2018

WATS STANDARD XML FORMAT MANUAL UUR

DESCRIBES THE WSXF UUR FORMAT AND HOW TO USE IT

VIRINCO AS

CONTENTS

UNIT UNDER REPAIR (UUR)

A unit under repair (UUR) is a WATS report type for units and is under repair or has been repaired. UUR reports are used to organize and document repaired or replaced components that failed in a unit under test (UUT). It is recommended generate UUR reports through the *User Interface – repair* functionality in WATS. Doing this will generally increase report quality and ensure test and repair report linking.

The WSXF format was developed to accommodate customers need to import their own test-data into WATS without a converter. WSXF has since then become a powerful format that allow customers to generate a report with all WATS's features and functionality.

UUR FEATURES

UUR reports contains features one would typically expect for repairing components. Below is a list of the main features that together makes up a repair report. These features and their properties will be future elaborated on in this document.

- **Failures**
  A unit can have multiple failures. Unlimited failures can be attached to a single unit.

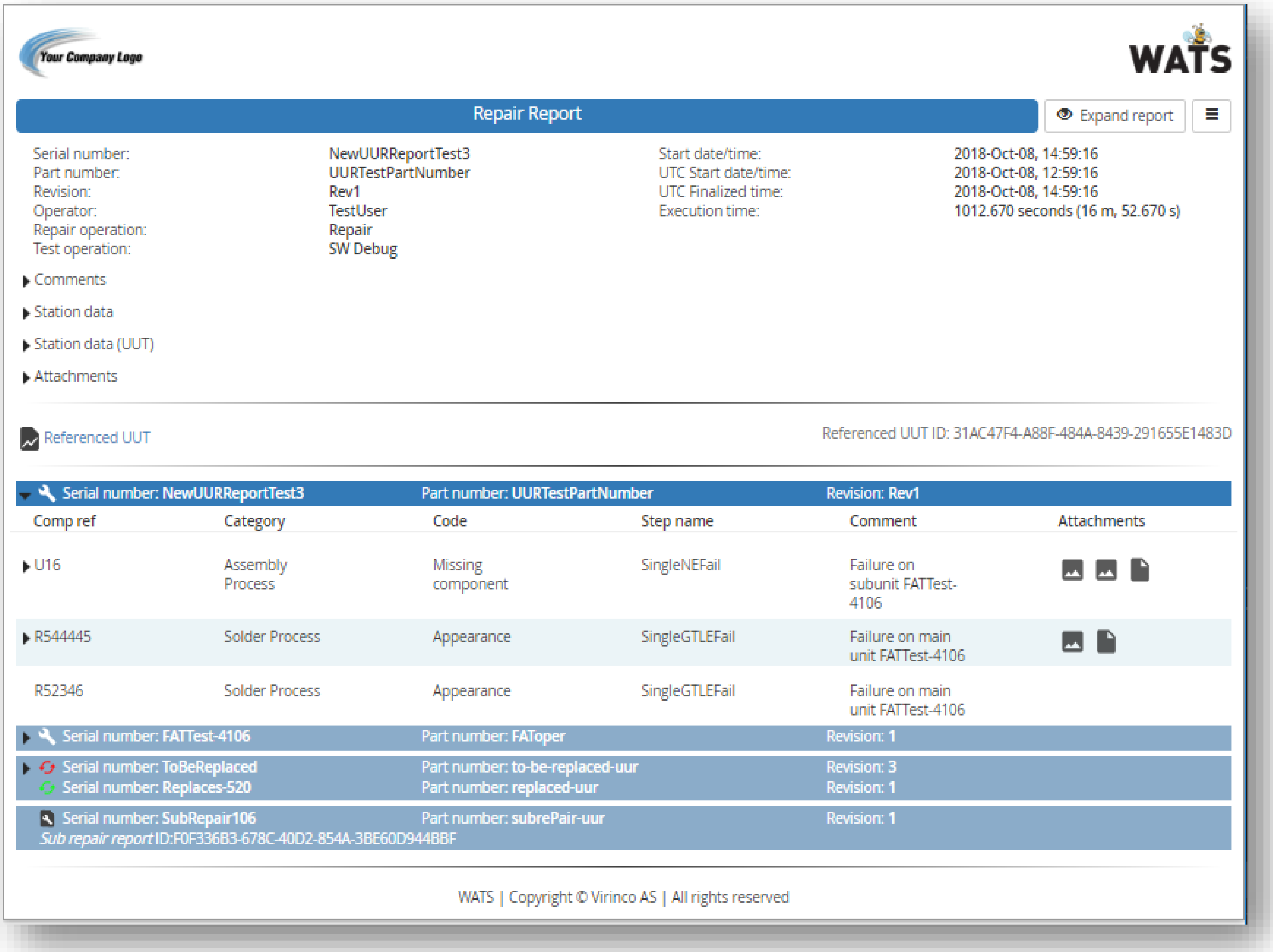
- **Attachments**
  Attachments can be added to a failure and/or at the report root level to document the failure.
  Attachments can be of all types, and unlimited in size*.
  *(default binary attachment size is set to 2MB, but this can be changes in the Settings.JSON file found in Programdata/Wats)

- **Replaced units**
  If it's more efficient to replace a unit instead of repairing it, one can register a unit replacement.

- **Sub-repair**
  If there are failures on multiple sub-units on a main unit, one can create repair reports for each sub-unit and attach the sub-repair reports to the main report.

UUR REPORT LAYOUT

When a UUR report is successfully created, it can be located in the UUR Report Section in the *Reporting* menu.



Repair Report

Serial number:	NewUURReportTest3	Start date/time:	2018-Oct-08, 14:59:16
Part number:	UURTestPartNumber	UTC Start date/time:	2018-Oct-08, 12:59:16
Revision:	Rev1	UTC Finalized time:	2018-Oct-08, 14:59:16
Operator:	TestUser	Execution time:	1012.670 seconds (16 m, 52.670 s)
Repair operation:	Repair		
Test operation:	SW Debug		

▸ Comments

▸ Station data

▸ Station data (UUT)

▸ Attachments

Referenced UUT

Referenced UUT ID: 31AC47F4-A88F-484A-8439-291655E1483D

▾ Serial number: NewUURReportTest3 Part number: UURTestPartNumber Revision: Rev1

Comp ref	Category	Code	Step name	Comment	Attachments
▸ U16	Assembly Process	Missing component	SingleNEFail	Failure on subunit FATTest-4106	
▸ R544445	Solder Process	Appearance	SingleGTLEFail	Failure on main unit FATTest-4106	
R52346	Solder Process	Appearance	SingleGTLEFail	Failure on main unit FATTest-4106	

▸ Serial number: FATTest-4106 Part number: FAToper Revision: 1

▸ Serial number: ToBeReplaced Part number: to-be-replaced-uur Revision: 3
Serial number: Replaces-520 Part number: replaced-uur Revision: 1

Serial number: SubRepair106 Part number: subrePair-uur Revision: 1
Sub repair report ID:F0F336B3-678C-40D2-854A-3BE60D944BBF

WATS | Copyright © Virinco AS | All rights reserved

GENERAL DETAILS

WATS format name	Wats Standard XML Format (WSXF)
Version	1.2
File example name	WatsStandardXMLFormat.xml
Last modified date	10.12.2018

This format is a standardised format that the WATS Client will automatically read and import into WATS (was first release in 2014) The WATS Standard XML Format follows the WSXF XML schema.

Each report consists of two main parts; a **Header data** part, a **Unit** part. The Header data part is further split into two sub parts; a list of predefined headers followed by optional data sections, such as miscellaneous UUR data, and comments.

The WSXF Converter receives an Xml file with UUR or UUR data. It goes through the elements in the file and creates a UUT/UUR report in the TDM API with all the tests and measurements included.

This document describes the features of the UUR report and how to generate one using the WSXF format.

A sample UUR report can be found at the following location:
https://virinco.zendesk.com/hc/en-us/articles/207424643-Wats-Standard-Xml-Format-WSXF-

XML

WSFX is XML format. XML is made up of elements. An XML element has an opening tag with a name and attributes, a closing tag, and content.

The opening tag of an element starts with a < character followed by the name of the tag. After the name comes an optional list of attributes for the element. Attributes are defined with the name of the attribute, then the value of the attribute in quotes after an equals sign. The opening tag ends with a > character.

```
<Report type="UUR" Start="2018-01-01T08:30:00" Result="Passed">
```

The closing tag of an element starts with </ followed by the name of the element, and ends with a > character.
WSXF uses <Report> element to organize the data.

```
</Report>
```

Some elements have no content and can be self-closing. In this case the opening tag ends with />

```
<Process Code="10" />
```

Between the opening and closing tags of an element is the contents of the element. The content can be text or other elements. Elements inside other elements are called nested elements and are children of the element they are nested in. All child elements must be closed before the parent can be closed.

```
<Reports>
   <Report type="UUR" Start="2018-01-01T08:30:00" Result="Passed"> ←Child of reports
      <Process Code=”10” />                                        ← Child of report 1
   </Report>
   <Report type="UUR" Start="2018-01-01T08:40:00" Result="Passed"> ← Child of reports
      <Process Code=”10” />                                        ← Child of report 2
   </Report>
</Reports>
```

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.

- **DateTime:**
  Use the ISO 8601 format (2018-09-12T14:26:16.977+02:00) e.g.
  If requested, support for custom numeric / date formats can be added.

REPORTS

A WSXF file should start with a <Reports> element. <Reports> can contain one <Report> elements, each containing data from one report.

```
<Reports
    <Reports xmlns="http://wats.virinco.com/schemas/WATS/Report/wsxf"></Reports>
```

HEADER DATA

The Header data part of the file consists of two parts. The first part is the ‘Report’ element containing basic report information, such as *SerialNumber, PartNumber and Revision.* The second part is the ‘UUR’ element which contains UUR information, such as *FinalizeDate* and *ExectutionTime*, and information about the referenced UUT linked to the UUR, and its process code / name.

Required information (Header)

```
<Report
    type="UUR"
    Start="1900-01-01T01:00:00.000+01:00"
    SN="FATTest-268" PN="FATPartNo"
    Rev="Rev1" >
  <UUR UserLoginName=”FAToper” />
  <Process Code="60" Name="FST" />
```

REPORT ELEMENT

Below is a list of the <Report> element attributes. It is recommended to fill in as much data as possible as this significantly improves the usability of the final WATS report and analysis of report data. Note that either “Start” or “Start_UTC” is required.

Attribute Name	Description	Data Format	Importance
Type	The type of report.	String (UUT or UUR)	Required
SN (Serial Number)	Serial number of the unit.	String(100)	Required
PN (Part Number)	Part number of the unit.	String(100)	Required
Start	Start time in local time. If missing, calculated based on Start_utc.	String(Formated according to ISO 8601): 2018-09-12T14:26:16.977+02:00	Required (can fill in «Start_utc» instead.
Rev	Revision of the unit.	String(100)	Recommended
Start_utc	Start time in UTC time zone. If missing, calculated based on Start.	String (See Start)	Optional
Location	Location where the test takes place.	String(100)	Optional
Machine Name	Name of test station. If missing, the computer name will be used.	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 will be generated.	GUID	Optional
Origin	ID of the WATS client that generated the report.	GUID or MAC address	Optional
Purpose	Automated test equipment purpose (Report origin)	String(100)	Optional

IMPLEMENTATION EXAMPLE

```
<Report type="UUR"
            Start="2018-10-08T14:59:16.9536707+02:00"
            Start_utc="2018-10-08T12:59:16.9536707Z"
            SN="RepairSerial" PN="SerialPartNumber" Rev="Rev1" origin="001CBFFBC178"
            MachineName="TestMachine"
            Location=”Repair Station Location"
            Purpose="Repair Station Purpose"
>
```

UUR ELEMENT

The first sub-element of the report element is the UUR Element. If the report is a UUR report. The UUR element contains additional header information regarding the UUR report. It is recommended to fill out as much data as possible as this significantly improves the usability of the final WATS report.

Attribute Name	Description	Data Format	Importance
UserLoginName	Name of the operator of the test system.	String(100)	Recommended
ReferencedUUT	Referenced UUT report identity (Guid)	Guid (String)	Strongly Recommended
ExecutionTime	Total duration of test in seconds.	Double	Recommended
FinalizeDate	Date when UUR was Finalized.	String(Formated according to ISO 8601): 2018-09-12T14:26:16.977+02:00	Optional (required if ConfirmDate is not set)
Parent	Reference to parent repair report. This repair report is considered a "sub-repair" if this value is present	Guid (String)	If the UUR report is a sub-report: Required /Strongly Recommended

The UUR may contain a <Comment> Element.

Element Name	Description	Data Format	Importance
Process	Process code used in the **UUT** report. If there is no referenced UUT report, use process code «10» as shown in the implementation example.	Double	Required
Comment	Basic text comment that can be attached to the UUR report header	String(5000)	Optional

IMPLEMENTATION EXAMPLE

```
<UUR ReferencedUUT="71dec753-4c17-4ba4-9a59-c2caae427f37"
    ConfirmDate="2018-04-01T13:40:40" FinalizeDate="2018-10-08T14:59:16.9536707+02:00">
   <Comment>This is a Comment</Comment>
   <Process Code=”10”/>
</UUR>
```

PROCESS ELEMENT / REPAIR TYPE

The <Report> element must contain a <Process> element. The element contains the process type used for the UUR. You can create and manage Processes in Control panel -> Configure/Settings -> Processes. Note that the checkbox for “Is repair operation” must be checked for UUR report processes. This <Process> element is a sub-element of the <Report> element, and should not be confused with the <Process> element of the UUR element

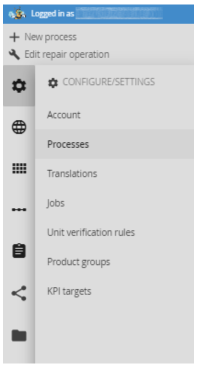
Attribute Name	Description	Data Format	Importance
Code	Operation type code for a WATS process.	Short	Required / Optional if Name is supplied
Name	Name of the operation type for a WATS process.	String(500)	Optional / required if Code is not supplied

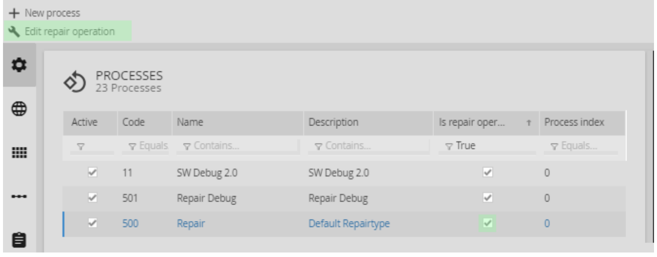
IMPLEMENTATION EXAMPLE

```
<Process Code="500" Name=”Repair”>
```

MISCELLANEOUS ELEMENT

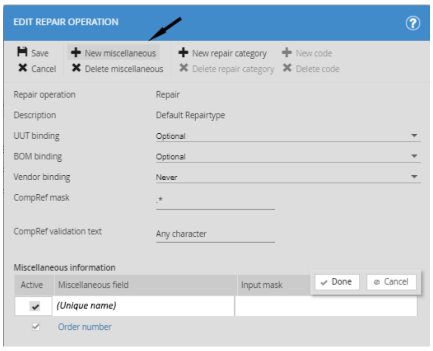
Any further data related to the UUR must be entered as miscellaneous UUR data. Miscellaneous info must be defined in a repair type before they can be used. Each miscellaneous description can only be used ***once*** in a UUR report. To register a new miscellaneous element, make sure you have permissions to access the Control panel -> Configure settings -> Processes settings. Make sure you have a registered repair operation. To edit a repair type, click on the operation to mark it, then click 'Edit repair operation'. You can now add 'new miscellaneous'. Each misc. element must have a unique name.





PROCESSES
23 Processes

Active	Code	Name	Description	Is repair oper...	Process index
	Equals	Contains...	Contains...	True	Equals...
✓	11	SW Debug 2.0	SW Debug 2.0	✓	0
✓	501	Repair Debug	Repair Debug	✓	0
✓	500	Repair	Default Repairtype	✓	0



Repair operation: Repair
Description: Default Repairtype
UUT binding: Optional
BOM binding: Optional
Vendor binding: Never
CompRef mask: .*
CompRef validation text: Any character

Miscellaneous information

Active	Miscellaneous field	Input mask
✓	*(Unique name)*	
✓	Order number	

Attribute Name	Description	Data Format	Importance
Description	Name of the miscellaneous info	String(100)	Required
Typedef	Defition of the miscinfo type.	String(30)	Optional

Data	Description	Data Format	Importance
Miscinfo Data	Miscellaneous element content	String(30) / Numeric	Optional

IMPLEMENTATION EXAMPLE

```
<MiscInfo Typedef="" Description="Order number">1234</MiscInfo>
```

SUB UNITS / REPORTUNITHIERARCHY

To link failures and attachments to a unit, one must create a unit hierarchy. Units added in the hierarchy will become “headers” that failures can be linked to.

To start off, create a unit in the hierarchy that contains the serial, part, and revision data from the “Report” element. This will be the report’s main unit. The “idx” number for main unit should always be 0, and failures and attachments related to this unit should have their index properties set to this value.

Attribute name	Description	Data (Format)		Importance
PN	Part number	String(100)		Required
SN	Serial number	String(100)		Required
Rev	Revision	String(100)		Required
Idx	Relation index to link sub unit to a failure element in the UUR report (Idx value = Failure attribute ‘PartIdx’)	Int(32)		Required
ParentIDX	Sub-part’s parent sub asset. Used to define the hierarchy of subparts.	Int(32)		Recommended
ReplacedIDX	Indicates that this subpart was replaced by the subpart with the Idx specified in this attribute	Int(32)		Optional
Position	Optional short sort-order value	Short		Optional
PartType	Sub-part type or category	String(50)		Optional

IMPLEMENTATION EXAMPLE (COMPUTER)

Main unit

```
<ReportUnitHierarchy SN="Computer1" PN="january_Computer1” Rev="4" Idx="0" />
```

Regular sub unit (Sub-unit of computer, was repaired and now works)

```
<ReportUnitHierarchy SN="Motherboard1" PN="january_motherboard1” Rev="1" Idx="1" ParentIDX="0" />
```

Replaced Unit (revision indicated that the PSU132 has had multiple repairs without passing)

```
<ReportUnitHierarchy SN="PSU132" PN="january_PSU132” Rev="3" Idx="2" ParentIDX="0"
ReplacedIDX="3"/>
```

Replacing/ New unit (new unit is passed test, and replaces PSU132)

```
<ReportUnitHierarchy SN=" PSU168" PN="="january_PSU168” Rev="0" Idx="3" ParentIDX="0" />
```

FAILURES DATA

The second part of the XML file contains the Failure data. To register a failure, you will need to have a fail category and/or a failure code. Failure types are defined within a repair operation, at the same location where you add misc. data. The WSXF Converter will search through all process categories for the specified code if Category is not specified and return the first instance Code with the matching name. It is therefore not recommended to have multiple codes with the same name. The table below describes the attributes for a <Failures> element.

Attribute Name	Description	Data Format	Importance
Code	Repair operation category repair code. Must be unique for each category.	String(200)	Required*
Idx	Failure index, increment for each Failure. Not to be mixed up with PartIndex, which is the index used to link failures to sub units.	Int(32)	Required
Category	Repair operation category name.	String(200)	Recomended
PartIdx	Used to link sub units and attachment to failues. See Idx.	Int(32)	Recommended. Defaults to 0 if not included
StepID	It is possible to link the failure to the UUT test step. Contains UUT step id	Int(32)	Recomended
CompRef	Electric Component referance	String(50)	Recomended
FunctionBlock	A group of electric components	String(100)	Optional
ArticleNumber	Electric component articlenumber	String(100)	Optional
ArticleRevision	Electric component revision	String(100)	Optional
ArticleVendor	Electric component vendor. Used to compare yield based on vendor	String(500)	Optional
ArticleDescription	Electric component description	String(500)	Optional
Failcode	Unique failcode ID. Not visible by default. Contact Virinco for more information. **Code or Catergory is not required if Failcode is specified.*	Guid	Not recomended

Element Name	Description	Data Format	Importance
Comment	Basic text comment that can be attached to the UUR report header	String(5000)	Optional

IMPLEMENTATION EXAMPLE

```
<Failures Category="Solder Process" Code="Appearance"
     Idx="0" PartIdx="2" CompRef="R55"  StepID="16">
    <Comment>Failure on main unit</Comment>
</Failures>
```

ATTACHMENT / BINARY

The WSXF supports attachments. Attachments can be included in the UUR report by using the <Binary> keyword. WSXF uses Base64 encoded contents to for handling attachments. Binaries are not structured hierarchically, and therefore requires an index (FailIdx) to link the attachment to a failure.

- To link an attachment to a unit, use the same value in FailIdx as in PartIdx.
- To link an attachment to report root, remove the “FailIdx” attribute from the binary tag.

Attribute	Description	Data (Format)	Importance
FailIdx	Relation index to link attachment to a failure element in the UUR report. (Idx value = Failure attribute ‘PartIdx’)	String(100)	Optional

DATA

Attribute	Description	Data (Format)	Importance
ContentType	MIME type (image/png..)	String(100)	Required
FileName	Displayed filename is the UUR report	String(100)	Optional
Size	Size of file	Int(32)	Optional

The Data element contains a Base64 encoded string.

Data	Description	Data Format	Importance
Binary	Miscellaneous element content	Base64 encoded string	Optional

IMPLEMENTATION EXAMPLE

```
<Binary FailIdx="2">
    <Data FileName="Subunit1" ContentType="image/png">
       /9j/4AAQSkZJRgABAQAAAQABAAD//AABEIALcBFAMBIgACEQEDEQH/xAA.....
    </Data>
</Binary>
```

To convert your file to binary, simply browse the internet for a “file to Base64” converter, and insert the value into the Binary Data element.