Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

Best Practice to Design Test Cases to Improve Reusability for Automotive Body Electronics System

Seungyong.Lee, Youngheum.Kim, Byounggu.Choi, Jaekyun.Chang

System Engineering Department, CARNES corporation
Seoul, Korea, (e-mail: seungyong.lee@carnes.co.kr)

Abstract: Verification and validation are getting more important for the success of system development. This is one of the issues in automotive domain as well. Therefore, proper verification and validation methods in automotive domain are necessary to identify system malfunction in early development phase. This paper introduces best practice and framework how to design test cases for automotive body electronics system by using a strict process and tool that enable to develop test cases effectively for increasing reusability.

1. INTRODUCTION

System verification and validation become more valuable in system development project to be successful in business. One of the best practical ways is early development of test specification after the requirement specification has been released in order to increase the quality of system in early development phase. Another is standardization of procedure and method for creating test specification to decrease personal dependability.

This paper introduces the best practice and framework of designing test cases that can be maintainable and possible to reuse for other project with less effort. This paper also gives an overview how the test cases shall be developed in order to help test engineers to reduce the efforts of redesigning test case in good manners.

This paper also introduces shortly the configuration management tool that support test engineer to easily extract and reorganize existing test cases.

In Chapter2, the general procedure and method of creating test case is explained in order to achieve good test case design for a target system. Chapter3 introduces best practice of designing test cases for automotive body electronics system with considerations of further extensions. In the final chapter, the supporting tool presents potential possibilities to utilize test cases in various purposes.

2. STANDARD PROCESS OF TEST CASE DESIGN

2.1 General Description

Reusability of test case relies on standardization of requirement specification. For instance, if the requirement specification is changed a lot in every single project, it won't be possible to reuse existing test cases for other projects. Therefore, the first step must be standardization of requirement specification as much as possible. So the requirement specification of a system needs to be well designed enabling to achieve the reuse of existing system.

But the standardization of test specification has some different points comparing with standardization of requirement specification. Various techniques should be applied depending on characteristics of a system considering reusability and scalability of developed test cases.

To increase the reusability of test cases that validates the requirement specification, the requirement specification needs to be decomposed first with the levels of reusable logical function units. And the logical function unit is encapsulated by configuration parameters that make the test cases to be reconfigured to meet the specific system variants such as regional sales variants and control signal interface changes to network signals from physical signals. Based on this, the generic test cases are hierarchically composed of using the reusable function units.

In Addition, there were also some points that many of existing test cases are quite different to each test designer even in same system. It raises a question of standard process of test case design method

To reach this goal, the standard procedure of test case design has been initiated to realize how test cases shall be designed in order to reach high reusability. This procedure that has been applied to design test case in automotive body electronics system is now introduced through providing quick overview.

In system development life-cycle, the development of test cases in test specification must be started right after a system requirement specification is released.

The standard process of test case design consists of three steps. See the figure1.

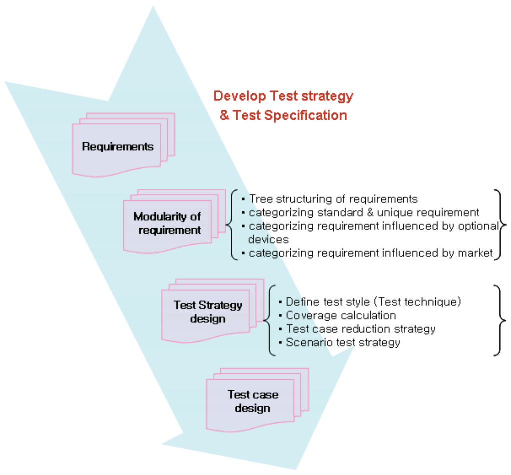

Develop Test strategy & Test Specification
Requirements
Modularity of requirement
• Tree structuring of requirements
• categorizing standard & unique requirement
• categorizing requirement influenced by optional devices
• categorizing requirement influenced by market
Test Strategy design
• Define test style (Test technique)
• Coverage calculation
• Test case reduction strategy
• Scenario test strategy
Test case design

Fig.1 Standard procedure for creating test specification

2.2 Standard process of good test case design

Modularity – The modularity of requirements is a kind of structuring a requirement into a tree to classify reusable function elements within the requirement. It may have similar approach like Classification Tree Method (CTE). The modular design of test case increases the reusability by extracting or integrating developed test cases. It is the concept to construct a three with selection of branches and leaves that are reusable. The modularity of requirement is divided into four steps.

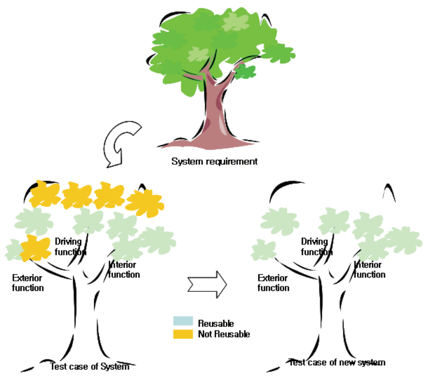


Fig. 2 Classification: Tree structuring to design test case

<u>Tree structuring of requirement:</u> it is simply to construct functional requirement tree from a system requirement. For instance, interior lamp requirement can derive 3 sub functions

- Interior Lamp Func
  - Foot lamp
  - Key Illumination
  - Room Lamp

<u>Categorizing functions</u>: The categorizing of function elements shall be executed according to type of requirement as well as the influence of optional devices and market dependency.

If a requirement varies depending on the market, optional devices, and projects, those test cases may not be possible to harmonize or may have high probability to be changed for the next project. Therefore, the classification of functions of design test cases must be conducted with consideration on the dependencies of requirements in a system.

Non-standard functional requirements imply high probabilities of changes and difficulties of reuse. Therefore, it shall consider separating the test cases from the standard test cases. It is usually possible to do in this way because the dependency of function comes from the optional devices or market demands based on the research.

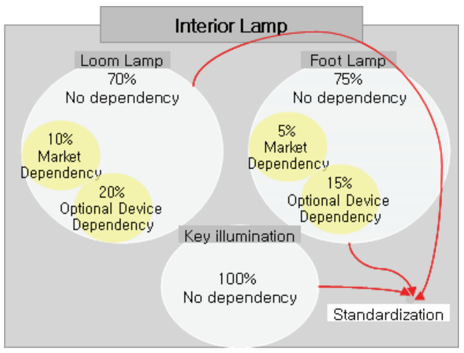


Fig. 3 Standardization: Categorizing standard functional elements from requirements

Test Strategy– Building the test strategy is to identify the intent of test and how to validate the test objects to achieve the coverage on a target requirement.

The definition of strategy is essentially important factor to develop reusable test cases for standard functions. The strategy can be very useful when the modification of test cases or failures of a system is found later. The great benefit of test strategy makes test engineers maintain and elevate the quality of test cases continuously.

Developing test strategy of standard function consists of three steps.

1st Step: Clearness of Test Intent and Method

2nd Step: Estimation of Quality level and Reduction Strategy

3rd Step: Building Test Matrix

1st Step: To clear test intents in order to make test engineer understand the purpose of the test. If test intents are clarified, proper test methods must be selected to validate and find problems out of the tests. To be more effective, the class of test methods shall be limited to avoid mistakes of test designer by selecting wrong or improper test method.

2nd Step: If test intent and test method are defined, it is important to measure the quality level. It can also be called as test coverage estimation. Depending on test method, the required number of test cases will vary. It is therefore very important to know how many test cases are mandatory to meet quality level or test coverage. These activities are very important in high risk functional requirement which may lead to significant effect in case of failure or malfunction.

If there are 6 different input aspects that are relevant to validate a function, the total test coverage can be a1*a2*a3*a*4*a5*a6. Each input (a1….a2) can have different number of state. We only consider the each input has only two states such as on or off condition as example. Then it requires total 64test cases (2^6) theoretically when the quality level needs to meet 100% condition and decision coverage.

We now consider deciding how many test cases must be developed and how many test cases can be removable. This activity cannot be performed automatically since it is difficult to distinguish valid or invalid test cases from the total required test case in any automatic tool up to now. We call the process as "Reduction Strategy." We adjust the balance of valid and invalid test cases to achieve desired quality level even by reducing test cases.

For instance, if a test target is to validate the output of logical combination "(A&&B)||(C&&D)", Total number of test case can be 16 test cases if all conditions are decisions are considered. A test designer is to define whether it is possible to reduce the test case by reducing the condition coverage with keeping required decision coverage. If condition coverage is less important in the test intent, Only 8 test cases can be developed because the each decision does not influence to each other to output behavior. All this description has to be document in the strategy for standard function test strategy in order to keep maintainability and reusability.

3rd Step: Building Test matrix is final activities of test strategy. Even though test intent and strategy of quality level is cleared, it is still difficult to write good test cases due to ambiguousness of test condition. The test matrix shall eliminate ambiguity about test objects. The matrix shall consist of three elements. Those are initial test condition and order of inputs and expected result. Many of test cases shows lack of information which make test engineer difficult to perform correctly. The test matrix is considered to overcome this problem systematically by establishing mandatory steps in test Matrix.

- Initial condition of Test object
- Order of Event in input values.
- Matrix of Input combination.

We found that many of ambiguous test cases were due to not enough definition of initial condition of test object. It often derives cryptic expression of expected result. Therefore, initial condition of test object must be well defined. Next step is to clarify the order of event of inputs. It means "Does output or requirement vary depending on the order of event of input values?". In the test matrix, it shall define whether order of input events must be taken into consideration or not. If it shall consider, then the number of test cases is increased by test designer to alter the order of inputs from the test matrix accordingly.

Finally, test designer shall create the test matrix, which contains the input states with expected result in each row. These activities have to be done by test designers but the writing test case is no longer necessary to be developed by test designers. Test cases can be written by test engineer or even anyone who has lack of knowledge on a system because the strategy prohibits possibility of making bad or wrong test case from the test matrix.

Test case –Outputs of each test strategy is expressed with matrix as explained, so test engineers can develop test case easily with matrix indicating how a test case shall be written. The test matrix complies enough with the definition IEEE Standard 610. "A set of Test inputs, execution, and expected results developed for a particular object, such as to exercise a particular program path or to verify compliance with a specific requirement."

3. BEST PRACTICE OF TEST CASE DESIGN FOR AUTOMOTIVE BODY ELECTRONICS SYSTEM

3.1 Goal to achieve

After the research of test cases used in automotive industries, the standard process of good test case design is setup as explained. In Chapter2, there are also been supplementary aspects considered in designing test cases for automotive body system.

Aspects:

Avoidance of interdependence of test case as possible

Intended design for validation via test automation

Intended design for verification of state model

Besides the reusability, it has been considered to use test case in multi-purposes. The question is how to fulfill all of these desires. Based on the research, there is a solution possible to achieve the desires.

Interdependence of test case is avoidable by classifying the test intent mainly in three conditions together with modular design of test case. According to the figure 4, we have established the type of test case to restrict unnecessary interdependence test case design. The basic reason of escaping interdependence is to attain reusability of plain design of test case from the complex function or system. The simplified test cases are later integrated into a test sequence to formulate complex test suite.

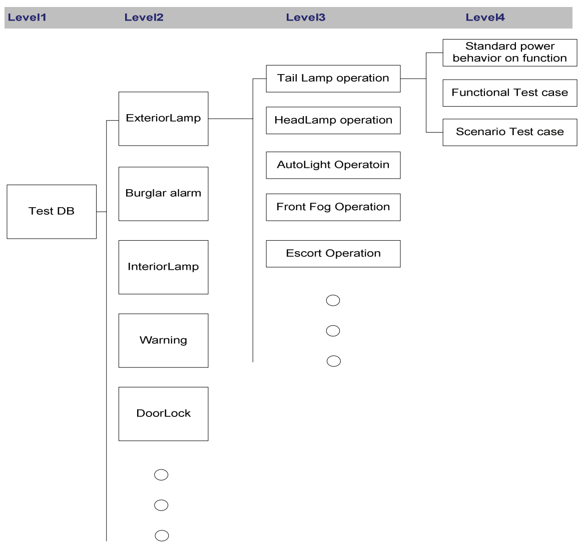
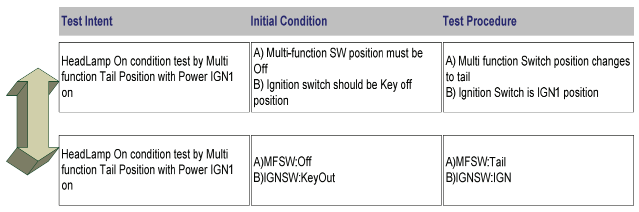


Fig. 4 Tree structure in Test database: Structure of test data base to avoid interdependence design of test case

For having possibilities of test automation and model validation purpose, test cases are designed by keyword instead of declarative manner. For example, Common test case used by declarative manner is replaced by simple keyword descriptions. It proves that the understanding of test case written in keyword description is not harmful for test engineers to perform test case. With this approach, it convinces that it could offer importing test case into automation tool directly from the test case database. It also gives an idea to generate synthetic signals that are useful to validate the model if a system is designed in simulation models.



Test Intent	Initial Condition	Test Procedure
HeadLamp On condition test by Multi function Tail Position with Power IGN1 on	A) Multi-function SW position must be Off B) Ignition switch should be Key off position	A) Multi function Switch position changes to tail B) Ignition Switch is IGN1 position
HeadLamp On condition test by Multi function Tail Position with Power IGN1 on	A)MFSW:Off B)IGNSW:KeyOut	A)MFSW:Tail B)IGNSW:IGN

Fig. 5 Test case: Test case described in keyword

3.1 Test case design for automotive body system.

Based on the developed process and supplementary aspects, test cases of automotive body system have been developed for a target vehicle.

- Analysis of requirement: Through the requirement analysis, functional tree and its branches have been built. This tree structure is the main stream of test case to increase reusability in other system as mentioned in the process.

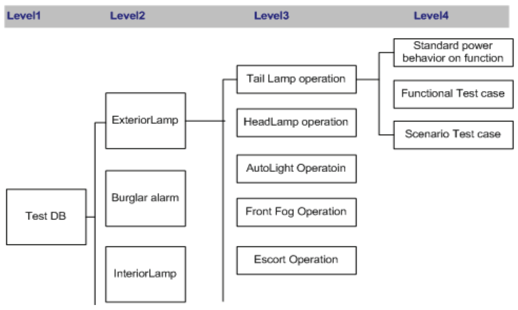
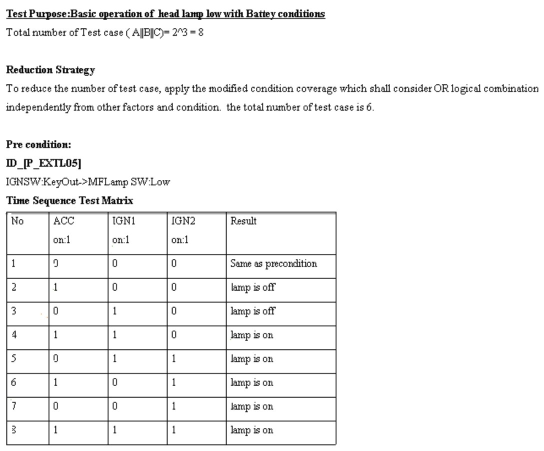


Fig. 6 Functional tree: Developing functional tree and branches from requirements

- Test Strategy: In case of Test strategy, there are two types of strategies considered practically. One is standard test strategy which shall use for standard power behaviour to lowest level of functional tree. The other is the strategy that a test designer must develop based on test intents. The selections of proper test method and coverage and reduction strategy are the key roles of a test designer.

To avoid deviation of each test designer, the formation of test strategy is also suggested.



Test Purpose:Basic operation of head lamp low with Battey conditions

Total number of Test case (A||B||C)= 2^3 = 8

Reduction Strategy

To reduce the number of test case, apply the modified condition coverage which shall consider OR logical combination independently from other factors and condition. the total number of test case is 6.

Pre condition:

ID_[P_EXTL05]

IGNSW:KeyOut->MFLamp SW:Low

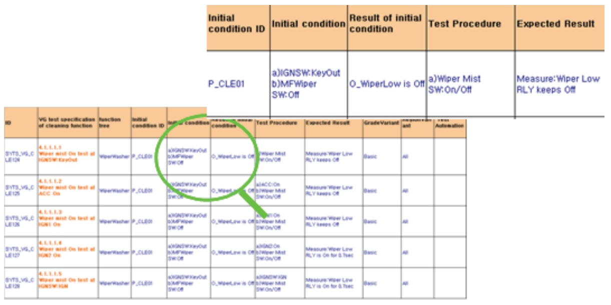
Time Sequence Test Matrix

No	ACC on:1	IGN1 on:1	IGN2 on:1	Result
1	0	0	0	Same as precondition
2	1	0	0	lamp is off
3	0	1	0	lamp is off
4	1	1	0	lamp is on
5	0	1	1	lamp is on
6	1	0	1	lamp is on
7	0	0	1	lamp is on
8	1	1	1	lamp is on

Fig. 7 Standard Test strategy: Elements of standard test strategy

For non-standard test strategy, it keeps same formation like standard test strategy in principle. This concept has been proven to be very efficient during reviews of test cases. We assure that it is no longer necessary to check each signal test case. Reviewing test strategies is good enough to find problems while the test case development.

- Writing test case: Now, writing test case is getting simple. The test case is derived from the test matrix designed in test strategy. Therefore, test case can be written by test engineers and other person who do not have good knowledge on a system. All keywords used in test case are stored in map file. This map file enables the test cases to be adapted to other projects if system architecture is changed. This map file can also be used to rename signals when test signals are generated for model validation.



Initial condition ID	Initial condition	Result of initial condition	Test Procedure	Expected Result
P_CLE01	a)IGNSW:KeyOut b)MFWiper SW:Off	O_WiperLow is Off	a)Wiper Mist SW:On/Off	Measure:Wiper Low RLY keeps Off

ID	VG test specification of cleaning function	function tree	Initial condition ID	Initial condition	Result of initial condition	Test Procedure	Expected Result	GradeVariant	[illegible]	Test Automation
SYTS_VG_CLE124	4.1.1.1.1 Wiper mist On test at IGNSW:KeyOut	WiperWasher	P_CLE01	a)IGNSW:KeyOut b)MFWiper SW:Off	O_WiperLow is Off	a)Wiper Mist SW:On/Off	Measure:Wiper Low RLY keeps Off	Basic	All	
SYTS_VG_CLE125	4.1.1.1.2 Wiper mist On test at ACC On	WiperWasher	P_CLE01	a)IGNSW:KeyOut b)MFWiper SW:Off	O_WiperLow is Off	a)ACC:On b)Wiper Mist SW:On/Off	Measure:Wiper Low RLY keeps Off	Basic	All	
SYTS_VG_CLE126	4.1.1.1.3 Wiper mist On test at IGN1 On	WiperWasher	P_CLE01	a)IGNSW:KeyOut b)MFWiper SW:Off	O_WiperLow is Off	[illegible] b)Wiper Mist SW:On/Off	Measure:Wiper Low RLY keeps Off	Basic	All	
SYTS_VG_CLE127	4.1.1.1.4 Wiper mist On test at IGN2 On	WiperWasher	P_CLE01	a)IGNSW:KeyOut b)MFWiper SW:Off	O_WiperLow is Off	a)IGN2:On b)Wiper Mist SW:On/Off	Measure:Wiper Low RLY is On for 0.7sec	Basic	All	
SYTS_VG_CLE128	4.1.1.1.5 Wiper mist On test at IGNSW:IGN	WiperWasher	P_CLE01	a)IGNSW:KeyOut b)MFWiper SW:Off	O_WiperLow is Off	a)IGNSW:IGN b)Wiper Mist SW:On/Off	Measure:Wiper Low RLY is On for 0.7sec	Basic	All	

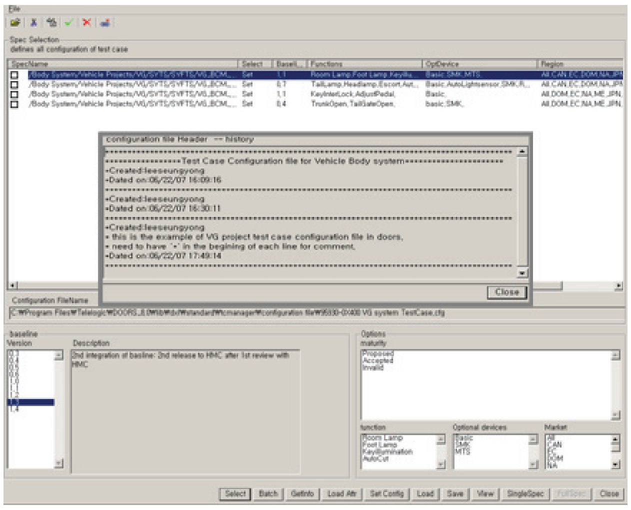
Fig. 8 Test cases: Exampled of developed test cases

3.3 Supporting tool

The test cases have been developed with database. Therefore, it becomes necessary to develop supporting tool to manage test cases to various purposes. We have focused on following targets primarily.

- Tool Targets
- Configuration management.
- Exporting data to Automation
- Generating signals for model verification.

- Configuration management: for a system, there are usually several variants depending on market and optional devices. We have found that there are some difficulties to have proper test specification quickly if new variant is introduced with a range of system. To have easy configuration management, a tool has been considered in order to extract and rearrange test cases by various configuration settings. It provides easy and quick integration of test specification to new variant and reuse test cases to other projects.



Spec Selection
defines all configuration of test case

SpecName	Select	Baseli...	Functions	OptDevice	Region
/Body System/Vehicle Projects/VG/SYTS/SYFTS/VG_BCM_...	Set	1,1	Room Lamp,Foot Lamp,KeyIllu...	Basic,SMK,MTS	All,CAN,EC,DOM,NA,JPN
/Body System/Vehicle Projects/VG/SYTS/SYFTS/VG_BCM_...	Set	0,7	TailLamp,Headlamp,Escort,Aut...	Basic,AutoLightsensor,SMK,R...	All,CAN,EC,DOM,NA,JPN
/Body System/Vehicle Projects/VG/SYTS/SYFTS/VG_BCM_...	Set	1,1	KeyInterLock,AdjustPedal,	Basic,	All,DOM,EC,NA,ME,JPN,
/Body System/Vehicle Projects/VG/SYTS/SYFTS/VG_BCM_...	Set	0,4	TrunkOpen,TailGateOpen,	basic,SMK,	All,DOM,EC,NA,ME,JPN,

configuration file Header --- history

```
*************************************************************************
******************Test Case Configuration file for Vehicle Body system**********************
*Created:leeseungyong
*Dated on:06/22/07 16:09:16
*************************************************************************
*Created:leeseungyong
*Dated on:06/22/07 16:30:11
*************************************************************************
*Created:leeseungyong
* this is the example of VG project test case configuration file in doors,
* need to have '*' in the begining of each line for comment.
*Dated on:06/22/07 17:49:14
*************************************************************************
```

Configuration FileName
C:₩Program Files₩Telelogic₩DOORS_8.0₩lib₩dxl₩standard₩tcmanager₩configuration file₩9530-0X400 VG system TestCase.cfg

baseline
Version: 0.3, 0.4, 0.5, 0.6, 1.0, 1.1, 1.2, 1.3, 1.4
Description: 2nd integration of basline: 2nd release to HMC after 1st review with HMC

Options
maturity: Proposed, Accepted, Invalid
function: Room Lamp, Foot Lamp, KeyIllumination, AutoCut
Optional devices: Basic, SMK, MTS
Market: All, CAN, EC, DOM, NA

Fig. 9 supporting tool: configuration management of test case in test database

The tool is able to have configuration file to each variant and data and configuration file can be reloaded and modified to extract required test cases from database. Several type of test specification template can be used for documentation. The configuration file itself can also be traceable with version management.

The tool can have various configurations with combination of levels since the test cases have been designed in 4 levels inside classification tree

- Exporting test case for Automation: one of the goals is to use the designed test case to other purposes. We mainly focus on the usage of Test automation and state model verification. Those activities are currently required huge effort to construct automatic test scripts based on the test cases. It generally invokes many failures of implementation due to human errors. Additional review process therefore becomes mandatory. One of the best ways is to provide main frame of test case by simple import of test case into automation system. The main frame containing useful information such as test intent, initial conditions and test procedure as well as expected result is loaded to automatically. It helps test engineer eliminating his or her errors while implementation. We also consider using test cases for model validation. To validate the model designed from off-the-shelf, there are various methods that can be used. But we found that the effective way of model validation is to use test cases developed by test engineers who have talent to detect failures on a system. The test cases from them are mostly well balanced to detect failures as well as performance.

Test case described in keyword is translated in the interpreter of configuration tool and each inputs as well as parameters defined in the initial test conditions are converted to signals through the signal builder map file. The generated signals are then imported to simulate the model.

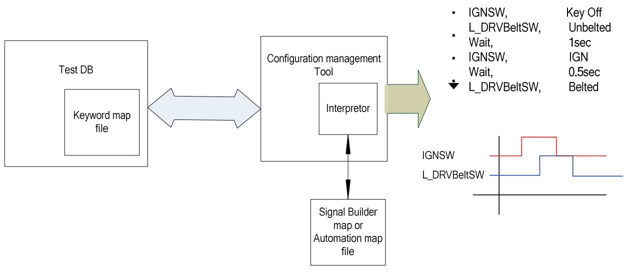


Fig. 10 supporting tool: Generation of exporting data for automation and model validation

4. CONCLUSIONS

In conclusion, the outcome of developing test case based on the rule suggested has shown several benefits but it has also drawn further considerations that may be dealt as weak points for wide range application of automotive industries.

The highlights of the outcome are high reusability of test case by avoiding interdependency test case, function tree structure and design consistence of test case to assist test engineers in maintaining and modifying test cases in same manners. It mostly makes test cases possible keep up formulating good test cases for any person in testing groups. There are also good points to apply test cases to automation. It is quite easy and less problematic even by manual insertion because test cases are have been described by keywords instead of declarative sentence.

The designed test case is also possible to be powered by supporting tool which enable test engineer to extract only required test cases from test database and to document it by the tool. Additional features presented in previous chapter are the benefits of test case application in broad ranges.

In contrast to benefits, there were suggestions in the paper to recommend restricting methods to design test strategy to minimize potential deviation of test strategy by each test designer. The limitation of test method such as stress, state-model based classification tree based, risk based, could deteriorate good design of test case. Nevertheless, we recommend you to limit the test methods in your system with early research on finding appropriate test methods or by replacing old test method by new method from the lesson learn in order to maximize reliability of your system in field with the best efficiency.

REFERENCES

John D.McGregor., David A.Sykes.,(1992). *Object-Oriented Software Development: Engineering software for reuse,* International Thompson.

M.Elizabeth C.Hull., Ken Jackson., A.Jeremy J.Dick.,(2002). *Requirements engineering,* Springer-Verlag London, London.

Cem Kaner,(2003), *"What is a good test case?",* Florida institute of technology.

Rex Black.,(1999), *Managing test process,* Microsoft press.

Claudia Dencker.,(2003), *Common Mistakes in test cases in* pacific northwest Software Quality Conference*,* Software SETT corporation.

Musa John.,(19988), *Managing test process,* Microsoft press.

Scott Loveland.,Geoffrey Miller.,Richard Prewitt Jr., Michael Shannon.,(2004), *Software Testing Techniques: Finding the defects that Matter,* Charles River Media.