

Modelling and simulation of the European Train Control System for test case generation

H.-M. Schultz^a, M. Meyer zu Hörste^a, B. Ptok^b & E. Schneider^a

^aInstitute of Control and Automation Engineering, Technical University of Braunschweig, Langer Kamp 8, D-38108 Braunschweig, Germany

Email: H.Schultz@tu-bs.de

^bDeutsche Bahn AG, Zentralbereich Neue Systeme, Völckerstr.5, D-80939 München, Germany

Abstract

The European Train Control System (ETCS) is in development. Based on its specification in natural language a formal Petri net model has been established using the tool Design/CPN by order of the Deutsche Bahn AG. Thus, an industrial scale example exists to investigate the impact of formal models on the product life cycle. This paper focus' on the test for approval since this will be in the next future a major matter of concern of the railway companies, as the industry will deliver first prototypes of the ETCS soon.

1 Introduction

1.1 Background

The process of European harmonisation is still in progress, which also concerns the railway companies. Each European railway company has one or more train control systems, which are mostly incompatible. So every train which shall run through different countries has to be equipped with several systems, or the train traction units and drivers have to be changed at every border. As a consequence, the first solution requires a complex system on-board, which leads to high installation and maintenance costs. The second one is a time consuming solution, which leads to increased operational costs. Thus, it is important to define a train

650 Computers in Railways

control system which will be a standard for all countries and provides a uniform and language-independent signalling information for the man-machine-interface (MMI).

Moreover, the increase of maximum speed of new developed trains demands a continuous, electronic communication based train control system, because at a travelling speed of more than 160 km/h the track side signals aren't surely recognisable any more.

The European Train Control System (ETCS) which is a part of the harmonised European Rail Traffic Management System (ERTMS) matches these two targets:

- It is a continuous, electronic communication based train control system, which is useful for both high and low speed railway services and
- it defines a standard for a uniform signalling system on the MMI (Man-Machine Interface) of the traction unit, so neither locomotives nor drivers have to be replaced when crossing borders. The textual display on the MMI is adaptable to correspond with the driver's language.

1.2 State-of-the-art in implementing ETCS

The development of the ERTMS and the ETCS is in progress. The specification and test of the new system on several test sites in Europe is founded by the European Commission/DG VII. The ETCS Specification written by the EEIG ERTMS Users Group describes the system in a textual, informal way. Several companies in Europe are working on implementations of ETCS components.

The first test called EMSET (European Madrid Seville Eurocab Test) is planned to be performed in Spain. This test is in preparation and should start this year. Also the test sites in Germany, France and Italy are already in preparation.

So, the task for the near future is to develop a method for the examination for the approval of the ETCS.

2 Positioning of Testing vs. Verification and Validation

Depending on the authors standpoint different definitions exist of what is to be understood by „test“. Therefore a short positioning of test in the life-cycle of a software product will be given.

We define „test“ as the central process of executing code to find errors, including the preparation of test input and the analysis of the results. Since code never runs on its own, we extend the definition by including

the hardware into a system under test (SUT). In [3] validation is defined as the determination of the correctness of the product of software development with respect to the user needs and requirements. Verification is the evaluation of a system to determine whether the products of the given development phase satisfy the conditions imposed at the start of that phase. Thus, validation and verification work with the same means of description as the specification process. Mostly, these are not executable. The object of test is readily implemented, executable code.

In the classic product life-cycle, test has been regarded as one of multiple steps from planning to the system [3]. Since this approach has shown many disadvantages economically as well as technically for large, complex systems like the ETCS, test has been positioned as a parallel process with close interactions to the system development process (cf. Fig. 1a).

Since an unstructured approach for testing is likely to result in problems in practice, each testing itself has to be regarded as a process (cf. Fig. 1b). Many approaches exist for every step of the test process.

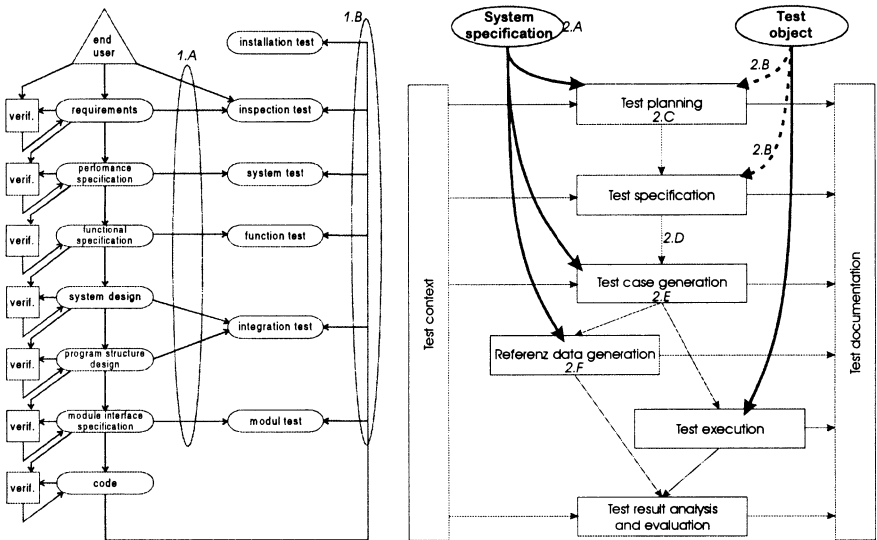


Fig 1 a+b: The integration of higher order tests [5] + The test process

3 The complexity of testing

As tests have been well positioned into the life-cycle by management (cf. Fig. 1a), now there are many points from where the complexity starts for the test.

Complexity arises from three main sources:

1. the integration of the tests in the product life cycle, esp. the combination of the results of different tests(cf. Fig. 1a)
2. the steps of the test process (cf. Fig. 1b)
3. the properties of the process to be tested

The complexity sources 1 and 2 have been marked in the figures above and can be explained as follows:

- 1.A - Different forms and means of description of input documents require different techniques and tools to use them for testing.
- 1.B - Since different techniques for the test case definition and the documentation are used, the tuning of each test to fit in the hierarchy of the different test levels and their summarisation are difficult.
- 2.A - Multiple input documents have to be considered and brought together to establish the reference of the test.
- 2.B - As only the test object (mostly the code) represents a structure reflecting the whole complexity to be dealt with, many test strategies are based on the test object itself, hence errors may be left undiscovered if they are not too obvious for the tester.
- 2.C - Test planning deals with the problem of defining which aspects have to be tested and to what degree of complexity. If exhaustive testing will be impossible, test quality measures have to be defined.
- 2.D - Test specification comprises of the development of the tools for the later test case generation, e.g. data flow analyser, cause-effect graphing, execution time analyser etc.
- 2.E - The test case generation can be supported by tools only if a formalised basis has been established in the previous step. Their complete coverage of the defined quality measures must be proven.
- 2.F - The reference data generation is based on the analysis of the system specification. An overall view of all aspects of the SUT must be derived, i.e. internal states, concurrency of processes etc.

Last, the SUT has some complexity-increasing properties: hard real-time constraints, a large number of interacting functions, multiple, not fully defined parallelism of operational procedures (cf. scenarios chap. 4), a large number of state variables, each with its own high variability. Furthermore, the ETCS is an open system and its interfaces are only

defined from the point of view of the system itself. The environment has to be considered as infinite variable.

4 Modelling approach

Modelling of control systems starts from different modelling paradigms. Before starting to model it is necessary to make sure that the methodology is suitable for achieving the modelling objective. A specific model for system functionality will be quite different from a model used for availability considerations.

In modelling the European Train Control System, different modelling aspects have been integrated [1, 2]:

- processes
- scenarios
- functions,

are shown on different model levels.

When modelling the process aspect, the focus is on communication and interaction of different subsystems. A general representation of these nets shows the subsystems and their interfaces, every subsystem being detailed on additional levels.

The scenario-based aspect is the modelling of operational procedures. Its main elements are the interaction between on-board and trackside equipment and the sequence of events required to maintain operation. Individual scenarios are connected to form groups and in this way they are integrated into the process model.

The functions are represented at lower model levels. The functions are specifically associated with the objects of the process aspect and represent the activities or the response to interaction requirements following from the scenarios. Functions are activated by internal systems states or events from scenarios. Some functional modules can be used in different objects and are hence modelled in so-called "functional blocks". These functional blocks are modelled as separate nets and can serve as functions in the different scenarios, without infringing the modelling principle of hierarchic decomposition. At the same time ensures that sub-nets and implementation components can be reused.

Modelling Paradigm	Aspect focused on
Process	Integrated system, e.g. interaction and communication between the on board and trackside (RBC) system

654 Computers in Railways

Scenario	specific operational procedures and the necessary interaction e.g. joining, splitting, ...
Function	separated functionality's e.g. release path, train absolute location

Systems modelled in this way by necessity are modular in structure - the hierarchic aspect being maintained at the same time. Because of their structure, models are easier to comprehend, easier to adapt, and they can be reused. Partitioning aspects can be considered, and symmetries can be utilised.

4.1 Up to date results

For the application of formal analysis methods to this system, two models have been developed. The partition of the model follows the partition of the real system, both the on-board and the track side system are represented each by one hierarchical model. Like the real system these models have interfaces where the data-exchange can be observed. The nets are divided into different types according to their position in the structure and their content. The Petri net models are implemented separately, so different models may be simulated on separate workstations. A simulation runs through a sequence of scenarios which all contain several paths.

In the actual state the two models consist of together about 200 nets and 2900 transitions and places. The application logic is supplemented by some additional sub-nets which perform supporting tasks like time synchronisation, stimuli generation, composing and decomposing telegrams etc.

Also a smaller model has been developed, which is simulating the environment, i.e. the track, the physical train etc. Two more models of the interlocking and the regulation are still in working progress.

Even if the modelling started this time as a test run after the completion of the systems specification, the technical possibilities for the introduction of Petri net as a central point in the product life cycle can be analysed. Already with this approach, the model could support the specification process by uncovering errors and by providing further inside in the problems of the domain by connecting the documents delivered.

5 Starting point for the use of a Petri net model for testing

The derived Petri net model offers multiple possibilities to be used during the whole life-cycle.

If Petri nets are used and refined from step to step in the life-cycle, they should be given the status of the only valid documents to refer to. This is possible due to their inherent properties as syntax check, integrated presentation of all system properties (depending of the Petri net dialect), and simulation possibilities. A Petri net model is more likely to be exact, complete and consistent than the documents it replaces. In each step of the development phase, verification and validation is carried out on the same object and thus efforts are concentrated. If other means of description are required to be accepted for approval by state authorities or system operators with deep knowledge in the application domain but less in information techniques, they should be derived from the checked model.

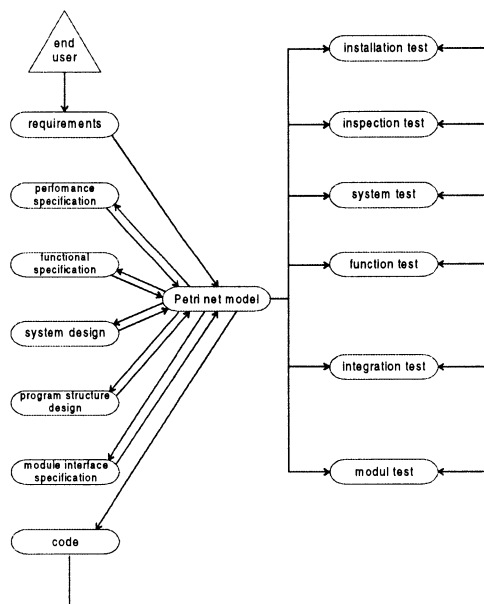


Fig 2 The central role of the Petri net model in the development process

Since a Petri net model can be executed, tests will cover a much broader domain. Each validation done with the model may be regarded as a test,

656 Computers in Railways

therefore letting test begin in the very early stage of the life-cycle. But, since the Petri net will not replace the code in the system in service, the definition of test will remain unchanged.

For each of the problems that is due to the sources of complexity mentioned above, the Petri net model helps to deal with:

- 1.A - There is now only one method to be dealt with. Thus, efforts for tool support can be concentrated.
- 1.B - As 1.A and all output will have the same format each test level can be easily adjusted to the other ones.
- 2.A - There is now only one reference for the test covering all aspects of the system.
- 2.B - Since the Petri net contains already all aspects of complexity of the application domain of the system developed, there is no longer the need to examine the test object to gain further inside in the domain.
- 2.C - The complexity due to system functionality can not be reduced but tools to support test quality criteria, e.g. function coverage, path coverage, adapt more easily to a formal method compared to a textual specification. The planning of test goals and the choice of test case selection criteria are better to justify because more inside in the domain can be gathered by execution of the model.
- 2.D - Since the Petri net model is a graphical representation itself, it can replace some of the techniques used before (e.g. cause-effect graph, data flow graph)
- 2.E - Test case generation could be automated if the selection rules for test cases according to the chosen criteria can be formalised and coupled with the model.
- 2.F - The reference data generation reduces to the matter of executing the model for each test case.

Finally, after these examples of how to use a formal Petri net model to support test, there exist other possibilities to extend their use. If the system developed may be subdivided into modules and these modules have the same interfaces as their correspondents in the Petri net model, the hardware where the model runs on will provide the basis for a hardware-in-the-loop test (HIL). Thus, the expensive development of a test environment may be substituted by much less effort which is necessary to extend the model to this point. It shall not be concealed that many Petri net tools are not capable to reach this since their overhead slows down the computer platform. Furthermore, measures to cover real-time requirements are not established in the tools.

6 Promising outlook into a Petri net based R+D environment

One of the most demanding tasks is the evaluation of the test carried out. Depending of the planned test goals, criteria have to be established and their degree of fulfilment has to be measured. This measure can than be expressed as a quotient like

$$\text{quality of test} \sim \frac{\text{number of test cases fulfilled}}{\text{number of all possible test cases}}$$

The more responsibility the developed system has to bear the more completeness of testing will be demanded. There are two possibilities to increase test quality: increase the number of performed tests or decrease the number of all possible test cases. The first way is limited by to the hardware to run the tests on and the labour effort for the test preparation. Advances in speed result only in a linear increase of test run, we believe that it will never be possible to reach completeness this way.

The second possibility demands to decrease the set of test cases to be taken into account. Therefore we may choose well defined criteria for the selection of a subset of test cases which will be considered as a sufficient representative of all possible test cases. All the definitions of such criteria are based on models about the system and its environment. The forming of these models can be supported by tools. But the evaluation of their efficiency is left to the user. This implies the great risk of neglecting critical subsets of possible test cases.

Another approach is to decrease the set of all possible test at all. Thus without choosing just a subset the quality of the test is raised. Furthermore, on the basis of the reduced set it is more easy to reason about the quality of a choice of a subset.

The number of the internal states of the SUT is defined by the complexity of its functionality and can not be reduced. But, as we gain more knowledge about the environment of the SUT we are able to reduce the complexity of the input at the system interfaces. Therefore, complex models of the environment would have to be established. This effort can be spared if Petri net models exist already from the development of the neighbouring systems, i.e. a long term strategy to use this modern approach of using a central model will result in a disproportional reduction of effort and in a gain of confidence in testing. If such models do not already exist, some aspects of this approach can be applied by using a reduced model of the environment showing its undisturbed operational behaviour.

658 Computers in Railways

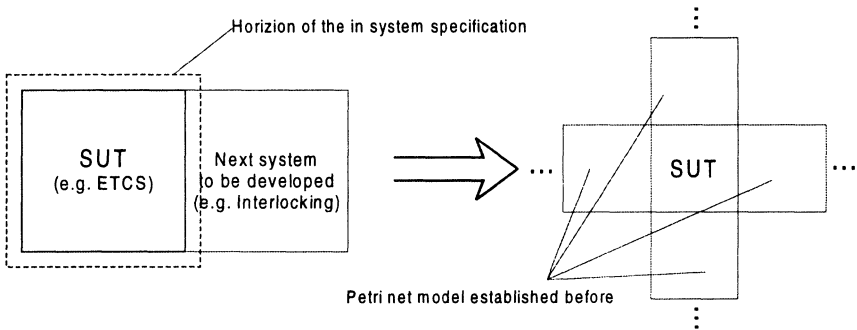


Fig 3 Gain through a uniformly Petri net based R+D strategy

References

- [1] Janhsen, A.; Lemmer, K.; Ptok, B.; Schnieder, E., Modelling and Simulation of the new European Train Control System, IMACS Symposium on Mathematical Modelling, Vienna, pp. 473 - 478, 1997.
- [2] Janhsen, A.; Lemmer, K.; Ptok, B.; Schnieder, E., Formal Specification of the European Train Control System, IFAC Conference on Transportation Systems, Chania, Greece, 1997.
- [3] British Computer Society, Specialist Interest Group in Software Testing, *Standard for Software Component Testing*, Vers. 3.1, 1996.
- [4] Ott, H.J., *Software-Systementwicklung*, Hanser-Verlag, München, p. 13, 1991.
- [5] Myers, Glenford J., *Methodisches Testen von Programmen*, München; Oldenbourg Verlag, 1991. (orig.: *The art of software testing*)