "csv". (Default value: "excel")
vectorKind	string	The stimuli vector type. Possible: "TC" or "Excel". Capitalization is required. (Default value: "TC")
folderUID	string	The UID of the folder you want to import into.
delimiter	string	The CSV file delimiter, can be: "Semicolon", "Comma", "Colon", "Pipe". Capitalization is required. (Default value: "Semicolon")
overwritePolicy*	string	Decides what happens in case of duplicate names. Can be "OVERWRITE" or "SKIP".

```
}
```

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed at /ep/progress/{progress-id}.

RESPONSE MODEL - application/json

```
{
  jobID  string  READ-ONLY
                 The ID of a job.
}
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

30.3 POST /ep/stimuli-vectors-export

Export Stimuli Vectors

Long running task Export single or multiple Stimuli Vector(s) by providing the list of the stimuli vectors which will be exported, the export directory, the export format and a list of additional options for export.

REQUEST

REQUEST BODY - application/json

```
{
  UIDs*              [string]   List with the UIDs of the elements which will be exported
  exportDirectory*   string     Directory where to export the elements
  exportFormat*      string     The format of the exported stimuli vectors. It can be: "Excel" or "CSV". Please keep the
                                specified format when you want to introduce a value.
  additionalOptions {
    csvDelimiter     string     Relevant only for CSV export format. It can have one of the following values: "Semicolon", "Comma",
                                "Colon", "Pipe". (Default value: "Semicolon").
    singleFile       boolean    Relevant only for CSV export format: false  - each vector will be exported in it's own file; true - all
                                vectors will be exported in same file. (Default value: true)
    architectureUid  string     Relevant only for Excel export format. It specifies the UID of the architecture on which the
                                interfaces of the vectors will be exported.
  }
}
```

RESPONSE

STATUS CODE - 202: Long running operation started. The status of the operation can be reviewed at /ep/progress/{progress-id}.

RESPONSE MODEL - application/json

```
{
  jobID  string  READ-ONLY
                 The ID of a job.
}
```

STATUS CODE - 400: Bad Request

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 403: Forbidden

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

30.4 GET /ep/stimuli-vectors/{stimuli-vector-uid}

Get a stimuli vector

Get a specific stimuli vector by UID.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*stimuli-vector-uid	string	The UID of the stimuli vector to be returned.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
{
  uid         string   READ-ONLY
                       The unique identifier (UID) of this object.
  name        string   The name of the StimuliVector.
  length      integer  The length of the vector.
  folderUID   string   The unique identifier of the folder the StimuliVector belongs to.
  scopeUID    string   The unique identifier of the scope the StimuliVector belongs to.
}
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

30.5 DELETE /ep/stimuli-vectors/{stimuli-vector-uid}

Delete a stimuli vector

Delete a stimuli vector by its UID.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*stimuli-vector-uid	string	The UID of the stimuli vector to be deleted.

RESPONSE

STATUS CODE - 200: OK

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

30.6 GET /ep/folders/{folder-uid}/stimuli-vectors

Get all stimuli vectors from a folder

Get all stimuli vectors from the specified folder.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*folder-uid	string	The UID of the folder from which to get stimuli vectors.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
[{
Array of object:
  uid         string   READ-ONLY
                       The unique identifier (UID) of this object.
  name        string   The name of the StimuliVector.
  length      integer  The length of the vector.
  folderUID   string   The unique identifier of the folder the StimuliVector belongs to.
  scopeUID    string   The unique identifier of the scope the StimuliVector belongs to.
}]
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

30.7 GET /ep/scopes/{scope-uid}/stimuli-vectors

Get all stimuli vectors from a scope

Get all stimuli vectors from the specified scope.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*scope-uid	string	The UID of the scope from which to get stimuli vectors.

RESPONSE

STATUS CODE - 200: OK

RESPONSE MODEL - application/json

```
[{
```

Array of object:

uid	string	READ-ONLY The unique identifier (UID) of this object.
name	string	The name of the StimuliVector.
length	integer	The length of the vector.
folderUID	string	The unique identifier of the folder the StimuliVector belongs to.
scopeUID	string	The unique identifier of the scope the StimuliVector belongs to.

```
}]
```

STATUS CODE - 404: Not found

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error

RESPONSE MODEL - text/plain

```
string
```

31. TEST

31.1 GET /ep/test

Test the connection to the REST server

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: Request successfully sent.

STATUS CODE - 500: Internal server error.

---

32. TOLERANCES

Import, reset and retrieve global and local tolerances.

32.1 PUT /ep/profiles/global-tolerances

Import the global tolerances

Use this command to import the global tolerances.

REQUEST

REQUEST BODY - application/json

```
{
  path*               string  The path to the xml file from which to import tolerances.
  toleranceUseCase*   string  The use case of the tolerances for which tolerances should be applied. Allowed values are RBT and
                              B2B.
}
```

RESPONSE

STATUS CODE - 200: Global tolerances imported.

STATUS CODE - 400: Bad request.

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 406: Not acceptable.

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

32.2 DELETE /ep/profiles/global-tolerances

Reset the global tolerances

Use this command to reset the global tolerances for the profile.

REQUEST

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
*kind	string	The use case kind. Can be either RBT or B2B.

RESPONSE

STATUS CODE - 200: Global tolerances reset.

STATUS CODE - 400: Bad request.

RESPONSE MODEL - text/plain

string

STATUS CODE - 406: Not acceptable.

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

string

32.3 GET /ep/scopes/{scope-id}/global-tolerances

Get the global tolerances

Use this command to retrieve the global tolerances. A valid scope uid and use case must be specified. The lead-lag-unit can be either Seconds or Steps.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*scope-id	string	The scope from which to retrieve the global tolerances.

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
*kind	string	The use case kind. Can be either RBT or B2B.
lead-lag-unit	string	For existing tolerances, lead and lag values should be displayed as either Seconds or Steps.

RESPONSE

STATUS CODE - 200: Global tolerances retrieved.

RESPONSE MODEL - application/json

[{
Array of object:

uid	string	READ-ONLY The unique identifier (UID) of this object.
name	string	
lead	string	
lag	string	
absTol	string	
relTol	string	
kind	string	

}]

STATUS CODE - 400: Bad request.

RESPONSE MODEL - text/plain

string

STATUS CODE - 404: Not found.

RESPONSE MODEL - text/plain

string

STATUS CODE - 406: Not acceptable.

RESPONSE MODEL - text/plain

string

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

string

32.4 GET /ep/test-cases/{test-case-id}/local-tolerances

Get the local tolerances

Use this command to retrieve the local tolerances. A valid test case uid must be specified. The lead-lag-unit can be either Seconds or Steps. Setting the create-if-missing to true will create local tolerances from the global tolerances.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*test-case-id	string	The UID of the test case for which local tolerances will be retrieved.

QUERY PARAMETERS

NAME	TYPE	DESCRIPTION
lead-lag-unit	string	For existing tolerances, lead and lag values should be displayed as either Seconds or Steps.
create-if-missing	boolean	If true, the local tolerances will be created if they do not exist yet. Default is: false.

RESPONSE

STATUS CODE - 200: Local tolerances retrieved.

RESPONSE MODEL - application/json

[{
Array of object:

uid string READ-ONLY
The unique identifier (UID) of this object.
name string
lead string
lag string
absTol string
relTol string
kind string
}]

STATUS CODE - 400: Bad request.

RESPONSE MODEL - text/plain

string

STATUS CODE - 404: Test case not found.

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 406: Not acceptable.

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

32.5 PUT /ep/test-cases/{test-case-id}/local-tolerances

Import the local tolerances

Import local tolerances for a given test case.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*test-case-id	string	The UID of the test case for which local tolerances will be imported.

REQUEST BODY - application/json

```
{
  path* string The path to the xml file from which to import tolerances.
}
```

RESPONSE

STATUS CODE - 200: Local tolerances imported.

STATUS CODE - 400: Bad request.

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Test case not found.

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 406: Not acceptable.

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```

32.6 DELETE /ep/test-cases/{test-case-id}/local-tolerances

Reset the local tolerances

Remove the local tolerances for a test case.

REQUEST

PATH PARAMETERS

NAME	TYPE	DESCRIPTION
*test-case-id	string	The UID of the test case for which local tolerances will be removed.

RESPONSE

STATUS CODE - 200: Local tolerances removed.

STATUS CODE - 400: Bad request.

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 404: Test case not found.

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 406: Not acceptable.

RESPONSE MODEL - text/plain

```
string
```

STATUS CODE - 500: Internal server error.

RESPONSE MODEL - text/plain

```
string
```