An Approach for Partially Automated Test Generation Based on Signal Recordings

Timur Eksen[1], Frank Thielecke[2]

Abstract: The increasing complexity of aircraft systems and the number of functions running on avionics components leads to a rapid growth of the number of required testing. In addition, the mission profile can change over the long life cycle of the aircraft, so that the originally developed tests may not cover all scenarios that occur during the operational use. Therefore, new test strategies are needed which allows tests to be generated without the effort growing along with it. At the same time, the aim is to identify and analyse previously unknown behavioural abnormalities.

This paper describes an approach for the semi-automated generation of new test scripts based on recorded data from operations or test and simulation activities. For this purpose, methods were developed to convert these data into a generic format, so that they can be managed and processed with uniform functions. The recordings can be filtered via a graphical user interface (GUI) to derive test stimuli and verdicts. For the generation of the test script an automated process was developed, which is based on the specification for the test description language Generic State Chart extensible Markup Language.

Keywords: system testing, test generation, data recordings, automation

1 Introduction

Aircraft and their systems are exposed to many different tasks during their life cycle. Even if the requirements are defined and implemented as precisely as possible during the development phase, it is still possible that the end user will use the aircraft for missions and purposes other than those intended. A passenger aircraft becomes a cargo aircraft and an aircraft originally planned as medium-range aircraft is further developed for long-haul flights. Due to the changed use, there may be unknown effects in the operational behaviour that were not considered during development, which can date back several decades, and were therefore not tested.

[1] Hamburg University of Technology (TUHH), Institute of Aircraft Systems Engineering, Nesspriel 5, Hamburg, 21129, timur.eksen@tuhh.de

[2] Hamburg University of Technology (TUHH), Institute of Aircraft Systems Engineering, Nesspriel 5, Hamburg, 21129, frank.thielecke@tuhh.de

The challenge now is to recognise these effects and to feed them back into the verification process as effectively as possible, so that it can be ensured that the systems are suitable and safe to use with the new scenarios. These scenarios can be identified in various ways. The adapted customer profiles could be specified directly by the operator and shared with the original equipment manufacturer (OEM), whereby the operator would first have to know the original scenarios. On the other hand, the developer could of course also accompany the operation of the aircraft. Both variants require additionally trained employees who would have to take over the feedback of these effects into the test process. An exact reconstruction therefore requires a high expenditure of personnel and time, which in turn is increasing the cost factor that should be avoided.

In addition to the changing operational scenarios, the systems are becoming increasingly complex and, made possible by the continuously increasing computational power, new functions are constantly being implemented on the avionic systems. Alongside the pure increase in complexity of individual systems, there is also an expanding networking of former self-sufficient components, whereby many interactions can occur. In order for the test process to withstand these developments, the degree of automation must be increased so that lengthy manual operations can be eliminated.

This paper presents a method for deriving tests from data recordings. This should reflect the increasing digitalisation of systems and the growing standardisation of data recording in flight operations. At the beginning, the relevant basics for the following topics are described, followed by the methodological procedure and the implementation of this approach. The methodological procedure is further subdivided into the areas of usable data sources and their management, the derivation of conditions for verdicts based on recordings using a graphical interface and the process for converting the data into tests. Then the application of the method is described by a virtual test and the results of the implementation are discussed. Finally, the contents of this paper will be summarised and an outlook on the following work will be given.

2 Fundamentals

For a better understanding of this paper, some fundamentals are initially introduced. First, the test strategy at the Institute of Aircraft Systems Engineering (FST) is briefly described and another publication on this topic is referenced. Then the test description language is introduced into which the signal recordings will be transferred at the end of the process. Finally, related work to this paper is presented.

2.1 Test Strategy

Due to the increasing complexity of systems and functions, test methods also need to be further developed so that the number of time and involved personnel does not become a blocking element in the development process. At the Institute of Aircraft Systems

Engineering (FST) of Hamburg University of Technology (TUHH), two different approaches are pursued for this purpose, a model-based and a data-driven approach, each of which addresses the problem of test creation and test coverage from a different perspective. Since the research on the two approaches is still ongoing, this paper does not compare them or give a final evaluation. This will be done at a later stage, so only a brief overview is given here.

In [HT20] a model-based method for creating test stimuli for operational scenarios was presented. In this approach, a modular scenario model as state chart is build up gradually, from which signals can be generated as test input. The modular approach allows elaborating individual environmental objects in detail systematically without directly editing the overall model, thus reducing the complex environment definition to simpler tasks. The basic system model is kept very abstract and is only specified during the later process. This allows an early implementation of an initial model and early testing. This forward-looking approach aims to identify undesired but realistic scenarios in the development process so that they can be avoided prior to the entry into service.

In contrast, the data driven approach takes a retrospective view of scenarios. Although the method allows the usage of recordings from simulation and test runs, the focus is on data from the operational handling of aircraft systems. It is assumed that, due to the high number of flights and the diversity of aircraft operators, a broad mission spectrum can be acquired, under the condition that the required data can be collected during the operation. Since the scenarios cover all performed real-world application scenarios if all flights are actually recorded, the data basis can be used to achieve very broad test coverage. The challenge of this approach, is to extract from the huge number of data those data sets which contain relevant not yet tested behaviour and which do not only represent the nominal case.

2.2 Generic State Chart extensible Markup Language

In the testing area, specialized description languages for test cases have been developed, which support many test procedures, but this heterogeneity also prevents easy exchange between project partners and the adoption of existing procedures for other systems. In [Bo18], State Chart extensible Markup Language (SCXML) was selected as the description language for the exchange format based on various application scenarios from the field of aircraft testing.

SCXML is an open source standard that was developed by the World Wide Web Consortium (W3C) [SC15]. It is an event-based machine language for the representation of state diagrams based on Harel state charts. The standard contains among other things constructs for the description of parallel and nested state charts, which gives many possibilities for test description.

In the research project, Agile-VT the existing standard was further developed with the aim to identify missing constructs needed for the test process and to specify them. This was done taking into account the intersection of functions given by the participating test system manufacturer dSPACE, TechSAT and Vector. In [Fr19] the requirements for an intermediate format for the exchange between industrial partners and the required structure are considered.

2.3 Related Work

The topic of using data recordings for the verification and validation process (V&V) has been the subject of several papers in recent years. The focus of these works was not always on the test process, but in some cases also in comparing existing results with recordings from real world application scenarios. The topic was pursued particularly strongly in the automotive area of Advanced Driver Assistant Systems (ADAS). Even though the direct area of application is different, some of the work will be presented due to the similar objectives.

1) Using Data Recordings for the V&V Process:

The work in [La18] describes a method for recognising new scenarios in relation to the existing test set. For the creation of a reduced test set of input vectors, an autoencoder is used which is adapted to the system under test by an initial data set. Through further test runs, additional recordings could be generated, which were separated into sequences of predetermined length and then added to the autoencoder as further training data sets. The distance to the original test set is used to determine which sequences are added to the test set. To avoid repetition, only sequences with the greatest distance are added so that the range of scenarios that can be tested is as wide as possible. Another approach to the applicability of real world recordings in the field of Rapid Control Prototyping (RPC) is presented in [Ba15]. The focus of the work is on the handling of components that not only require a time-dependant input, but also have to react to certain events (e.g. recognition of road signs). For this purpose, an analysis of the dependencies of the input signals is carried out so that they can be grouped into different signal types. In order to create a coherent overall stimulation, these are then coupled to the movement of the vehicle and used in X-in-the-loop simulations.

A slightly different approach is presented in [Ro16], as the real world data is not used to stimulate a test system, but as a benchmark for autonomous driving functions in road traffic. Since these should be at least as safe as the average human driving behaviour, this behaviour must be processed as a reference.

Compared to the aforementioned works, this paper aims to describe a more generic method to make data from different sources (test, simulation and operational use) reusable in the V&V process. While taking into account the experience gained during testing with specific systems, it will still allow easy use with different systems and system levels. This is also supported by the use of generic SCXML, so that there is no direct dependence on just one test system.

2) Test Automation:

The challenges and opportunities of automated testing were highlighted in [BWK05] using several projects as example. Although the projects do not come from the field of aircraft systems engineering, but from various software projects (e.g. sales support for insurance companies), many fundamental points can still provide important insights. In the projects, the tests were carried out more effectively when it was possible to carry out tests in different system levels (e.g. unit, integration) and not just limit oneself to certain areas or applications. This allowed test procedures to be integrated earlier into the development and the parallel progress meant that the test cases could be better adapted to the actual needs. Furthermore, it became clear that the processes during automated testing must be captured and documented as completely as possible, as it can happen that components and systems have to be tested again after a new development step.

The work described shows that increased automation of all areas of test execution (test creation, configuration, execution and evaluation) can lead to a more error free System Under Test (SUT). However, the best results were achieved when a mixture of manual and automated processes was used, so that the test process is not executed in a black box with no oversight from the test engineer.

3 Methodical Approach for the Test Generation

The developed method for test generation is based on four essential steps, so that data recordings from different sources can be used for the testing process:

1. Pre-Processing of the recordings into a generic structure, consolidation of meta data and creation of database entry
2. Selection of the signals to be used to stimulate the System Under Test
3. Selection of the signals to be used as reference for the verdicts
4. Automatic generation of the test script

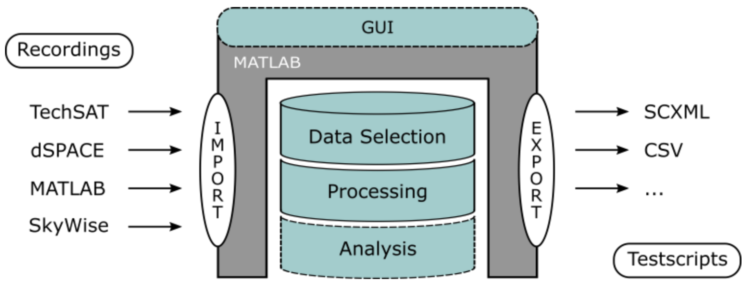


Figure 1: Schematic Structure of the Process

The second step is currently still a manual activity and requires mainly system knowledge and good knowledge of the existing recordings. As this is a rather trivial task apart from the acquisition of the relevant information, it is not described in detail in this paper. In the current implementation, the signals can be filtered using structured query language (SQL) functions and then added to the signal site for test generation (3.3) via a manual confirmation.

3.1 Data Processing and Management

In order to work efficiently with the recordings and to allow various functions to access the data, a relational SQL database was set up which serves as the basis for all further processes. Import functions make it possible to use different data sources as a data basis. These convert the recordings from their original format into a generic structure. The numerical values are stored in a matrix and the signal names in an additional cell array. This has the advantage that during further processing there is a clear assignment of the data types to the containers and further conversion steps are omitted. The connection between signal names and numerical data is ensured by the same position in the matrix and the cell array. All other implemented functions can thus have a uniform structure and can act independently of the data sources. So far, import functions for recording formats of the test systems from dSPACE and TechSAT, exported data from the Airbus database SkyWise and data generated in MATLAB/Simulink have been created. By importing recordings from these data sources, three different directions are covered. Data from test and operational runs as well as from simulations can be processed. The recording systems typically use a proprietary data format for the direct storage of the recordings that cannot be read by MATLAB. In order for the import function to read the data, it must be converted into a non-proprietary format in an intermediate step by the use of proprietary tools. With the sources mentioned above, export as a comma-separated-values (CSV) file has proven to be a good solution, as it can be exported by all systems used so far. The structure of the information in the file still shows significant differences but it can be easily processed by fitting adapter functions.

The read-in data is not stored directly in the database, but in one MATLAB workspace file per data recording. This has the advantage that no further conversion between the internal data formats of SQL and MATLAB is required when the recordings are loaded during the test generation process. By using the workspace files, MATLAB is able to access individual vectors in the file directly, which results in a great increase in performance, as only a small part of the data has to be loaded into the working memory.

During the import process, metadata (e.g. minimum, maximum and mean value) is automatically collected, making it possible to compare the recordings. In contrast to the recordings, this metadata is stored directly in the database, as its data volumes are much smaller than those of the recordings and do not significantly affect the processing performance. Furthermore, the SQL database search and filter functions can be used for the metadata. On the one hand, manual searches for specific data samples can be carried out quickly and on the other hand, the search and filter functions can also be easily automated. The search and filter criteria can be easily combined with each other in SQL, so that from a large selection of data recordings, only those that are needed for the test stimulus or the verdicts remain as the result. For example, all data recordings that contain a certain signal, which exceeds a predefined threshold, could be selected. Afterwards, the result of the filtering could be sorted by date, so that the most recent entries are displayed first. The metadata contains the storage paths to the individual workspaces and the exact position of the stored signals, so that the actual recording can be accessed via this information. The database thus acts as managing interface between the data and all the functionalities implemented in MATLAB. The schematic overview of the developed structure is shown in Figure 1.

3.2 Graphical Selection of the System States to be checked

The main task of a test is to verify the correct behaviour of the SUT. In addition to stimulating the system, criteria are needed to determine whether the system behaves as expected. These criteria can be often taken from requirements, for example redundancy conditions.

In the data based approach presented here, recorded signals are used as the basis for the verdicts. These signal courses contain the knowledge about normal or abnormal behaviour of the used system. The recorded signals can be those from previous tests or simulations. Therefore, verdicts can be generated if the SUT is used for the first time and no further information exists. The signals in the database are not categorised as input or output signals, so that no additional restriction is imposed for the selection process. The classification as input or output signal is currently only recognisable by the signal name, if a predefined naming scheme was used during recording. For a meaningful signal selection, expertise on the system is needed so that the test condition can generate further insight into the system behaviour.

A pre-selection must be made from the database, from all available signals, so that the handling is simplified. Then the number of verdicts needed must be determined and which signals from the pre-selection are intended for which verdict. The assigned signals can then be plotted for each verdict. In the plot, the timestamps from which the parameters for the verdict are to be derived can be selected directly in the GUI. For this purpose, a vector is formed from the values of all selected signals recorded at that time. Figure 2 shows possible signal courses and an exemplary selection of state vectors for the verdicts. The broken lines (v1-v6) mark the selected timestamps at which the values of the signals for the verdict generation are determined. The signals could be, for example, the status (active/passive) of loads in the cabin. The graphic representation allows good overview of the signal course and a precise and easy selection of the state vectors. Finally, these can be adapted to the needs of the test. If several signals are used for a verdict, they can be linked together by logical conditions (Table 1). In addition, an upper and lower tolerance can be defined for each value individually and independently of each other. As not all required signal states may not always exist in the available recordings, it is also possible to parametrise additional verdicts purely manually. When the verdict is selected, entries for them are created in the GUI with the corresponding parameters. These can also be changed or they can be created without parameters to produce them completely manually.

During the test execution, the verdict signals are checked at the selected timestamps. Depending on the selection of the parameters, these must lie within a tolerance range or correspond exactly to the specified value. From the individual results, whether the signal corresponds to the expectations or not, the overall result for the respective verdict is obtained by using the logical connections. A check of the signal curve over a certain period is currently only possible with the use of several verdicts arranged consecutively.

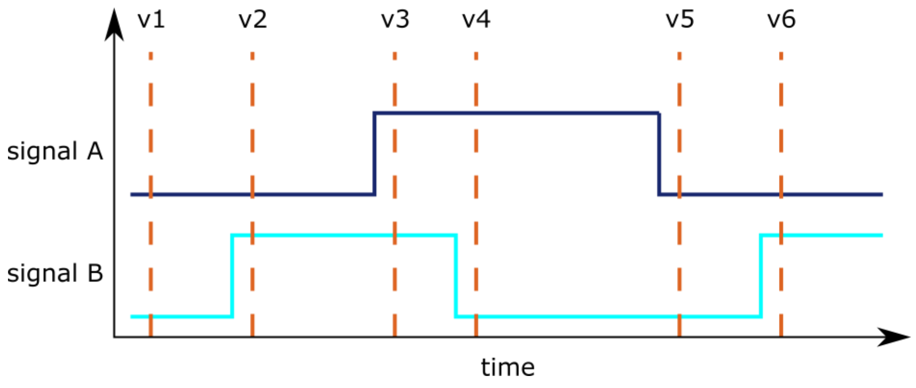
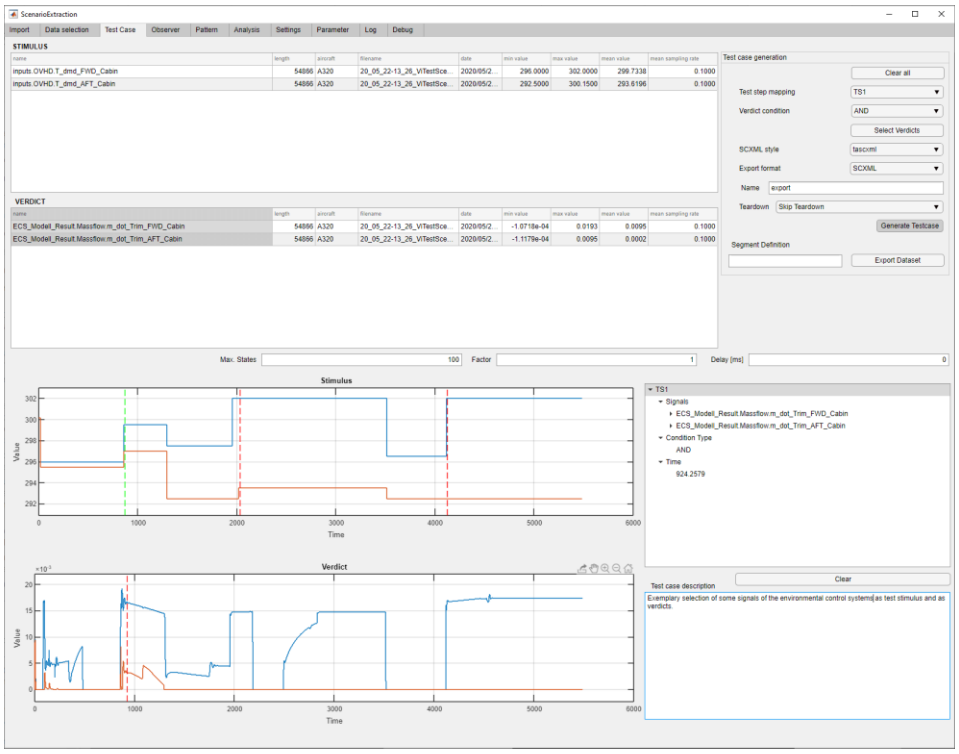


Figure 2: Exemplary Selection of Signal States for Verdict Generation

Logic	**Description**
AND	All signals must be within the defined tolerance range.
NAND (Not AND)	At least one of the signals must be outside the defined tolerance range.
OR	One of these signals must be within the defined tolerance range.
NOR (Not OR)	All signals must be outside the defined tolerance range.
XOR (Exclusive OR)	If two signals are selected, one must be within the defined tolerance range and the other must be outside.

Table 1: Logical Conditions for Linking Verdict Conditions

Figure 3 shows a screenshot of the GUI for selecting the test stimuli and the verdicts. At the top left, all signals that have been preselected as stimulus or verdict reference are listed in tabular form. The plots at the bottom left represent these graphically, whereby the upper plot contains all signals for the stimuli and the lower plot contains the signals for the verdicts. For the verdicts, only the signals are shown that were assigned for the specific test step. In the overview, to the right of the plots, the desired test step can be selected and values taken over from the graphic view can be edited manually there.



STIMULUS

name	length	aircraft	filename	date	min value	max value	mean value	mean sampling rate
inputs.OVHD.T_dmd_FWD_Cabin	54866	A320	20_05_22-13_26_ViTestSce...	2020/05/2...	296.0000	302.0000	299.7338	0.1000
inputs.OVHD.T_dmd_AFT_Cabin	54866	A320	20_05_22-13_26_ViTestSce...	2020/05/2...	292.5000	300.1500	293.6196	0.1000

VERDICT

name	length	aircraft	filename	date	min value	max value	mean value	mean sampling rate
ECS_Modell_Result.Massflow.m_dot_Trim_FWD_Cabin	54866	A320	20_05_22-13_26_ViTestSce...	2020/05/2...	-1.0718e-04	0.0193	0.0095	0.1000
ECS_Modell_Result.Massflow.m_dot_Trim_AFT_Cabin	54866	A320	20_05_22-13_26_ViTestSce...	2020/05/2...	-1.1179e-04	0.0095	0.0002	0.1000

Test case generation

Test step mapping: TS1
Verdict condition: AND
SCXML style: tascxml
Export format: SCXML
Name: export
Teardown: Skip Teardown
Segment Definition:

Max. States: 100
Factor: 1
Delay [ms]: 0

TS1
- Signals
  - ECS_Modell_Result.Massflow.m_dot_Trim_FWD_Cabin
  - ECS_Modell_Result.Massflow.m_dot_Trim_AFT_Cabin
- Condition Type
  - AND
- Time
  - 924.2579

Test case description: Exemplary selection of some signals of the environmental control systems as test stimulus and as verdicts.

Figure 3: GUI for selecting the test stimuli and the verdicts

3.3 Automated Generation of Test Scripts

A test script can be automatically generated from the selected stimulation signals and verdict conditions. In an intermediate step, the data is structured in a way that fits the test description language specification. Since the specific structure of the selected test description language is only taken into account during the export, it is possible to implement export functions for other languages without having to modify other sections of the method. In this case, the export was implemented for generic SCXML, which was further developed in the project.

To enable the stimulus signals to be converted with the commands available in generic SCXML they are classified first. The goal is to select whether the signals are mapped identically to the original in SCXML or whether an approximation is useful. For this purpose for each signal, it is determined how often the signal value changes in comparison to the previous data point (e.g. two value changes for a rectangular function). If the rate of change is low in relation to the length of the signal vector, it can be easily mapped using set commands. For signal sequences with many value changes (e.g. sine waves), this method leads to extremely long scripts due to the underlying SCXML syntax. The size of the script also increases the effort to translate and compile the script for the different test systems and at the same time reduces the auditability of the test script. The more severe this becomes, the more difficult it becomes for the test engineer to detect and check deviations during the test execution. To eliminate this problem, a possibility for approximation of the test stimulation has been integrated.

One objective in selecting the approximation method was that the method should be able to use commands from the generic SCXML specification. This was intended to avoid that the representation of the approximation in the SCXML script increases the size. The implemented algorithm is a piecewise linear approximation (PLA) based on the sliding window method [Ke01]. Starting from the first data point, a linear substitute segment is set up, which first goes to the following data point. For this segment, the local error is determined and if it is smaller than the maximum specified error, the segment is extended gradually. This is done until the threshold is exceeded, which is also the beginning of a new segment .The root-mean-square error (RMSE) is used to calculate the difference between the original signal and the approximated segment. These segments are parametrised in the test script as generic SCXML ramp function. For the test generation, the algorithm was implemented in such a way that the tester can influence it significantly by three parameters (Table 2). If one parameter was selected as the main criterion, the other two parameters are variable, so that the approximation can achieve the main goal. For example, if a maximum number of states is given, then an approximation is created starting with an initial value for the deviations. After the first run, the local error is adjusted by a certain percentage, depending on whether the maximum value was exceeded or not. If the maximum number of states has been exceeded, than a larger deviation is allowed, so that fewer ramps are used. If the maximum value is undershot, in contrast, the local error is reduced until either the maximum value would be reached on the next pass or if there are no more significant changes in the number of states between two passes.

Parameter	**Description**
State Limit	Maximum number of allowed states the test may have. Corresponds to the maximum number of ramps or segments into which the stimulation signal is broken down.
Local Error	Maximum deviation between the ramp as approximation and the original segment.
Global Error	Maximum deviation between the complete approximations obtained from all ramps compared to the original signal.

Table 2: Parameters of the Approximation Algorithm

Either after the representation has been determined for all signals as a set or as a ramp command and the corresponding parameters have been determined, a consolidation of the required states takes place. Since the commands for each signal were determined individually, it can happen that several command calls have to take place simultaneously. These are bundled in a common state so that no additional parallel path is created (Figure 4). This simplifies the implementation of the generation process on the one hand, and on the other hand, the test procedure is easier for the user to understand. This is especially important for automatically generated test scripts to increase the acceptance of it.

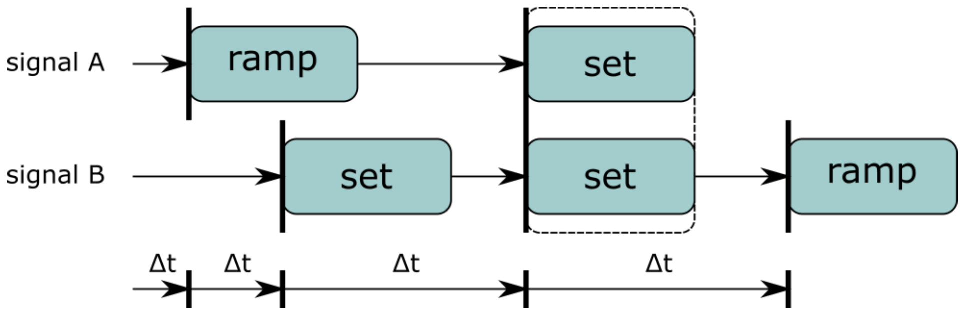


Figure 4: Summary of Stimulation Commands into Common States

The template according to which the test script is generated is shown in Figure 5. The states startup and teardown are used to set the system to the appropriate initial state or to reset it after the test is finished. With the data based approach, the first state vector is taken from the recording. It is assumed that the recording does not start directly with the abnormal behaviour, but that the nominal case is still present at the beginning. Since this is not always true, the activation of these two states is optional. The actual stimulation of the test system takes place in the compound state, which is called logic. Based on the conversion of the selected signals, as one-to-one conversion or as an approximation, the required commands are converted into SCXML states. The timing is implemented by the use of delayed events, so that the transition to the next state is only achieved after the delay has elapsed. The verdicts are integrated into the timing of the test script equivalent to the stimulus commands.

Table 3 shows an exemplary process description of three signals that are to be converted into an SCXML script. The signal values at time zero are used by default as the initial values of the test run and are set in the startup state. The following signal changes are written to the logic state of the test script. In this example with three points in time, the states ‘logic_c0_c0’, ‘logic_c0_c1’ and ‘logic_c0_c2’ are created for this purpose, which are named according to the naming scheme defined in the Agile-VT project. Within these states, the commands (set and ramp) for the individual signals are integrated. After 0.5s, signal A is set to the value 10 and a wait command delays the transition to the next state. At time 1s, signal B is stimulated by a ramp function, whereby the values in the vector specify the slope, the start value and the execution time of the ramp. In this case, the ramp starts with the value five and then runs for 10s with a slope of one until the value 15 is reached. The last table entry shows the case where two signals are changes at the same time.

Time [s]	**Signal**	**Value**	**Command**
0	Signal A	0	Set
	Signal B	5	Set
	Signal C	-2	Set
0.5	Signal A	10	Set
1	Signal B	[1, 5, 10]	Ramp
3	Signal A	0	Set
	Signal C	[-2,-2,15]	Ramp

Table 3: Exemplary Process Description for the Test Case Generation

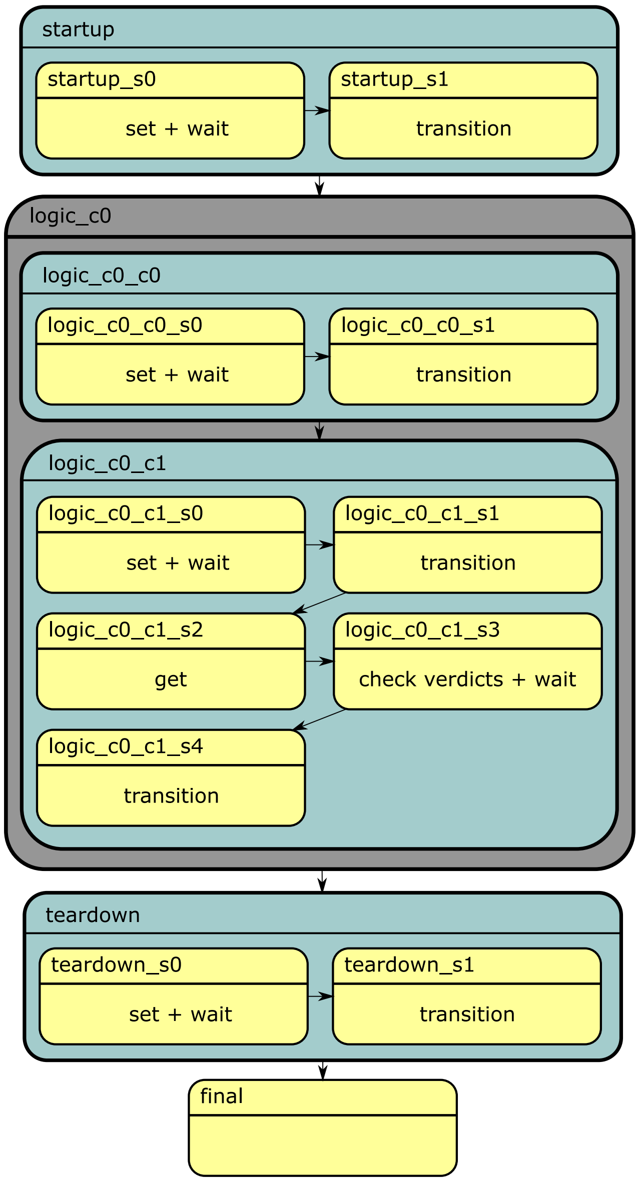


Figure 5: Template for Generation of Test Scripts in Generic SCXML

4 Verification of the Method

The developed approach for test generation based on recordings was tested on a virtual oxygen test system of the A350. This system offered the possibility to represent real processes in real time and to include manual interactions during the runtime. This makes it easy to create different procedures to test the method, and the manual interface creates opportunities to quickly change the system state to see if the conditions detect system malfunction.

For the verification of the method, a logical operational sequence was created together with the project partner Airbus as a use case in a simulation environment. The simulation execution and the recording of the data was performed using tools of the test system manufacturer TechSAT. During the recording, the sequence of actions was performed manually via a control GUI and consisted of the following steps:

1. Activation of the oxygen supply to the captain
2. Activation of the oxygen supply for the first officer and the third crew member
3. After a phase of normal oxygen consumption, the pressure in the active cylinder is reduced to such an extent that the switchover to the reserve cylinder is forced

Using the presented approach, signals from the recording were selected for stimulation so that the test steps described above, originally executed manually, could be generated by the test script. From other recorded signals the verdicts were derived, by which the success of the executed sequence could be determined. The generated generic SCXML script was then executed on the test system. The correct execution was verified by several measures. During the running test, the functional flow was monitored by the verdicts and the visible behaviour in the GUI. In addition, a new recording of the test was made, which was compared to the original recording and which allowed the correct timing to be checked.

Based on this use case it could be shown that the presented method for databased test generation is successfully applicable. The partly automated derivation based on existing recordings has significantly accelerated the generation process and it is possible to execute for users without knowledge of the test description language. Therefore, existing sequences from previous activities (test, simulation or operational activities) can be easily returned to the test operation.

During the development and testing of the method, limitations became apparent that must be taken into account when selecting recordings and signals. Depending on the structure and configuration of the test bench, not all signals that have been recorded are suitable for stimulating the system. When selecting the signals, care must be taken to ensure that only input signals from the associated systems are used to avoid illogical stimulation. This can happen if the signals have a name throughout the simulation and it is not possible to see at which point of the system or subsystem the signal value was recorded. This problem can be avoided by a clearly defined naming convention, as it is always traceable to which system the signals belongs and whether it is an input or output signal. Furthermore, it may be that the signals are set actively from other simulations. Although the test script stimulates the signals, they are immediately overwritten so that the desired behaviour cannot be achieved. When selecting the signals, it is important to make sure that these signals are not selected for stimulation. This can be done by addition of metadata about the signals or existing system knowledge of the engineer.

Furthermore, it has been shown that depending on the executing system and the settings, some system processes are sampled periodically. Among other things, this can lead to the fact that the activation of a button may not be recognized in the GUI. This GUI can be used to activate errors or change parameters and system states of the oxygen system. The actuation sequence (0>1>0) sets the value to one for only a few milliseconds, which corresponds to the real recorded process, where the actuation coincides with the cyclical query of the value in the GUI. To ensure the correct execution, either the GUI interaction must be longer than the cycle or the actuation sequence must be automatically extended during test generation. For both variants, specific system knowledge is required to identify the problem and to judge that the adjustments do not distort the desired test procedure.

5 Summary and Conclusion

This paper presented an approach for the semi-automated generation of test scripts based on recordings of previous simulations, tests or operational activities. The presented method has shown with which structure of data processing and management it is possible to take different data sources as basis for the test generation process. A software framework based on MATLAB and a SQL database was developed, which allows executing generic functions without direct dependencies to the original storage format. Afterwards the graphical approach for the selection of verdict conditions, the implementation as MATLAB GUI and the settable parameters were presented.

Furthermore, the automated generation process was explained, which can convert recordings into test scripts based on the selected stimulus signals. The challenge was made clear that depending on the test description language, further preparation of the data could be useful. For this purpose, a categorisation of the signals was carried out, whereby the highly fluctuating signals are converted into stimulation commands via an approximation.

The automated generation process has demonstrated the fast adaption of new signal recordings for test procedures, as the time consuming manual formulation of the sequences is no longer necessary. This process also ensured that all test scripts are generated without errors and that they are conform to the specification of the test description language. As long as test environment (e.g. test bench, configuration) and test script fit together, the operability can be assumed with a high degree of certainty.

It has also been shown that some knowledge about the involved systems is still required to be able to carry out the test and achieve meaningful results. On the one hand, this is to the correct selection of stimulation signals so that they are compatible with the test system, and on the other hand, stimuli must be selected that produces a relevant scenario.

6 Outlook

The presented approach has presented the basic procedure for deriving test cases based on data recordings, but many of the described procedures are still based on manual activities of the test engineer. These steps are to be further developed in future activities so that they can be automated or that the activities of the test engineer can be supported by providing further information. Especially the selection of the data sets to be used for test generation currently requires a precise knowledge of the recording processes in order to generate a meaningful stimulation.

Since operational data is also used for this process, it can be assumed that the required knowledge will not always be available due to the large number of different recordings. Therefore, it is essential to integrate analysis algorithms into the selection process, which will create a pre-selection of the available data sets. Assuming that the nominal case has already been tested extensively during development, it will then be the task of the analysis procedures to identify the recordings in which abnormal behaviour has occurred and to test it more precisely afterwards. For this purpose, different pattern recognition algorithms and approaches will be investigated by which nominal and abnormal behaviour patterns can be marked in the signal data. Afterwards, it is planned that all newly imported recordings will be automatically examined based on the known patterns and that conspicuous data sets can be thus highlighted for the test generation.

During the development, so far it became clear that test engineers only accept automated process steps if the process is traceable and if it is possible to intervene in it if necessary. Therefore, it is planned to automatically create a documentation of the process for each test generation. This should record all settings whether they were set automatically or whether the tester set them manually. For this purpose, intervention points must be created in the generation process so that they can be checked and, if necessary, adjusted at relevant points (e.g. when determining the maximum deviation of the approximation). The implementation of the automatic documentation also examines the extent of which it can fulfil requirements for the certification process.

7 Acknowledgement

This paper is based on research work done in the course of the Agile-VT project (contract code: 20X1730J), which is funded by the Federal Ministry of Economic Affairs and Energy in the national LuFo V-3 program.

8 Bibliography

[HT20] D.Hillig, F.Thielecke, „Approach to Systematic Test Signal Definition for Operation Scenarios of Aircraft Systems“, 2nd Workshop on Avionics Systems and Software Engineering (AVIOSE’20), Innsbruck, Austria

[Bo18] Timm, Bodo et al. (2018):“Schlussbericht zum Vorhaben STEVE System-Technik und virtuelle Erprobung: Förderprojekt im Rahmen des Luftfahrtforschungsprogramms LuFoV-1: Laufzeit des Vorhabens: 01.01.2014-30.09.2017“

[SC15] World Wide Web Consortium (W3C). (2015) State chart XML (SCXML): State machine notation for control abstractions. [Online]. Available: https://www.w3c.org/TR/SCXML/

[Fr19] Franke M., Meyer V.HW., Rasche R., Himmler A., Thoben KD. (2019) Interoperability of Test Procedures Between Enterprises. In: Popplewell K., Thoben KD., Knothe T., Poler R. (eds) Enterprise Interoperability VIII. Proceedings of the I-ESA Conferences, vol 9. Springer, Cham.

[Ke01] E. Keogh, S. Chu, D. Hart and M. Pazzani, "An online algorithm for segmenting time series," Proceedings 2001 IEEE International Conference on Data Mining, San Jose, CA, USA, 2001, pp. 289-296

[HT19] M.Halle, F.Thielecke, „Tool Chain for Avionics Design, Development, Integration and Test“, 1st Workshop on Avionics Systems and Software Engineering (AVIOSE’19), Stuttgart, Germany

[La18] J. Langner, J. Bach, L. Ries, S. Otten, M. Holzäpfel and E. Sax, "Estimating the Uniqueness of Test Scenarios derived from Recorded Real-World-Driving-Data using Autoencoders," 2018 IEEE Intelligent Vehicles Symposium (IV), Changshu, 2018, pp. 1860-1866, doi: 10.1109/IVS.2018.8500464

[Ro16] C. Roesener, F. Fahrenkrog, A. Uhlig and L. Eckstein, "A scenario-based assessment approach for automated driving by using time series classification of human-driving behaviour," 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), Rio de Janeiro, 2016, pp. 1360-1365, doi: 10.1109/ITSC.2016.7795734

[BWK05] S. Berner, R. Weber, R. Keller. (2005). Observations and lessons learned from automated testing. 571- 579. 10.1109/ICSE.2005.1553603.

[Ba15] Bach, J., Bauer, K., Holzapfel, M., Hillenbrand, M., & Sax, E. (2015). Control based driving assistant functions' test using recorded in field data.