

# Test Infrastructure Engineering: Oracle Manufacturing Failure Analysis Data Transfer Specification

---

## Overview

This document is intended as a reference as to how to create a failure analysis ticket for consumption by TDMS.

## Audience

This document is targeted at test and manufacturing systems software engineers who are required to collate test results for TDMS delivery. A fundamental knowledge of XML, networking and software design is mandatory before attempting to create the Failure Analysis data.



# 1 CONTENTS

2	Failure ANALYSIS	3
3	LIFECYCLE OF A TICKET	3
4	Failure Analysis TICKET XML	4
4.1	Mandatory Attributes within the ticket	5
4.2	STATES of a ticket	6
5	The Ticket Id	8
6	Action Element	8
6.1.1	Part Categories	10
6.2	Action Types.	10
6.2.1	Type : repairAction Description : Use when a general repair of a part is being carried out, which cannot be categorized under any other Action.	10
6.2.2	Type : reseatAction Description : When a component is reseated into its slot or location.	11
6.2.3	Type : reruntestAction Description : When the system is being retested as part of debug, or verifying that the corrective action was successful.	11
6.2.4	Type : addAction Description : If you were to add a new part, or missing part to the system..	11
6.2.5	Type : removeAction Description : Use when you are removing a part from the system.	11
6.2.6	Type : swapAction Description : Use when two similar parts are being swapped , such as DIMMS which could be swapped between 2 slots.	11
6.2.7	Type : partReplacementAction Description : Use when a part appears faulty and you are completely replacing. This could also be achieved using the REMOTE and ADD actions	12
6.2.8	Type : abortAction Description : When a ticket is being aborted , the details as to when and why should be recorded as an action, though optional , since the STATE element should record the moving of STATE to ABORTED. .	12
6.2.9	Type : returntovendorAction Description : Use when a part is being returned to a vendor such that tracking information can be recorded. .	12
7	ASSOCIATING A TICKET TO A TEST RESULT	12
8	TICKET CATEGORIZATION	14
9	CROSS REFERENCE TICKETS	15
10	Solution Level Tickets	15
11	Accompanying Jar file.	16
12	Packaging the Ticket Bundle	16
13	Sending the ticket to oracle	17
13.1	Cloud PUSH	17
13.2	Rest Service Upload	18
14	FREQUENCY OF DELIVERY	18

## 2 FAILURE ANALYSIS

When a unit fails a test run, a ticket is raised to allow tracking of the analysis and corrective action, until the failure is resolved.

All these details should be provided to Oracle allowing for internal reporting and analysis of the steps taken to fix the problem.

The ticket provides input to engineering groups, both hardware and quality to allow for continual improvement in the product and process.

## 3 LIFECYCLE OF A TICKET

When a system fails at an operation, be it within an automated operation or manual (such as HIPOT), there should be a ticket created. This ticket will then be populated with all details.

A basic lifecycle for a ticket could be as follows:

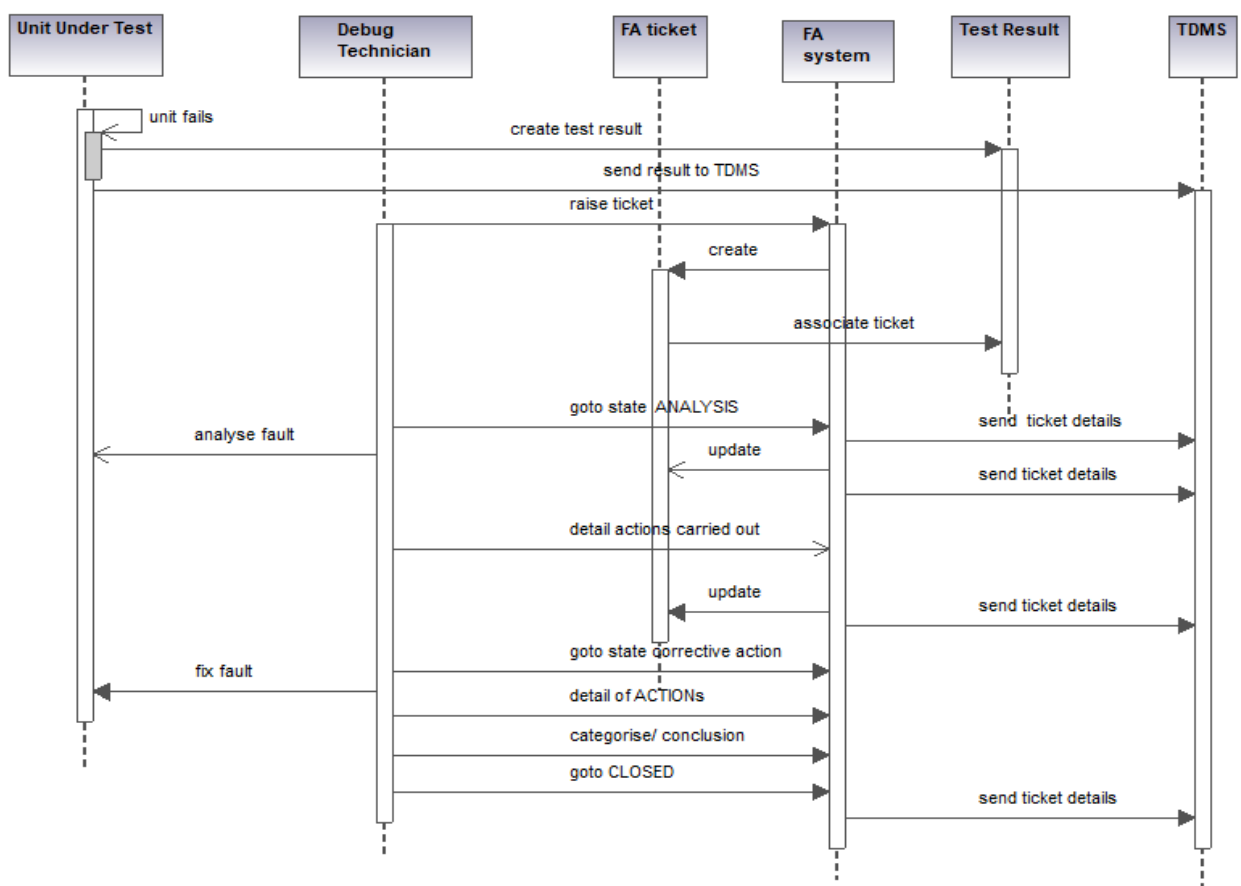


Figure 1 Sample Lifecycle of a ticket

Note that in the above, we have various points at which the data is sent to TDMS. This however is somewhat dependent on the FA system being used, and as to the requirement from Oracle as to when visibility of change would be required.

## 4 FAILURE ANALYSIS TICKET XML

The top-level XML structure for a ticket is as follows:

Note: the diagram shown may differ slightly to the schema that you will be given.

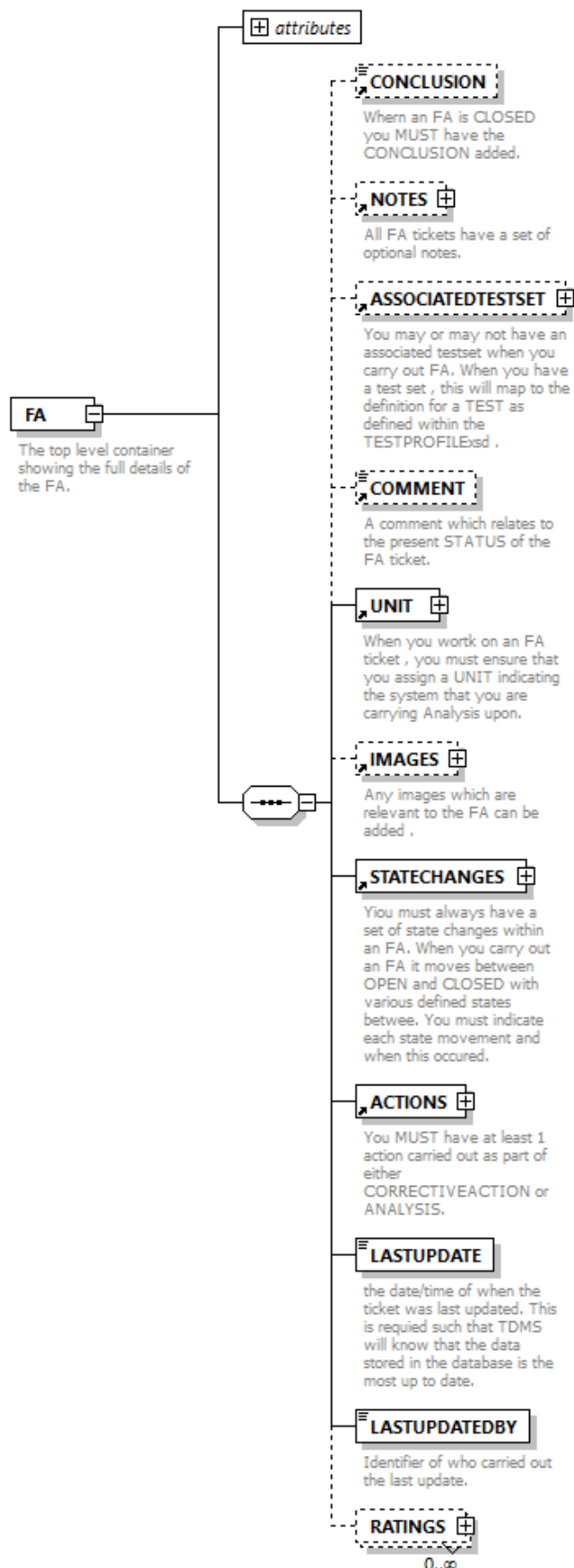


Figure 2 FA XSD top level

The key areas within the ticket which indicate the steps taken to root cause, and corrective action are.

1. **UNIT**: Intended to detail the system which is being debugged.
2. **STATECHANGES**: Intended to show the lifecycle of the ticket. From birth (OPEN) to closure (CLOSED).
3. **ACTIONS**: The actions carried out within each state to fix the issue.
4. **ASSOCIATEDTESTSET**: the details of the failed test result to which this ticket belongs. The association is made not to the test result but to the actual test set which failed within the test result. This provides insight into the test process and indicates exactly what it was within the test result that failed during the test run.
5. **LASTUPDATE** : Any time that ticket data is being sent to Oracle, the LASTUPDATE element should be updated so that Oracle TDMS will be able to establish that it has the most up to date data within its database.

## 4.1 Mandatory Attributes within the ticket

Attribute	Comment
Id	The ticket ID which is a unique identifier with no whitespace
status	The status reflects the most up to date STATE element entry. States are UPPERCASE with no whitespace.
owner	The present owner of the ticket, which may differ from the person who opened the tickets. For example, a ticket may move from Technician to Engineer as the fault is dealt with.
category,type,fault,subfault	These attributes combine produce a hierarchy of the root cause of the fault. See section 8 for categorization.
creationDate	The date time of when the ticket was created, to millisecond level.
createdBy	The person who raised the ticket which may differ from the owner.
failedLocation	The location where the system physically was tested and failed.
failedLogop	The operation that the system failed within. This should be an uppercase operation with no whitespace.
UNIT	The unit element should contain the details regarding the system that has failed such as the family, serial number etc. This is the target system being debugged.

## ACTIONS/

There must be a set of actions for a ticket. At the least 1 action and 3 states i.e.  
OPEN STATE, <N>STATE, CLOSED STATE  
Within the central state an action would be created.

*It cannot be overstated as to how important it is to populate the FROMPART and TOPART elements within the various actions as this is the only means by which under will be able to trace the movement of parts in and out of the system during repair.*

## 4.2 STATES of a ticket

When a ticket is created, it will contain a STATE element which has a value of OPEN. This tells us the details of as to when the ticket was opened and by whom.

All state movements are recorded within the : FA/STATECHANGES element.

No actions can be added to a ticket in a state of OPEN. It is assumed that the fault has been recorded and a ticket associated. However, no work has started as yet.

Once work commences on the ticket, a new STATE element would be added to indicate that it has moved to a new phase of the process, such as ANALYSIS.

Once in this state, ACTIONS would be added, indicating the steps that have been taken to attempt to root cause the issue.

When ANALYSIS has completed, the state may move to CORRECTIVEACTION when ACTIONS are taken to fix the issue. Alternatively, the ticket could be placed ONHOLD, until there are resources available to carry out the work.

If it were deemed that a ticket is no longer valid, then it would be moved into a state of ABORTED, with detail as to why it was ABORTED.

Finally, when all the work has been completed the ticket would be moved to a state of CLOSED.

### Restrictions:

1. States must be in uppercase
2. No embedded whitespace
3. No trailing whitespace.
4. There can be only 1 instance of OPEN and 1 instance of CLOSED or ABORTED.
  - a. Tickets cannot be re-opened once closed.
  - b. Tickets cannot be re-opened once aborted.
  - c. A CLOSED ticket indicates that it has been fully categorized and given a conclusion.

The default set of states of a ticket are suggested as being follows, however this is by no means a final complete list and your system may have many more states.

Possible States	Comment
OPEN	Mandatory. There must always be an OPEN state. No actions can be taken in an OPEN state.
CLOSED	Mandatory, there will always be a CLOSED state on a completed ticket
ANALYSIS	When you start you analysis phase of debug. Which should be the first phase.
CORRECTIVEACTION	When you are carrying out non investigative work.
ABORTED	If the ticket should be aborted for some reason.
ONHOLD	If the ticket is not being actively worked on, for the time being.

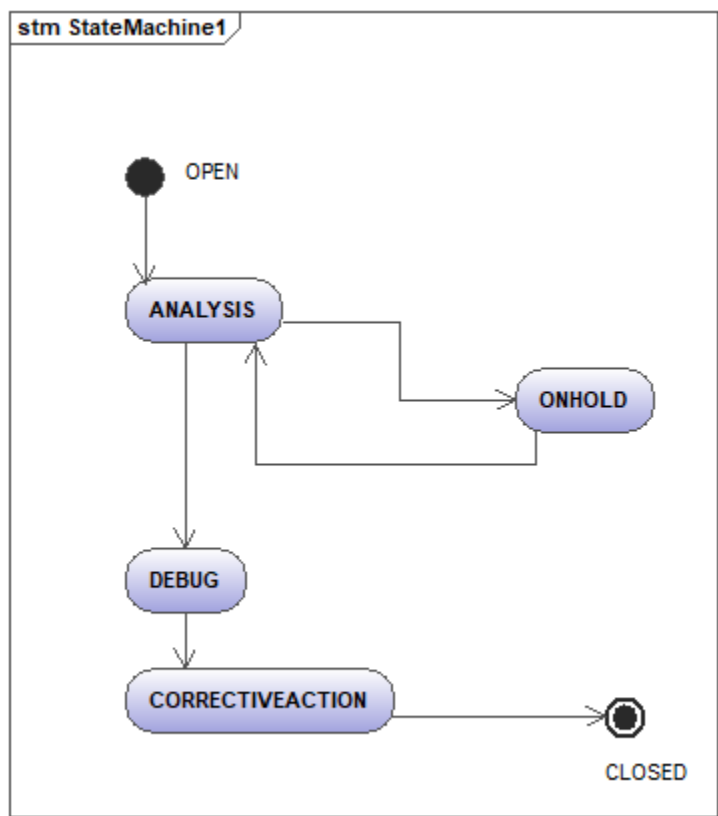
When populating the STATE elements, please ensure that the Xpath : FA/@status attribute is also populated with the present state value .

E.g.

```
<STATECHANGES>
  <STATECHANGE timeChanged="2021-07-28T15:35:36.057-05:00" fromState="INITIAL"
    toState="OPEN" user="stuart@oracle.com" id="0">
    <REASON>Initially Created</REASON>
  </STATECHANGE>
  <STATECHANGE timeChanged="2021-07-28T15:35:37.976-05:00" fromState="OPEN"
    toState="ANALYSIS" user="matt@oracle.com" id="1">
    <REASON>Attempting to look into the issue in more detail.</REASON>
  </STATECHANGE>
  <STATECHANGE timeChanged="2021-07-29T06:25:56.428-05:00" fromState="ANALYSIS"
    toState="CLOSED" user="stuart@oracle.com" id="2">
    <REASON>issue was related to power within shop floor.</REASON>
  </STATECHANGE>
</STATECHANGES>
```

Within the ticket, the FA/@status would be set in this case to the last entered state i.e. CLOSED. It will always have the value of the last FA/STATECHANGES/STATECHANGE/@toState.

The states of a ticket are dependent on the FA process being adopted. For example, if we look at a set of possible states of a ticket in the next diagram.



These states are merely included for example purposes. State names are dependent on the Oracle engineers and the FA system adopted at your local site. .

We can see that the ticket moves to ONHOLD and back to ANALYSIS before moving into DEBUG and ultimately CORRECTIVEACTION. However, all tickets will have an OPEN and CLOSED, and there can be no actions carried out within the OPEN and CLOSED state.

You must have at least one state between OPEN and CLOSED as TDMS expects ACTIONS to be assigned to STATES.

Coupled with this, a ticket cannot enter the CLOSED state multiple times. Once CLOSED a ticket cannot be re-opened.

## 5 THE TICKET ID

Your ticket ID can consist of any alpha numeric character combination, coupled with – (dash) or \_ (underscore ). All other characters should be avoided.

The max length of this id should be 32 characters, with no whitespace whatsoever. (Note that this is specified within the XSD as collapsed).

It is suggested that you prefix your ID with an identifier of sorts to make it easier to locate your specific tickets generated by your system.

## 6 ACTION ELEMENT

When a ticket is moved into a STATE, there should be N number of ACTION elements generated to audit the process you took to fix the issue.

Action types are identified within an ELEMENT using the xsi:type attribute. If we take a sample action such as a **swapAction**, the following XML would be generated.



```

<ACTION effective="N"
  datetime="2022-03-31T08:41:14.39-05:00"
  type="INVESTIGATIVE"
  summary="Swapped P1/D4 to P0/D6"
  executedBy="muhammad.khan@foxconn.com"
  state="ANALYSIS"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="swapAction">
    <DESCRIPTION>Fault fault.memory.intel.memory_ch.training-failed
      on FRU /SYS/MB/P1/D4 with 100% certainty + 2 other fault(s).
      See the log.
    </DESCRIPTION>
    <FROMPART refLocation="/SYS/MB/P1/D4" partNumber="8201157"
      serialNumber="80AD012206550B5B67" category="DIMM"/>
    <TOPART refLocation="/SYS/MB/P0/D6" partNumber="8201157"
      serialNumber="80AD012206550B5B7B" category="DIMM"/>
  </ACTION>.

```

In the above example, the highlighted section is what is expected to be populated for all actions within a ticket.

Every action executed should be recorded within the ticket, and additionally effectiveness. This effective attribute may have a value of Y or N (or true and false) and indicates what was and was not successful during debug/fix. By recording all actions, we have a complete picture as to how the issue should be tackled in the future if again encountered. (This also assists Oracle in understanding issues associated with its products).

The **state** of action is determined by the present STATE of the ticket when the action took place. Thus, if we are a state of ANALYSIS, any action generated within this state will have the attribute set to this value.

Within the **state**, the type of action being carried out is depended on what is being requested by the Oracle product. However, if no values are provided then this attribute should be set to the same value as the state attribute. In the above you can see that the type is INVESTIGATIVE as part of the ANALYSIS phase.

The next area within the XML is used for defining the type of action that this XML represents.

```

<ACTION effective="N"
  datetime="2022-03-31T08:41:14.39-05:00"
  type="ANALYSIS"
  summary="Swapped P1/D4 to P0/D6"
  executedBy="muhammad.khan@foxconn.com"
  state="ANALYSIS"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="swapAction">
    <DESCRIPTION>Fault fault.memory.intel.memory_ch.training-failed
      on FRU /SYS/MB/P1/D4 with 100% certainty + 2 other fault(s).
      See the log.
    </DESCRIPTION>
    <FROMPART refLocation="/SYS/MB/P1/D4" partNumber="8201157"
      serialNumber="80AD012206550B5B67" category="DIMM"/>
    <TOPART refLocation="/SYS/MB/P0/D6" partNumber="8201157"
      serialNumber="80AD012206550B5B7B" category="DIMM"/>
  </ACTION>

```

Due to the fact that there is a conflict between the type attribute and the element class type, you are required to identify a further type attribute, prefixed with the namespace of xsi. This is called the Schema instance namespace and needs to have a formal declaration. Thus, you will have to define the namespace itself.

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

The various types are detailed within the next section of the document, including the various elements required for each type.

Next within the ACTION element, we will define a DESCRIPTION of what the action entailed, then we will go on to define the actual parts that were swapped.

It is important that the **FROMPART** and **TOPART** elements are populated with the exact part details as this allows inventory to be tracked, as well as the various categories of parts.

### 6.1.1 Part Categories

In the above example you can see DIMM has been used, however, please note that it is Oracle's responsibility to provide a list of the various high level categories of parts, where a part should be an uppercase xs:String, with no whitespace.

## 6.2 Action Types.

Within the TDMS FA system, there are a set of actions which can be carried out to correct a failure.

The Action is intended as a placeholder for details on everything that was done to root cause and correct a fault. Actions that had both had a positive effect, and those which did not should be recorded, providing a true picture of the technicians steps.

The Action type has many attributes, which are dependent on the xsi:type being detailed. However, the following attributes must be defined for all Actions.

Attribute	Type	Comment
ACTION/description	String	Detailed description about the action
ACTION/@summary	String	Concise summary of the action
ACTION/@datetime	Date/time	When the action took place
ACTION/@effective	Boolean	Was the action effective
ACTION/@executedBy	String	The person who carried out the action
ACTION/@type	String	Type as: ANALYTIC, CORRECTIVE

The following actions are expected during carrying out failure analysis

### 6.2.1 Type : **repairAction**

**Description :** Use when a general repair of a part is being carried out, which cannot be categorized under any other Action.

Attribute	Type	Comment
ACTION/FROMPART/@serialNumber	String	Serial number of the repaired part.
ACTION/FROMPART/@partNumber	String	Part number of the repaired part
ACTION/FROMPART/@location	String	The location that the part resides in, which was repaired.

### 6.2.2 Type : **reseatAction**

**Description :** When a component is reseated into its slot or location.

Attribute	Type	Comment
ACTION/FROMPART/@serialNumber	String	Serial number of the repaired part.
ACTION/FROMPART/@partNumber	String	Part number of the repaired part
ACTION/FROMPART/@location	String	The location that the part resides in, which was repaired.

### 6.2.3 Type : **reruntestAction**

**Description :** When the system is being retested as part of debug, or verifying that the corrective action was successful.

Attribute	Type	Comment
ACTION/logop	String	The operation that was executed
ACTION/ocation	String	The location that the system was retested at
ACTION/testResultStatus	String	The result of the test
ACTION/guti	String	The global identifier
ACTION/teststarttime	Date/Time	The teststarttime of the testresult.
ACTION/testResultStatus	String	The result of the test
ACTION/location	String	Where the test took place.

### 6.2.4 Type : **addAction**

**Description :** If you were to add a new part, or missing part to the system..

Attribute	Type	Comment
ACTION/FROMPART/@serialNumber	String	Serial number of the repaired part.
ACTION/FROMPART/@partNumber	String	Part number of the repaired part
ACTION/FROMPART/@location	String	The location that the part resides in, which was repaired.

### 6.2.5 Type : **removeAction**

**Description :** Use when you are removing a part from the system.

Attribute	Type	Comment
ACTION/FROMPART/@serialNumber	String	Serial number of the repaired part.
ACTION/FROMPART/@partNumber	String	Part number of the repaired part
ACTION/FROMPART/@location	String	The location that the part resides in, which was repaired.

### 6.2.6 Type : **swapAction**

**Description :** Use when two similar parts are being swapped , such as DIMMS which could be swapped between 2 slots.

Attribute	Type	Comment
ACTION/FROMPART/@serialNumber	String	Serial number of the repaired part.
ACTION/FROMPART/@partNumber	String	Part number of the repaired part
ACTION/FROMPART/@location	String	The location that the part resides in, which was repaired.
ACTION/TOPART/@serialNumber	String	Serial number of the repaired part.
ACTION/TOPART/@partNumber	String	Part number of the repaired part
ACTION/TOPART/@location	String	The location that the part resides in, which was repaired.

### 6.2.7 Type : **partReplacementAction**

**Description :** Use when a part appears faulty and you are completely replacing. This could also be achieved using the REMOTE and ADD actions

Attribute	Type	Comment
ACTION/FROMPART/@serialNumber	String	Serial number of the repaired part.
ACTION/FROMPART/@partNumber	String	Part number of the repaired part
ACTION/FROMPART/@location	String	The location that the part resides in , which was repaired.
ACTION/TOPART/@serialNumber	String	Serial number of the repaired part.
ACTION/TOPART/@partNumber	String	Part number of the repaired part
ACTION/TOPART/@location	String	The location that the part resides in , which was repaired.

### 6.2.8 Type : **abortAction**

**Description :** When a ticket is being aborted , the details as to when and why should be recorded as an action, though optional , since the STATE element should record the moving of STATE to ABORTED. .

Attribute	Type	XPath	Comment
ACTION/Description	String		Detailed description about the action
ACTION/Summary	String		Concise summary of the action
ACTION/actionDate	Date/time		When the action took place
ACTION/effective	Boolean		Was the action effective
ACTION/executedBy	String		The person who carried out the action
ACTION/type	String		Type as : ANALYTIC , CORRECTIVE

### 6.2.9 Type : **returntovendorAction**

**Description :** Use when a part is being returned to a vendor such that tracking information can be recorded. .

Attribute	Type	Comment
ACTION/shippingDate	Date/Time	Detailed description about the action
ACTION/shippingCompany	String	Concise summary of the action
ACTION/trackingNumber	String	When the action took place
ACTION/vendor	String	Vendor Identifier.

## 7 ASSOCIATING A TICKET TO A TEST RESULT

If we have an operation (logop), which has a failed test result automatically being sent to TDMS, it is requested that you associate this result with a FA ticket. This provides a cross reference of the failure details to the corrective action adopted.

To set this association you should populate the

## ASSOCIATEDTESTSET

Element with details of failed TESTSET within the test run.

```
<ASSOCIATEDTESTSET
  completeTime="2022-03-30T20:17:19.908-05:00"
  guti="6244D285009D0000000000000000FCTX30-D07"
  name="INITIALIZE_HOST"
  startTime="2022-03-30T19:09:35.128-05:00"
  status="F" testStartTime="2022-03-30T16:59:27.971-05:00">

  <FAILUREMESSAGE>
    Fault fault.memory.intel.memory_ch.training-failed on FRU /SYS/MB/P1/D4 with
    100% certainty + 2 other fault(s). See the log.
  </FAILUREMESSAGE>

</ASSOCIATEDTESTSET>
```

Figure 3 Associated Testset xml

As can be seen, the details provide TDMS with enough information to identify an existing failed TESTSET.

The attributes are defined as follows.

ATTRIBUTE	TYPE	DESCRIPTION
<b>completeTime</b>	Date / Time to ms	The completion time of the test set which failed. le : //TEST/PROCESS/TESTSUITE/@completeTime
<b>guti</b>	String	The GUTI of the Test result to which the failed test set belongs le: //TEST/@guti
<b>name</b>	String	The name of the test set that failed (Note that it is assumed testset names are unique within a test result . If not the case then this association cannot be made and the xml is deemed illegal). //TEST/PROCESS/TESTSUITE/TESTSET/@name
<b>status</b>	String	The status of the result, which can be F or A (failure or aborted). //TEST/PROCESS/TESTSUITE/PASS or FAIL or ABORTED elements since these are mutually exclusive.
<b>startTime</b>	Date/Time to ms	The start time of the failed test set. I.e. //TEST/PROCESS/TESTSUITE/TESTSET/@startTime
<b>testStartTime</b>	Date/Time to ms	The test start time of the Containing test result. This starttime can be located within : //TEST/PROCESS/TESTSUITE/@startTime

Additionally, the failure message element within the FA element should reflect the same failure message as the TESTSET. (if there were no associated test set on a ticket, i.e. from an operation which does not generate a result, then the top level //FA/FAILUREMESSAGE would be the only element populated ).

i.e.

```
<ASSOCIATEDTESTSET
  completeTime="2022-03-30T20:17:19.908-05:00"
  guti="6244D285009D00000000000000000000FCTX30-D07"
  name="INITIALIZE_HOST"
  startTime="2022-03-30T19:09:35.128-05:00"
  status="F" testStartTime="2022-03-30T16:59:27.971-05:00">

  <FAILUREMESSAGE>
    Fault fault.memory.intel.memory_ch.training-failed on FRU
    /SYS/MB/P1/D4 with 100% certainty + 2 other fault(s). See the log.
  </FAILUREMESSAGE>

</ASSOCIATEDTESTSET>

<FAILUREMESSAGE>Fault fault.memory.intel.memory_ch.training-failed on FRU
/SYS/MB/P/D with % certainty + other fault(s).See the log.</FAILUREMESSAGE>
```

Figure 4 Associated test set xml

Attribute	Type	Comment
FA/ASSOCIATEDTESTSET/@completeTime	Date/Time	Complete time within the testset result. ie TEST/TESTSUITE/TESTSET/@completeTime
FA/ASSOCIATEDTESTSET/@guti	String	The guti from the test result TEST/@guti
FA/ASSOCIATEDTESTSET/@name	String	The name of the test set * testset names must be unique )
FA/ASSOCIATEDTESTSET/@startTime	Date/Time	The starttime of the testset ie TEST/TESTSUITE/TESTSET/@startTime
FA/ASSOCIATEDTESTSET/@status	String	The status of the testset, Which should be either F or A.
FA/ASSOCIATEDTESTSET/@testStartTime	Date/Time	The start time of the encoding parent test result : TEST/TESTSUITE/@startTime

## 8 TICKET CATEGORIZATION

Once you have completed a ticket, you must categorize the fault encountered. This can be achieved within the FA schema by defining the fault within a hierarchy of up to 4 levels.

e.g.

HARDWARE > PRODUCT > CABLE > MECHANICAL

FA@category > FA/@type > FA/@fault > FA/@subFault

Coupled with the categorization, you should also provide a FA/CONCLUSION element which is a free form text providing a brief synopsis as to what the root cause and corrective action was. This provides a quick reference for anyone reviewing the ticket.

## 9 CROSS REFERENCE TICKETS

If you were to have the need to cross reference a ticket, such as to indicate a false failure if a system were to fail but it was caused by the failing of a sibling under test. Then you can reference the ticket using the attribute

`crossReference`

within the ticket. This should be populated with the ticketed which references the true failed system. (i.e. the overall cause of the failure if we were batch testing or solution testing).

## 10 SOLUTION LEVEL TICKETS

When testing a solution (where a solution consists of various disparate systems, switches, servers etc.), it is likely that a failure of one component will influence other areas of the solution.

*For example, if we were to have a switch failure on a solution, then many systems may fail due to the switch failure.*

If it were required to record a ticket on every failure within the solution (including the overall solution itself) then you would create a ticket for the solution being tested, and then for each failure.

On each system that failed you would set the attribute:

`FA/@parentTicketId`

This would point to the top-level ticket raised for the solution which failed.

Additionally, if there was no ticket raised on the parent, you can assign the serial number of the solution by setting the `FA/@parentContainerId` attribute. This attribute indicates that the serial number being debugged is within a containing system of sort.

During the debug process you will eventually find the root cause of the failure. This may be located within one of the many failures recorded. All other failures were a consequence of the failing component. These are victims of the failure.

To model this within the ticket structure, you should set the attribute

`FA/@crossRerence`

And populate with the ticketed of the overall cause of the failure.

This would create a structure like the following.



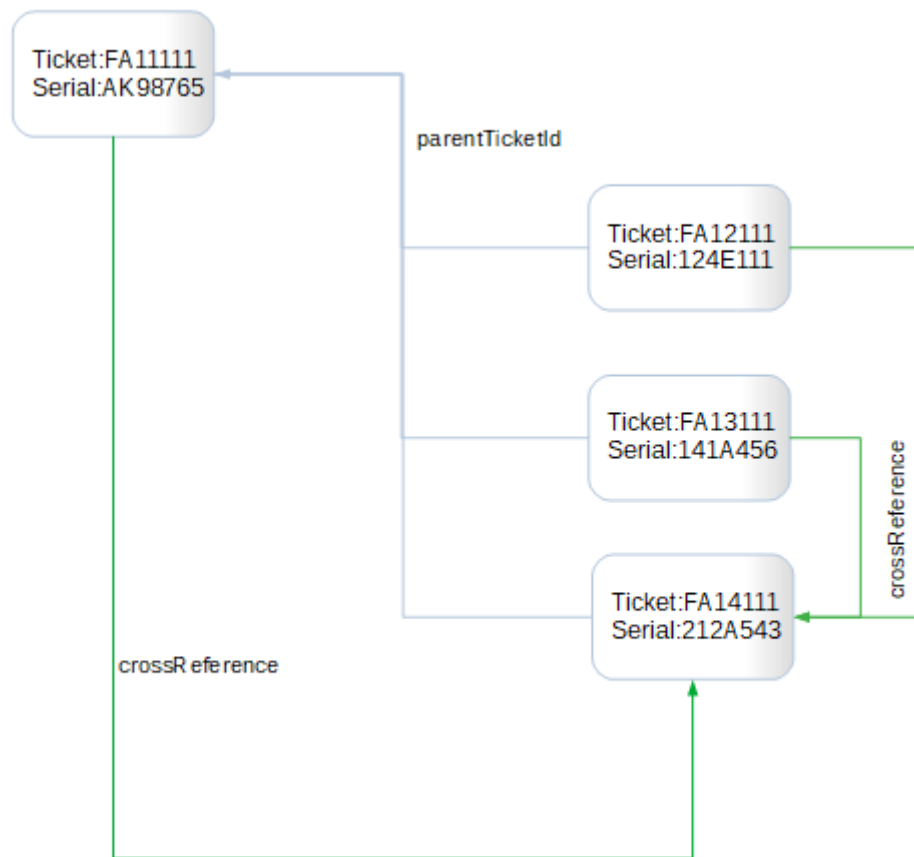


Figure 5 Ticket hierarchy

## 11 ACCOMPANYING JAR FILE.

It is possible to associate a jar file with your ticket such that you can add attachments such as documents, png or jpg images etc. The purpose of the file is to add supporting documentation to the ticket such that the issue and solution are clearly defined.

When attaching the jar file (see next section), you should provide a link to the file using the attribute

@FA/@archiveName

This all inform TDMS that there is an additional file, and what name to look for within the collected jar file.

## 12 PACKAGING THE TICKET BUNDLE

Once you have your ticket and your accompanying jar file populated with supporting material, you are required to place this into a hierarchical jar structure.

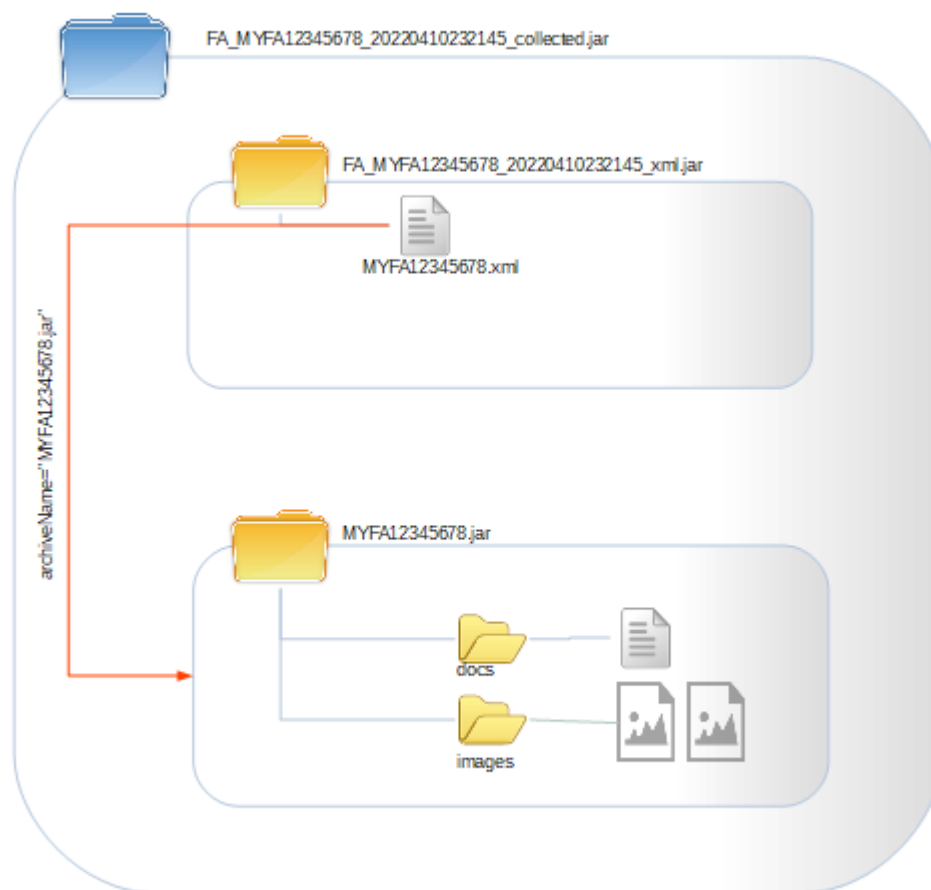
1. **Create your XML file**  
take your XML place into a file named <ticketid>.xml . for our example the ticket id will be MYFA12345678
2. **Place XML file into jar**  
Place you MYFA12345678.xml file into a jar file , named  
FA\_MYFA12345678\_YYYYMMDDHHMMSS\_xml.jar



3. **Add Additional jar if required**  
If it is required to provide additional information with the FA , then place this into a jar file named <ticket>.jar .
4. **Create the collected jar file.**  
Place both your xml.jar and your accompanying jar file into a container jar called FA\_MYFA12345678\_YYYYMMDDHHMMSS\_collected.jar

*Note : The manifest within the jar is not required.*

The final structure of your FA collected jar file should be similar to the following.



*Figure 6 structure of Jar file*

You are now ready to forward this file to TDMS for processing. Please reference the 911-2411 document for details on how to transfer data to TDMS for processing.

## 13 SENDING THE TICKET TO ORACLE

### 13.1 Cloud PUSH

When you are a remote site with no local instance of the TDMS environment , you can utilise the Oracle S3 cloud bucket service to push results to Oracle.

Oracle will provide an endpoint and secret key details which you can then use to connect to the S3 service using whatever your preferred cloud client technology is.

Oracle will provide a bucket that you will use to push your FA data . Note though that the format of the data should be the same as sending a test result ( see document : 911-2411 ).

## 13.2 Rest Service Upload

If you should have a local instance of TDMS , you can push your collected jar file to the Oracle TDMS upload service .

This is a REST service which should be invoked as follows.

Service		
URL of provider	<a href="https://&lt;tdmsServer&gt;/COLLECTOR">https://&lt;tdmsServer&gt;/COLLECTOR</a>	This is the application providing the service.
Service	rest/dataProcessing/upload/receptionArea/{targetArea}	Service to invoke
Param	targetArea	This will be provided by y TDMS however by default, this should be set to FA
Content type	Multipart form	The form should include a param called file which will contain the filename being uploaded.
Response	Application/json	
Response Codes	404 201 503 500 401	Unknown area provided SUCCESS Service offline Parse error sender ip is blocked.

## 14 FREQUENCY OF DELIVERY

It is expected that you push the details of an FA ticket to Oracle at any major update, such as moving state or creating key ACTIONS.

Each time you push the ticket, the old details will be removed from TDMS , and a complete reload will take place. Thus, it important that your ticket show the complete audit trail each time it is sent.

Note : The points within the lifecycle of a ticket where an update should be sent to Oracle will be dictated to by the Oracle site /Product engineers.

## Revision History

REVISION	MODIFIED BY	DATE	COMMENT
2	N/A	5 <sup>th</sup> April 2022	Initial Release.
3	N/A	8 <sup>th</sup> June 2023	Modification to the type definitions when creating an action. Discriminator added. Details on hierarchical tickets.

- When Document Template is complete, email source file to [eso\\_business\\_docs\\_us\\_grp@oracle.com](mailto:eso_business_docs_us_grp@oracle.com)
- All hard copies of this document are uncontrolled and are to be used for reference only.
- For questions or comments about this document, please send an email to: [eso\\_business\\_docs\\_us\\_grp@oracle.com](mailto:eso_business_docs_us_grp@oracle.com)