Quality Indicators for Automotive Test Case Specifications

Katharina Juhnke
Daimler AG
Group Research & MBC Development
Ulm, Germany
Email: katharina.juhnke@daimler.com

Matthias Tichy
Ulm University
Institute of Software Engineering and Programming Languages
Ulm, Germany
Email: matthias.tichy@uni-ulm.de

Frank Houdek
Daimler AG
Group Research & MBC Development
Ulm, Germany
Email: frank.houdek@daimler.com

***Abstract*—Testing is an important quality assurance activity during development of automotive software. Automotive OEMs and suppliers use test case specifications to specify, mostly informal, test cases as well as supporting information like traces to requirements. While the quality of the test case specifications has a high influence on the quality of the subsequent testing, quality of informal automotive test case specifications have not been investigated yet. In this paper, we present 7 potential quality indicators, ranging from requirement coverage to contents of a test step. The quality indicators have been identified in a case study of 816 current test case specifications specified by an OEM and suppliers.**

***Index Terms*—Automotive software testing, test case specification, quality indicators**

I. Introduction

A lot of innovation in the automotive domain is nowadays addressed by software and electronic systems. Solid testing processes are an integral part of the development process to verify that the implemented software works as expected. Standards like ISO 26262 [1] or Automotive SPICE [4] require a consistent test documentation. An essential part of the test documentation is the test case specification, which is defined by the software testing standard ISO 29119 [2]. The test case specification contains a set of test cases derived from the test basis for a particular test object [3]

Failures and misinterpretation of the test case specification shall be avoided at any time. While techniques like mutation testing can be used to assess the quality of automated tests [8], automated approaches to assess the quality of informal test case specifications are scarce [5] or restricted to domain-specific test languages, i.e. for Testing and Test Control Notation (TTCN-3) [6], [7]. In order to identify potential areas of improvement for automotive test case specifications, the aim of this paper is to examine how the quality can be evaluated using a given test case specification template.

The analysis is based on data extracted from the test case specifications suitable as indicators for quality assessment of test case specifications. The analysis provides insights into the specification of automotive test cases and identifies whether there is sufficient formalization within these test case specifications to enable an automated evaluation of the quality.

As data source, a total of 816 test case specifications from a total of 16 diverse projects have been collected from an automotive OEM. The included test case specification were either created in-house or by suppliers.

In Section II, we give an overview about test case specifications including an example. Section IV contains a description of potential quality indicators resulting from the analysis including quantitative data on selected test case specifications. Thereafter, we conclude and present some outlook on future work in Section V.

II. Automotive Test Case Specification

Test case specifications are a central part of the test documentation [2] in the automotive environment. They are used to document the test cases to be performed. A test case specification contains a set of test cases that are necessary to adequately test a particular test object according to defined test objectives. A test case is basically characterized by a unique identifier, pre- and postconditions, inputs, expected results, priority for the test execution and traceability information (e.g. references to the associated requirements) [2]. In addition to these *test case basic attributes*, other domain or company specific test case attributes are often specified such as status, test objective, author, model series or test platforms. These attributes are called *test case meta data.*

Test case meta data and basic attributes are relevant for a test case and can be defined in a test case specification template. An example of an automotive test case and the application of a test case specification template is shown in Figure 1. The test case of a wiper and wash system is described by its basic attributes and automotive specific test case meta data (e.g. vehicle family, test platform). The test case attributes are represented by the columns and the rows are used to define different object types such as text, headlines, test cases or test steps. Not all attributes are relevant for each object type, so some cells shall be empty (dark gray cells in Fig. 1). Different object types have a hierarchical relationship to each other. For instance test steps must be assigned to a test case.

ID	Object Type	Object Text	Precondition	Action	Expected Result	Postcondition	Priority	Requirement
WWC-TS-123	*test case*	**Switch on KL15R**	- ISw_Stat = IGN_OFF - FWS = OFF and RWS = OFF - Front and Rear Wiper in Parking Position - WIPERCONTROL_ACTIVE = 0 - Gearbox Postion = P - Rain sensor coded, no rain			- None	2	WWC-REQ-567, WWC-REQ-765
WWC-TS-124	*test step*	Step 1		- FWS = 0/3 - RWS = 1	- WIPERCONTROL_ACTIVE = 0 - Front and rear wipers must not start wiping			
WWC-TS-125	*test step*	Step 2		- ISw_Stat = IGN_ACC	- WIPERCONTROL_ACTIVE = 1 - Rear wiper must start wiping in continuous wipe mode 1 as soon as IGN_ACC has accepted			
WWC-TS-126	*test step*	Step 3		- ISw_Stat = IGN_OFF	- WIPERCONTROL_ACTIVE = 0 - Wipers must stop as soon as IGN_OFF has accepted (delay max. 20 ms)			

(a) Test case basic attributes

ID	Object Type	Object Text	...	Test Goal	Release	Workflow State	Vehicle Family	Test Platform
WWC-TS-123	*test case*	**Switch on KL15R**	...	Functionality	2.0	Approved	BRXYZ	Vehicle HiL
WWC-TS-124	*test step*	Step 1	...					

(b) Test case meta data attributes (dashed border)

Fig. 1. Example of an automotive test case definition using a test case specification template

III. Data Collection

We collected data from a IBM Rational DOORS database of an automotive OEM and identified a total of 2435 test case specifications. Thereafter, obsolete test case specifications have been excluded reducing the number of test case specifications to 972. Obsolete test case specifications are those not based on the current test case specification template or whose last modification date is before 2015. Furthermore, duplicates, backups, or test case specifications marked as obsolete by name have not been included resulting in a further reduction to 816 test case specifications from a total of 16 diverse projects. Projects are related to vehicle domains such as powertrain, chassis or comfort systems.

The identified test case specifications based on a test case specification template as shown in Figure 1. The test cases are derived from natural language requirements and therefore the relevant test case attributes are also specified by using natural language.

We analyzed the collected data quantitatively based on information that can be determined programmatically. Therefore, we used the DOORS Extensible Language (DXL) to extract quantitative data from the identified test case specifications.

IV. Potential Quality Indicators

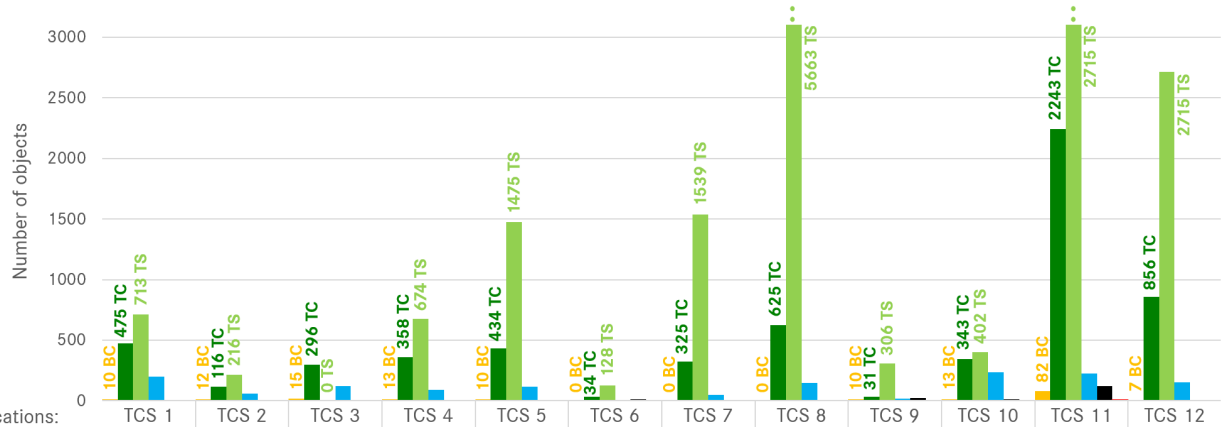
The results of the analysis are presented in detail based on a representative project. The selected project includes a total of 12 test case specifications from the powertrain domain that are representative for automotive test case specifications. Figures 2 - 5 show the analysis results for this project. In the following, the identified quality indicators are discussed on the basis of the various criteria examined.

A. Criterion 1: Size of the test case specification with respect to requirement specification.

The average number of test cases in a test case specification is 511 (cf. number of test cases in Fig. 2). However, very large test case specifications contain more than 2200, small ones can contain only 31 test cases (cf. Fig. 2). However, the size of the test case specification is not very meaningful and does not make any statement about the correctness of the test case. Therefore, they must be considered in correlation to the testable requirements specified in the corresponding requirement specification. For instance, a test case specifications with 856 test cases (cf. Fig. 2, TCS 12) can be classified as large. The requirement specification associated with test case specification 12 contains 2462 testable requirements. It is recommended that one requirement shall be tested by at least one test case. Therefore, it is obvious that there should be more test cases to verify 2462 related requirements. However, the relationship between the size of the test case specification and the number of testable requirements can only be an indicator of the completeness of the test case specification. Reference values for an appropriate size of a test case specification can be determined by previous test case specifications of a system.

B. Criterion 2: Distribution of contained object types.

Test case specifications contain a large number of test cases and test steps, as shown in Figure 2 by the green bars. Objects of type base scenario indicate reusable preconditions that can



Test case specifications:	TCS 1	TCS 2	TCS 3	TCS 4	TCS 5	TCS 6	TCS 7	TCS 8	TCS 9	TCS 10	TCS 11	TCS 12
base scenarios (BC)	10	12	15	13	10	0	0	0	10	13	82	7
test cases (TC)	475	116	296	358	434	34	325	625	31	343	2243	856
test steps (TS)	713	216	0	674	1475	128	1539	5663	306	402	4046	2715
text (T)	201	57	123	88	118	9	51	146	18	237	227	151
undefined (U)	0	0	7	2	0	10	8	0	21	11	120	0
not template compliant (N)	0	0	0	0	0	0	0	0	0	0	1	0

Fig. 2. Number of test cases (dark green) and other contained object types per test case specification

be referenced by several test cases. The analysis revealed that there is often only a very small number of base scenarios in the test case specifications or that base scenarios are not used at all. Instead, preconditions are often copied from test case to test case instead of reusable constructs being stored and referenced in base scenarios. This is particularly noticeable in large test case specifications. By using base scenarios, duplicates could be reduced and the reuse of established constructs can be increased. Reduction of duplicates also minimizes the amount of time and effort needed to make changes in the reused parts, since the changes can then be made centrally. Test case specifications also contain undefined objects to which no object type is assigned (e.g. TCS 11, Fig. 2). A small number of test case specifications contain object types that are individual and not predefined by the template. This is a violation of the template, which can have an effect on export to downstream tools.

C. Criterion 3: Size of test cases.

The average size of a test case, measured in terms of the number of test steps to be performed, is 3,83 test steps. However, test cases with up to 76 test steps (see TCS 8, Fig. 3) could also be identified in the representative project. Large test cases are more error-prone in the later test execution and make debugging more difficult. For instance, it is time-consuming to execute 76 test steps manually in order to reproduce a failed step. In addition, an excessive number of test steps can be an indicator that several test cases have been combined into one large test step. Hence, it makes sense to split them up in several test cases.

D. Criterion 4: Type of test case specification.

Test cases usually have test steps that structure the test case flow. This has the advantage that very large, extensive and coherent test procedure descriptions are avoided. It also supports the assignment of documented expected results to a clearly defined set of inputs and actions. For very small test cases, the template used for the analyzed test case specifications allows the definition of actions and expected results without using test steps. The analysis revealed that test case specifications exist when test steps are omitted completely (see TCS 3, Fig. 3). In such cases, it can often be observed that the inputs and expected results are overloaded. Several pairs of inputs and expected results are combined in one test step. That means that it is no longer possible to unambiguously correlate the inputs to the expected results. This is also the case when test case specifications use both approaches. In most cases, a mixture of both approaches is found within a test case specification (e.g. 9 of 12, Fig. 3). This form is more error-prone and makes it difficult to understand.

E. Criterion 5: Type of linked object types.

The analysis of the linked object types revealed links between test cases and artifacts. The table in Figure 5 shows a linkage scheme with allowed linkages. In general, these are external links (e.g. artifacts are requirements whose correct implementation shall be verified, see TCS 8 in Fig. 4) or internal links, i.e artifacts such as base scenarios which are referenced by a test case, see TCS 11 in Fig. 4). In some cases no requirements have been linked (cf. TCS 5, 6 and 7 in Fig. 4). In such cases, there is an insufficient requirement coverage. The link analysis can also be used as quality indicator to detect errors in the documented requirements, i.e. if the object type is not set for a requirement (undefined links, e.g. TCS 6, 7 and 11 in Fig. 4) or the requirement specification is not based on a standard template (unknown links, e.g. TCS 5 in Fig. 4). In the case of the considered test case specifications and the linked

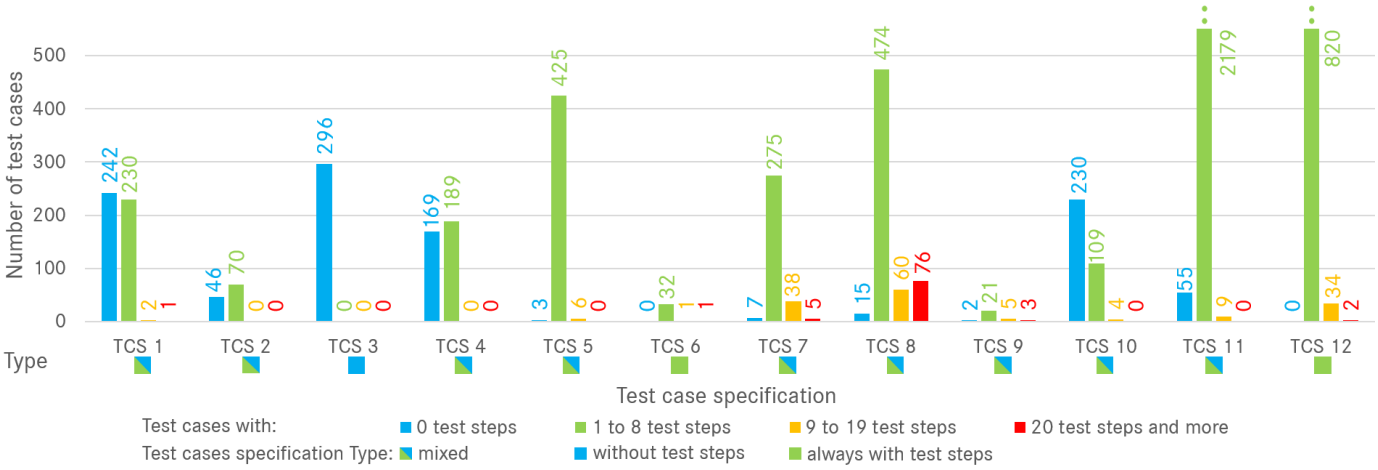


Fig. 3. Number of test steps per test case and type of test case specification

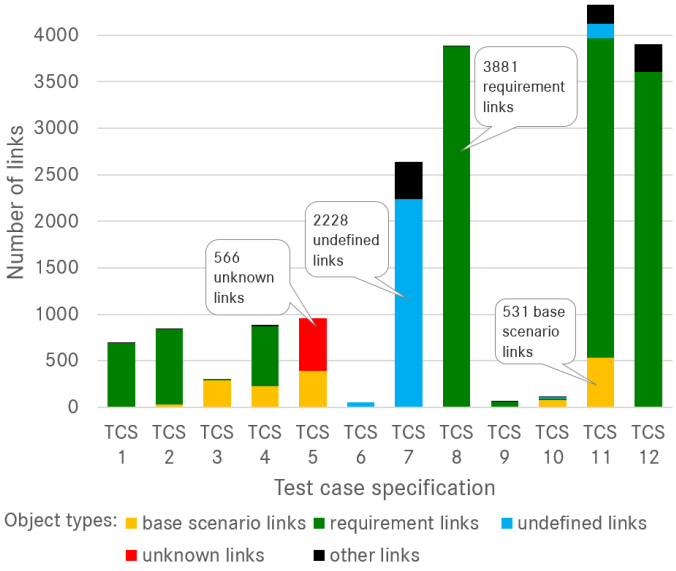


Fig. 4. Number of links according to the object types of the targets for each test case specification

Test case specification object type	Target artifact object type: text	base scenario	test case	test step	requirement	other	undefined	unknown
text	2	0	39	0	14	3	25	0
base scenario	0	12	17	0	8	28	0	0
test case	1	747	7	3	11159	719	2254	729
test step	0	782	0	1	4880	379	17	0

Linking rules: allowed links to base scenarios; forbidden links; allowed links to requirements

Fig. 5. Detailed evaluation of all links from test case objects (rows) to target objects (columns) according to the linkage scheme (table). The cells contain the total number of all links summarized from all test case specifications.

requirement specifications, the object type *requirement* must be set for requirements. Furthermore, non template compliant and incorrect links can be detected (see black bars in Fig. 4). This includes links to *headings*, *information* or *process requirements* that cannot be considered as testable in the context of a requirements-based testing approach.

F. Criterion 6: Number of linked object types.

A test case is linked to an average of 1,68 requirements. This also corresponds to the premise that each requirement should be checked by at least one test case. However, there are also enormous deviations where a test case with 121 requirements is linked. Such a high number of linked requirements can be seen as an indicator to check such a test case, either because it is too large and could be divided into several test cases or because it seems rather unlikely that a single test case can cover that many requirements. According to our experience, more than 20 linked requirements can be considered critical.

G. Criterion 7: Template conformity.

The analysis revealed several violations of the template guidelines. These have a significant negative effect on the further processing of the test case specification (e. g. automated verification mechanisms are not applicable or the export to downstream tools fails). For example, customizing the chapter structure (adding or renaming chapters) means that test cases are not included in the export or the export fails. Furthermore, it can be an indicator for the quality of a test case specification if certain mandatory attributes have not been filled (e. g. inputs, expected results, test platform, model series).

V. Conclusion

Our investigations show that test case specifications are not completely documented in accordance with the guidelines of the used test case specification template. Therefore, the root cause needs to be investigated further. Adequate formalization is required for an automated quality assessment of the test case specifications. Due to the highly individualized attribute sets of the test case specifications, a structural evaluation based on the template does not appear to be feasible for all test case specifications. It is also difficult to evaluate the content of the test cases and compare them with the contents of the requirements and the test concept programmatically, since the test cases are usually written in prose. In order to perform automated quality assessments, test cases must be formalized and template guidelines must be fulfilled. Furthermore, it should be noted that contents from the test concept and the underlying requirements play an important role in the quality assessment of a test case specification. Since this information is not available in the same data format (e.g. requirements, test cases in DOORS and test concepts in Word) an automated evaluation of compliance with the guidelines is not possible.

The analysis shows that there exist criteria which can be used as quality indicators for a first quality assessment with regard to a examination of the structure of a test case specification. This includes in particular the test case size with respect to the requirement specification, number of linked requirements, adherence to a given linking scheme or the implementation of the test objectives and test platforms defined in the test concept. These criteria can be used to gain a first impression of the quality of the test case specification. This could be a condition for whether a more detailed examination is reasonable at the given time.

Our future work will focus on the qualitative analysis of test case descriptions, which are mostly based on natural language. Mechanisms are required to support and accelerate reviews of test case specifications.

References

[1] ISO 26262, Road Vehicles - Functional Safety, 2011.

[2] ISO 29119, Software and Systems Engineering - Software Testing, 2013.

[3] ISTQB, Glossary of Testing Terms, 2015.

[4] VDA QMC Working Group / Automotive SIG. “Automotive SPICe: Process Reference Model Process Assessment Model”, 3rd ed, 2016.

[5] R. Lachmann and I. Schaefer, “Towards Efficient and Effective Testing in Automotive Software Development”, GI-Jahrestagung, 2014, pp. 2181–2192.

[6] B. Zeiss, D. Vega, I. Schieferdecker, H. Neukirchen and J. Grabowski “Applying the ISO 9126 Quality Model to Test Specifications - Exemplified for TTCN-3 Test Specifications”, Software Engineering GI, vol. 15 (6), pp. 231–242, 2007.

[7] H. Neukirchen, B. Zeiss and J. Grabowski “An Approach to Quality Engineering of TTCN-3 Test Specifications”, International Journal on Software Tools for Technology Transfer, vol. 10 (4), pp. 309–326, 2008.

[8] Y. Jia and M. Harman, “An Analysis and Survey of the Development of Mutation Testing”, IEEE Transactions on Software Engineering, vol. 37 (5), pp. 649–678, 2011.