2024

WATS STANDARD JSON FORMAT MANUAL

DESCRIBES THE WSJF FORMAT AND HOW TO USE IT

VIRINCO AS

CONTENTS

GENERAL DETAILS

WATS format name	Wats Standard JSON Format (WSJF)
Version	1.2
File example name	WatsStandardJSONFormat.json
Last modified date	2024-Apr-18

The WATS Standard JSON Format is a standardised format for importing or exporting WATS reports. The WATS Standard JSON Format adheres to the WSJF JSON schema.

This document describes WSJF for use with the built-in WATS Client converter, WSJF-Importer, for test reports.

JSON

WSJF is a JSON format. JSON defines objects with properties, which have values. Property values can be one of the following.

STRING

Strings are quoted using “. They may have a format specified, for example date-time.

```
"property": "string value"
```

NUMBER

Numbers are not quoted. The decimal separator is ‘.’, so PI should be written as 3.14159. Do not use thousand separators. Only digits and decimal point is allowed.

```
"property": 3.14159
```

INTEGER

Integers are not quoted. Do not use thousand separators.

```
"property": 100
```

BOOLEAN

Boolean values are not quoted. They can be either “true” or “false”.

```
"property": false
```

ARRAY

Arrays contain multiple of another value type.

```
"property": ["value 1", "value 2", "value 3"]
```

OBJECT

Objects have properties.

```
"property": {
      "property1": "some value 1",
   "property2": "some value 2",
}
```

NULL

Null is an empty value. The property then has no value.

```
"property": null
```

IMPORTANT

Required properties must be defined. If a property has value type null, it can have no value, even if the property is required. If a property is not required, it can be omitted. For export, some properties are not included when their value is null.

NUMERIC AND DATE FORMATS

Numeric: By default, '.' (period) is used as decimal separator, so PI should be written as 3.14159. Do not use thousand separators. Only digits and decimal point is allowed.

Numeric properties have a corresponding -Format property where you can set how the number will be displayed in WATS. It uses C-style printf formatting.

Date and time: Use the ISO 8601 format, e.g. 2019-09-12T14:26:16.977+02:00

WATS CLIENT PARAMETERS

Some fields in WSJF can be configured as parameters in the WATS Client Configurator. These parameters are used if the corresponding field is not included in the WSJF report.

machineName, location, and purpose fields will use the computer name, location, and purpose values from the WATS Client if the fields are not included.

start will use the current time when the converter is running if start is not included.

status and result are only used when the testMode parameter is Import. When testMode is Active the WATS Client will calculate status based on values and limits, and ignore the status field.

REPORT

WSJF must begin with an object. The root object describes the report.

Property Name	Description	Data Format	Importance
type	The type of report.	String (1) (T for a UUT report)	Required
result	The outcome of the test.	String (1) See status codes	Required
root	The root test step of the report.	Step object	Required
pn	The part number of the unit.	String (100)	Required, parameter
rev	The revision of the unit.	String (100)	Required, parameter
sn	The serial number of the unit.	String (100)	Required, parameter
processCode	The operation type code for a WATS process.	Integer (16)	Required, parameter
location	The location where the test takes place.	String (100)	Required, parameter
purpose	The purpose of the test machine.	String (100)	Required, parameter
machineName	The name of the test station.	String (100)	Required, parameter
start	The start date and time in local time.	String (date-time)	Required, parameter
startUTC	The start date and time in UTC time.	String (date-time)	Optional
processName	The operation type name for a WATS process	String (100)	Optional
id	A Globally Unique ID of the report. A report submitted with the same ID as another will overwrite the report. If missing it will be generated.	GUID	Optional
miscInfos	Searchable miscellaneous information about the report.	Array of MiscInfo objects	Optional
subUnits	Information about sub units of the unit.	Array of SubUnit objects	Optional
additionalData	Additional header data.	Array of AdditionalData objects	Optional
uut	The header data for a UUT report.	UUT object or null	Optional (Not included if null)
miscInfos	The miscellaneous data of the report.	Array of miscinfo objects or null	Optional (Not included if null)
subunits	The sub units of the tested unit.	Array of subunit objects or null	Optional (Not included if null)

assets	The assets used in test.	Array of asset objects	Optional (Not included if null)

IMPLEMENTATION EXAMPLE


```
{
  "type": "T",
  "id": "bf5e5f36-8d25-4140-9ca9-dd1dea24154f",
  "pn": "WATS FAT",
  "sn": "1894.212031",
  "rev": "FAT",
  "processCode": 10,
  "processName": "SW Debug",
  "result": "F",
  "machineName": "VIC-OEF-TEST2",
  "location": "VM",
  "purpose": "apptest",
  "start": "2019-10-15T11:22:26.57+02:00",
  "startUTC": "2019-10-15T09:22:26.57Z",
  "root": null,
  "uut": null,
  "miscInfos": [],
  "subUnits": [],
  “assets” : []
}
```

UUT

The UUT object contains header data specific to a UUT report. It is recommended to fill out as much data as possible as this significantly improves the usability of the final WATS report.

Property Name	Description	Data Format	Importance
user	The name of the operator of the test system.	String (100)	Required, has configurable default
execTime	The total duration of the test in seconds.	Number or null	Optional
batchSN	The serial number of the batch the unit belongs to.	String (100)	Optional
testSocketIndex	The index of the test socket used when parallel or batch execution.	Short (100)	Optional
fixtureId	The id of the fixture used to perform the tests.	String (100)	Optional
errorCode	The error code of any error that occurred during the test.	Integer (32)	Optional
errorMessage	Message of the error that occurred during the test, if any.	String (500)	Optional
batchFailCount	The number of tests that failed in this batch.	Integer (32)	Optional
batchLoopIndex	The number of failed tests in the batch.	Integer (32)	Optional
stepIdCausedUUTFailure	The id of the step that caused the report to fail.	Integer (32)	Optional
comment	A comment about the report.	String (5000)	Optional

IMPLEMENTATION EXAMPLE


```
"uut": {
   "execTime": 34.5362799,
   "testSocketIndex": -1,
   "batchSN": "BatchSerialNumber",
   "comment": "sgfsd",
   "errorCode": 0,
   "errorMessage": null,
   "fixtureId": "Your fixture ID",
   "user": "administrator",
   "batchFailCount": null,
   "batchLoopIndex": null
}
```


MISCINFO

Any further data concerning the UUT or the general test run must be entered as miscellaneous information. It can for example be used for software/firmware versions, or for whatever purpose suits the test. There can be any number of MiscInfo objects in a report.

Property Name	Description	Data Format	Importance
description	The name of the misc info	String (100)	Required
typedef	The type defition of the misc info.	String (30)	Optional
numeric	The numeric value of the misc info	Integer (32)	Optional
text	The text value of the misc ifno	String (100)	Optional

IMPLEMENTATION EXAMPLE

```
"miscInfos": [
  {
    "description": "Misc info 1",
    "typedef": null,
    "text": "Misc string 1",
    "numeric": 1
  },
  {
    "description": "Misc info 2",
    "typedef": null,
    "text": "Misc string 2",
    "numeric": 2
  }
 ]
```

SUB UNITS

You can attach sub-units to the test report, which will be linked in the report header. This allow you track systems and their individual component, which gives traceability to the systems lifecycle.

Register a sub unit by adding a SubUnit object to the SubUnits array.

Property name	Description	Data Format	Importance
pn	The partnumber of the sub unit.	String (100)	Required
sn	The serial number of the sub unit.	String (100)	Required
rev	The revision of the sub unit.	String (100)	Required
partType	The type of sub unit.	String (50)	Required

IMPLEMENTATION EXAMPLE

```
"subUnits": [
  {
    "partType": "Sub UUT 1",
    "pn": "Sub PN 1",
    "rev": "Sub rev 1",
    "sn": "Sub SN 1"
  },
  {
    "partType": "Sub UUT 2",
    "pn": "Sub PN 2",
    "rev": "Sub rev 2",
    "sn": "Sub SN 2"
  }
 ]
```

ASSETS

You can log assets used during test to the test report, which will be linked in the report header. This allow you track which assets were used to test each unit and the get the asset's statistics at the time the test was performed.

Register an asset by adding an Asset object to the Assets array.

Property name	Description	Data Format	Importance
assetSN	The asset serial number.	String (100)	Required
usageCount	The number of times the asset was used during test	Integer (32)	Required

IMPLEMENTATION EXAMPLE

```
"assets": [
   {
     "assetSN": "asset123",
     "usageCount": 1
   }
]
```

ADDITIONAL DATA

You can add additional header data to the report. Additional data is a free-form structure of properties and values. If you use the name “Station info”, that additional data will be treated as additional station data.

Add an additional data by adding a AdditionalData object to the AdditionalData array.

Property Name	Description	Data Format	Importance
name	Name of additional data.	String	Required
props	Properties of additional data	Array of AdditionalDataProperty objects	Required

ADDITIONAL DATA PROPERTY

Stores information about an additional data property.

Property Name	Description	Data Format	Importance
name	Name of property.	String	Required
type	Value type of property.	Number, String, Bool, Obj, or Array	Required
props	Properties of additional data	Array of AdditionalDataProperty objects	Required
flags	Bit flags of property.	Number	Optional (not included if null)
value	Value string of property	String	Required for Number, String, and Bool types, else Optional (not included if null)
comment	Comment of property	String	Required for Number, String, and Bool types, else not included.
props	Array of sub-properties. Used for type Obj.	Array of AdditionalDataProperty objects	Required for Obj type, else not included.
array	Array information. Used for type Array.	AdditionalDataArray object	Required for Array type, else not included.

ADDITIONAL DATA ARRAY

A property can be an array. Information about the array is stored in an AdditionalDataArray object.

Property Name	Description	Data Format	Importance
dimension	Dimension of array.	Number	Required
type	Type of values in the array.	Number, String, Bool, Obj, or Array	Required
indexes	Indexes of the array.	Array of AdditionalDataArrayIndex objects	Required

ADDITIONAL DATA ARRAY INDEX

Each index in an array and its value is stored as an AdditionalDataIArrayndex.

Property Name	Description	Data Format	Importance
text	The index written as text.	String	Required
indexes	Array of indexes ordered by dimension.	Array of Number	Required
value	The value of the index.	AdditionalDataProperty	Required

IMPLEMENTATION EXAMPLE

```
"name": "Additional UUT info",
 "props": [
    {
       "name": "Port"
       "type": "Number"
       "value": 13
    },
    {
       "name": "Coordinates"
       "type": "Obj"
       "props": [
          {
             "name": "X"
             "type": "Number"
             "value": 4
          },
          {
            "name": "Y"
            "type": "Number"
            "props": 3.5
    },
 ]
```

STEPS

The root property of the root object contains the actual test data. It is the root step of the step hierarchy and must therefore only contain a sequence call.

Where the content of the step is depends on the step type. For example, if the step is a numeric limit step, the content is in the numericMeas property.

Property Name	Description	Data Format	Importance
name	The name of the test step.	String (100)	Required
status	The outcome of the test step.	String (1) See Status Codes	Required
stepType	The step type, a textual description of the step.	String. Defined in each step type below	Required
start	The start date and time of the step execution.	String (date-time)	Optional
group	The step group	String (1) (S (Startup), M (Main), or C (Cleanup))	Optional
totTime	Duration of the step execution in seconds.	Number	Optional
errorCode	The error code of the error that occurred during the step execution	Integer (32)	Optional
errorMessage	The error message of the error that occurred during the step execution	String (200)	Optional
causedSeqFailure	A flag indicating if this step caused the sequence to fail.	Boolean	Optional
causedUUTFailure	A flag indicating if this step caused the report to fail.	Boolean	Optional
reportText	The step comment.	String (5000)	Optional
interactiveExeNum	The interactive exe number of the step.	Integer (32)	Optional
tsGuid	The step id from TestStand.	String (30)	Optional
steps	A list of sub steps for this step. Only for steps of type SequenceCall.	Array of Step objects	Optional (not included if null)
numericMeas	A list of numeric measurements belonging to this step.	Array of NumericMeasurement objects	Optional (not included if null)
stringMeas	A list of text measurements belonging to this step.	Array of StringMeasurement objects	Optional (not included if null)

booleanMeas	A list of boolean measurements belonging to this step.	Array of BooleanMeasurement objects	Optional (not included if null)
additionalResults	A list of additional results for the step.	Array of AdditionalData objects	Optional (not included if null)
seqCall	The information about the sequence call.	SequenceCall object	Optional (not included if null)
callExe	Information about the executable called by this step.	CallExe object	Optional (not included if null)
messagePopup	Information about the popup message belonging to this step.	MessagePopup object	Optional (not included if null)
chart	The chart belonging to this step.	Chart object	Optional (not included if null)
attachment	The attachment belonging to this step.	Attachment object	Optional (not included if null)
loop	Information about the loop the step is in.	LoopInfo object or null	Optional (not included if null)

IMPLEMENTATION EXAMPLE


```
"group": "M",
"stepType": "SequenceCall",
"interactiveExeNum": null,
"name": "MainSequence Callback",
"start": "2019-10-15T11:22:26.94+02:00",
"status": "F",
"errorCode": 0,
"errorMessage": null,
"tsGuid": "ID#:xnW0PT0ORE2el7wF7uaxyB",
"totTime": 34.5362799,
"causedSeqFailure": true,
"causedUUTFailure": false,
"reportText": null
```

STEP TYPES

Step type	Description	Dailed information
SequenceCall	Used to create a hierarchy structure	SequenceCall
NumericLimit	Numeric limit test	NumericLimit
StringValue	String value test	StringValue
PassFail	Pass/Fail test	PassFail
Chart	Graphs with data series	Chart
CallExe	Logs data from a call .exe step	CallExe
Attachment	Attachments such as README, images, etc.	Attachment
MessagePopup	Logs data from a message popup step	MessagePopup
Action	Used to log test action data and information	Action

SEQUENCE STEPS

Sequence steps are steps which contains a sequence call. Sequence calls are used to create hierarchy structured data. A sequence step can have the following attributes:

Property Name	Description	Data Format	Importance
name	The name of the sequence.	String (200)	Required
version	The version of the sequence file	String (30)	Required
path	The path to the sequence file	String (500)	Required

IMPLEMENTATION EXAMPLE FOR SEQUENCE CALLS


```
{
"group": "M",
"stepType": "SequenceCall",
"interactiveExeNum": null,
"name": "NI steps (NI seq call)",
"start": "2019-10-15T11:22:27.636+02:00",
"status": "F",
"errorCode": 0,
"errorMessage": null,
"tsGuid": "ID#:+u20X0V9vkGTSuFPK0137D",
"totTime": 18.768614,
"causedSeqFailure": true,
"causedUUTFailure": false,
"reportText": null,
"steps": [
 {
  [...]
 },
 {
  [...]
  "booleanMeas": [
   {
    "name": null,
    "status": "P"
   }
  ]
 },
"seqCall": {
 "path": "Z:\\Test and Demo\\FAT-SAT\\TS2016\\SeqFiles\\TS2016 - FATv1.seq",
 "name": "NI steps",
 "version": null
}
```

TEST STEPS

A step containing a measurement is defined as a test step. There are three test step types:

- NumericLimitTest
- StringValueTest
- PassFailTest

A step with one of these types must contain the corresponding measurement property.

Test steps comes in two different forms:

1. Single steps (one measurement) (*recommended*). Do **NOT** use the name property.
2. Multiple steps (two or more measurements). The name property **MUST** be set and have a unique name within the step.

Test steps must have status attribute filled in (Passed/Failed). For single steps it is also possible to use Skipped, Done, Terminated and Error.

This table explains the proper keyword to use for single and multiple steps.

Type	Keyword (single)	Keyword (multiple)
Numeric Limit Test	StepType=”ET_NLT”	StepType=”ET_MNLT”
String Value Test	StepType=”ET_SVT”	StepType=”ET_MSVT”
Pass Fail Test	StepType=”ET_PFT”	StepType=”ET_MPFT”

NUMERIC MEASUREMENT TEST

Numeric tests are used to measure numeric values. Numeric limit tests are defined with a value and a lower and/or higher limit. There is also a comparison operator to specify how to compare the measurement to the limit(s).

Property Name	Description	Data Format	Importance
value	The measured value.	Number	Required
compOp	Which comparison operation to use on the measures value.	String (10) See Comparison Operations	Required
status	The status of the test.	String (1) See Status Codes	Required
unit	The unit of the measured value.	String (20)	Required
name	The name of the limit test. Only use for multiple measurements.	String (100)	Forbidden if single test in step. Required & Unique if multiple tests in same step.
lowLimit	The limit used to compare the measured value in greater than and equal/not equal comparisons.	Number	Optional / Required if comparison operation is GE, EQ, or NE
highLimit	The high limit used to compare measured value in less than comparisons.	Number	Optional / Required if Compare Operator is LE

IMPLEMENTATION EXAMPLE


```
{
  "group": "M",
  "stepType": "ET_NLT",
  "interactiveExeNum": null,
  "name": "Numeric Limit Test",
  "start": "2019-10-15T11:22:46.604+02:00",
  "status": "P",
  "errorCode": 0,
  "errorMessage": null,
  "tsGuid": "ID#:F4CrB6yGQEChoY6s9p4QyB",
  "totTime": 0.0443013,
  "causedSeqFailure": null,
  "causedUUTFailure": false,
  "reportText": null,
  "numericMeas": [
    {
      "compOp": "GELE",
      "name": null,
      "status": "P",
      "unit": null,
      "value": 0.0,
      "highLimit": 10.0,
      "lowLimit": 0.0
    }
  ]
}
```

STRING MEASUREMENT TEST

String tests are used to compare text. String tests are defined with a value and a limit. There is also a comparison operator to specify how to compare the text to the limit.

Property Name	Description	Data Format	Importance
value	The text to compare.	String(100)	Required
compOp	Which comparison operation to use on the text.	String (10) See Comparison Operations	Required
status	The status of the test.	String (1) See Status Codes	Optional
name	The name of the measurement test. Only use for multiple measurements.	String (100)	Forbidden if single test in step. Required & Unique if multiple tests in same step.
limit	The text to compare against.	String (100)	Required / Optional if Comparison Operator is Logging

IMPLEMENTATION EXAMPLE


```
{
  "group": "M",
  "stepType": "ET_SVT",
  "interactiveExeNum": null,
  "name": "String Value Test",
  "start": "2019-10-15T11:22:46.696+02:00",
  "status": "P",
  "errorCode": 0,
  "errorMessage": null,
  "tsGuid": "ID#:0iGa1tIVlEqgfYvRXkWwyD",
  "totTime": 0.0023617,
  "causedSeqFailure": null,
  "causedUUTFailure": false,
  "reportText": null,
  "stringMeas": [
    {
      "compOp": "IgnoreCase",
      "name": null,
      "status": "P",
      "value": "this is a string result",
      "limit": "this is a string result"
    }
  ]
}
```

BOOLEAN MEASUREMENT TEST

Boolean or pass/fail tests are tests without measurements. They only have a status.

Property Name	Description	Data Format	Importance
status	The status of the test.	String (1) See Status Codes	Required
name	The name of the measurement test. Only use for multiple measurements.	String (100)	Forbidden if single test in step. Required & Unique if multiple tests in same step.

IMPLEMENTATION EXAMPLE


```
{
  "group": "M",
  "stepType": "ET_PFT",
  "interactiveExeNum": null,
  "name": "Pass/Fail Test",
  "start": "2019-10-15T11:22:47.754+02:00",
  "status": "P",
  "errorCode": 0,
  "errorMessage": null,
  "tsGuid": "ID#:WV00LdKgZEii64JAako6vC",
  "totTime": 0.0035034,
  "causedSeqFailure": null,
  "causedUUTFailure": false,
  "reportText": null,
  "booleanMeas": [
    {
      "name": null,
      "status": "P"
    }
  ]
}
```

ACTION STEPS

Used to log test action data and information. These steps are always single steps, never multiple.

IMPLEMENTATION EXAMPLE


```
{
  "group": "M",
  "stepType": "Action",
  "interactiveExeNum": null,
  "name": "WATS test steps",
  "start": "2019-10-15T11:22:46.563+02:00",
  "status": "D",
  "errorCode": 0,
  "errorMessage": null,
  "tsGuid": "ID#:Kw7PDR4uKEuNqK6qbHl9CA",
  "totTime": 0.0009073,
  "causedSeqFailure": null,
  "causedUUTFailure": false,
  "reportText": null
}
```

CHART

Charts can be added to empty steps, or in addition to tests.

Property Name	Description	Data Format	Importance
chartType	The type of chart.	String (30) (Line, LineLogXY, LineLogX, LineLogY)	Required
label	The name of the chart	String (100)	Required
xLabel	The name of the X axis	String (50)	Required
xUnit	The unit of the X axis	String (20)	Required
yLabel	The name of the Y axis	String (50)	Required
yUnit	The unit of the Y axis	String (20)	Required

CHART SERIES

A Series element (or XY-plot) contains a semicolon separated list of x-values and y-values. This version only supports the XYG datatype. Each chart can have up to 10 series.

- YData and XData can't have more than 10000 entries each.
- The number of elements in Y and X must match.

Property Name	Description	Data Format	Importance
name	Name of plot.	String (100)	Required
xdata	All data for x-axis.	String (Semicolon-separated list)	Required
ydata	All data for y-axis.	String (Semicolon-separated list)	Required
dataType	Attribute (only XYG supported)	String (10)	Required

IMPLEMENTATION EXAMPLE


```
  "chart": {
  "chartType": "Line",
  "label": "Clabel",
  "xLabel": "Xlabel",
  "xUnit": "Xunit",
  "yLabel": "Ylabel",
  "yUnit": "Yunit",
  "series": [
   {
     "dataType": "XYG",
     "name": "PlotName 0",
     "xdata": "0;1;2;3”
     "ydata": “0.505966492022607;0.64645051547995;0.639659061179134;0.911752170787661”
   }
}
```

ATTACHMENT

Attachments can be included in a step by adding an Attachment object to a step. WSJF uses Base64 encoded contents to attach a file.

Property Name	Description	Data Format	Importance
name	The name for the attachment.	String (100)	Required
contentType	The Mime-type of the attachment.	String (100)	Required
data	A base64 encoded string of the contents of the file.	String	Required

IMPLEMENTATION EXAMPLE

```
"attachment": {
  "name": "stepdata.xml",
  "contentType": "text/xml",
  "data": "PFRoaXMgaXMgdGhlIFhNTCBzdHJpbmcgXD4="
}
```

MESSAGE POP UP

Message popup is used when a pop-up message is displayed.

Property Name	Description	Data Format	Importance
button	The code for which button was pressed.	Integer	Required
response	The message from the pop up.	String (100)	Required

IMPLEMENTATION EXAMPLE

```
"messagePopup": {
  "response": null,
  "button": 2
}
```

CALL EXECUTABLE

Step which contains a reference to an executable.

Property Name	Description	Data Format	Importance
exitCode	The exit code from the executable.	Number	Required

IMPLEMENTATION EXAMPLE

```
"callExe": {
  "exitCode": 0
}
```

ADDITIONAL RESULTS

A step can have additional results, which is a free-form structure of properties and values.

Property Name	Description	Data Format	Importance
name	Name of additional data.	String	Required
props	Properties of additional data	Array of AdditionalDataProperty objects	Required

COMPARISON OPERATORS

The table below describes the different Comparison operators for test types.

CompOperator	Description	Step type
EQ	Equal	Numeric, String
NE	Not Equal	Numeric, String
GT	Greater than	Numeric, String
LT	Less than	Numeric, String
GE	Greater or equal	Numeric, String
LE	Less or equal	Numeric, String
GTLT	Greater than low limit, less than high limit	Numeric
GELE	Greater or equal than low limit, less or equal than high limit	Numeric
GELT	Greater or equal than low limit, less than high limit	Numeric
GTLE	Greater than low limit, less or equal than high limit	Numeric
LTGT	Less than low limit, greater than high limit	Numeric
LEGE	Less or equal than low limit, greater or equal than high limit	Numeric
LEGT	Less or equal than low limit, greater than high limit	Numeric
LTGE	Less than low limit, greater or equal than high limit	Numeric
LOG	Loging values (no comparison)	Numeric, String
CASESENSIT	Case sensitive	String
IGNORECASE	Ignore case / !CASESENSIT	String

STATUS CODES

Below is a list of supported status codes for a step.

Status	Code	UUT description	Step / measurement description	Additional information
Passed	P	UUT passed	Step passed	
Failed	F	UUT failed	Step failed	
Done	D	Not used.	Step completed.	Can be used for single steps, should not be used for multiple steps
Skipped	S	Not used.	Step execution skipped.	Can be used for single steps, with caution for multiple steps.
Error	E	An error occurred during test execution	An error occurred during step execution.	Can be used for single steps, should not be used for multiple steps.
Terminated	T	Operator terminated test execution	Operator terminated step execution.	Can be used for single steps, should not be used for multiple steps.

The UUT report itself can be Passed, Failed, Error or Terminated.

A single step can have all statuses.

A single measurement (in a multiple step) can only be Passed or Failed or Skipped.