|  |
| --- |
|  |
| **Title\*:** | Ericsson’s proposals to resolve issues raised to TDL MM draft |
|  | 04 February 2015 |
| from **Source**\*: | L.M.Ericsson |
| Contact: | György Réthy  |
|  |  |
| input for **Committee**\***:** | MTS |
|  |  |
| Contribution **For\*:** | Decision | **X** |  |
|  | Discussion |  |  |
|  | Information |  |  |
|  |  |
| Submission date**\***: | 2015-02-04 |
|  |  |
| Meeting & Allocation: | [**MTS-TDL\_Steering Group meeting**](http://webapp.etsi.org/MeetingCalendar/MeetingDetails.asp?mid=16052) **on 05th February 2015** |
|  |

Please find below the result of the analysis and proposed solutions to the issues raised by Siemens in an email following MTS#64. For completeness the whole email text is attached at the end of this TD.

***1) A sentence in clause 9.4.6 Interaction***

**Issue raised**: The statement "The occurrence of the <undefined> value within the 'DataUse' specification of 'argument' causes undefined semantics of the 'Interaction' behaviour at runtime." has the same meaning as stating that the argument shall be fully specified.

**Conclusions**:

- The sentence can be understood as if <undefined> value in the argument of an Interaction \***caused in all cases**\* an undefined runtime semantics of the Interaction itself. This is certainly not the case. An <undefined> value, just like e.g. an \_*incorrectly defined concrete*\_ value, may cause an unexpected runtime behaviour of the SUT. But this is an error in the TDL specification or in the underlying infrastructure, not a direct consequence of the existence of <undefined>.

- This concrete question has been discussed and decided by TB MTS to allow partially defined messages.

**Solution**:

The above **sentence** has been deleted in paragraph 5 of clause 9.4.6 and re-inserted again as paragraph 7. It shall be **deleted** in the latest final draft as well; adding a note, warning the users though may be useful (strikethrough font is by Gyorgy, in the draft the text is in normal font):

“~~The occurrence of the <undefined> value within the 'DataUse' specification of 'argument' causes undefined semantics of the 'Interaction' behaviour at runtime.~~

NOTE: The  <undefined> value within the 'DataUse' specification of 'argument' is resolved outside of the TDL specification that may leave to an unexpected runtime behaviour of the SUT.”

***2) Semantics of OmitValue***

**Issue raised**: ”It explains the meaning of OmitValue only in context of optional members in a structured data type. However OmitValue can be assigned indirectly also to variables or other mandatory members at runtime. The provision of static semantics rules (constraints) is not sufficient to forbid this case.”

**Conclusion**: Let OmitValue be handled like other special values within the TDL specification. This would also allow checking if it is assigned to a member, parameters or variable.

**Solution**:

- Add a the sentence to clause 9.4.6 Interactions:
“In the 'DataUse' specification, which is the 'argument' property of the 'Interaction ', 'OmitValue' shall be assigned optional 'Member's only (see clauses 6.3.1 and 6.3.8).'

***3) Assigning OmitValue to variables and mandatory fields runtime***

**Issue raised**: “needs to be stated in the relevant clauses what it means when OmitValue is assigned to a variable or a mandatory member at runtime”

**Conclusion**: Due to the resolution of the previous comment, OmitValue is just an ordinary value like any other data instance or specific value.

However, the newly introduced requirements to clauses 6.3.10 and 6.3.12 has net been discussed and agreed by the TDL Technical session on 28th January, and they are technically not correct. For example, an **rnd(in seed) return float** functions’s behaviour why would become undefined, if no seed is passed to it (seed is <unspecified>)? Why would it be unable to return a random value?

**Solution**:

 The following changes are required in MTS(15)64\_003r4:

- In clause **6.3.10** FunctionCall **delete** **the newly inserted sentence** (strikethrough font is by Gyorgy, in the draft the text is in normal font):
“~~If one of the 'DataUse' specifications provided in the 'ParameterBinding' of the 'FormalParameter's has the value <undefined>, the 'FunctionCall' represents also the <undefined> value.~~”

- In clause **6.3.12** VariableUse **delete the newly inserted sentence** (strikethrough font is by Gyorgy, in the draft the text is in normal text):

“~~If the accessed 'Variable' has the value <undefined> assigned to it, the 'VariableUse' represents also the <undefined> value.~~”

***4) Checking if an optional member is omitted or not***

**Issue raised**: ” A user needs to have the possibility to check at runtime whether a variable or a member have the OmitValue assigned to it…”

**Conclusion**: Solved by moving the restriction from clause 6.3.8 to the Interactions clause (see above).

**Solution**: No change is needed.

