AN ENVIRONMENTAL SIMULATION TESTER AS APPLIED TO TRAFFIC CHARACTERISTICS EVALUATION

Zeny Koono, Soichiro Matsumoto, Rikuo Ozaki, Masahiro Soga and Kenichi Ozaki*

Totsuka Works
Hitachi, Ltd.
Yokohama, Japan

* Hitachi Electronics Engineering Ltd.
Ashigara Kamigun
Kanagawa, Japan

Presentday situation makes development burden of conventional environmental simulators heavier. This paper proposes a one machine concept for an environmental simulator to include also external function tests. Environmental simulations are performed by a kind of test programs. In order to be capable of programing environmental simulations without increasing normal test program productivity, two levels of high level macro languages are provided. Macros for traffic evaluation and concurrent operation of tests makes environmental simulations easier than before. As the system is structurally designed in system, software and hardware, the system is flexible and expandable. The operating experiences proved the wide range capability.

1. INTRODUCTION

This paper reports an expanded environmental simulation tester.

In the development of electronic switching systems, call handling capacity is an important performance factor. Environmental simulators have been used for this purpose. Formerly a big switching system development project usually developed its own environmental simulator and used it for traffic evaluation and software aging. As the number of service features (F) and the number of switching systems (S) in a switching system manufacturer becomes large, to develop FxS times environmental simulators is becoming impossible.

This paper first overviews the current situation and points out one machine concept which covers software and system test including traffic evaluation and is applicable to various switches. Design target, design philosophy, architecture and actual operating experience will be explained. (Ref. 1)

2. ENVIRONMENTAL SIMULATORS IN SWITCHING SYSTEM DEVELOPMENT

2.1. Traffic Evaluation

The object of traffic evaluation in a switching system is to provide services at a good quality with economy. Quality means blocking probability and response delay. Theoretical calculations or computer simulations are usually used for planning to keep the response time limit. Systems engineers decide processor systems architecture, the on-line program processing structure and maximum permissible dynamic steps for respective programs. At the final stage of development, evaluations on the realized system are made by applying load by traffic generators or an "Environmental Simulator". It is a big stored program controlled simulator that simulates the actual environment of a switching system. The response times are measured and the system receives tuning if necessary.
2.2. Current Status of Switching System Development

Present day switching system development faces (1) new service feature addition every year, (2) some new services consume much larger dynamic steps/call, (3) need for very large systems (4) much shorter response time for such as ISDN data service, and (5) many versions of switch such as old space division and time division or domestic and export. Figure 1 shows a trend of service features of a series of PBX. In step with adding ISDN features, several hundreds of PBX features must be implemented on an existing central office software. Figure 2 shows a trend of the maximum call handling capacity of switching systems. It increases like a computer MIPS curve. The maximum number of ports (telephone lines and trunk circuits) reaches 300,000 to 500,000 now.

These circumstances result in an increase of switching software developments. Various supports have been provided for increasing switching software productivity. An environmental simulator is one of them but used only at the final part of the development for evaluation and aging. Big development efforts, such as to provide respective environmental simulators and evolve their features and increase their capacity, can not be assigned for them.

A software development consists of design and testing. Early parts of the test are executed on general purpose computers. But tests using actual hardware are indispensable. They are;
   (1) Verification of a new feature.
   (2) Combination tests of the new and related features.
   (3) Regression tests in order to assure that the new feature addition does not degrade existing hundreds of features.
   (4) System tests for functions and loads (using environmental simulators).
   (5) Forced aging. Heavy and worst load is applied for a certain period (Using environmental simulators).

2.3. Future Direction

Clearly the way to solve these problems for an environmental simulator is to take over all these jobs. Usually terminal (such as a telephone) operating procedures for environmental simulation are more complicated than usual ones for a feature functional test. Therefore, if an environmental simulator is to designed so as to be operated by test procedures, one hardware system may be utilized for all tests described above. In order to apply heavy traffic, the use of many such testers instead of a conventional large one is more reasonable. For further heavier load, conventional traffic generators may be used with.
3. TEST SYSTEM

3.1. Design Objectives

Based upon the various needs described before, the test system's requirements or design objectives were settled as follows:

1. Wide application area
   - All functional tests
   - All traffic tests
   - Signaling system expandability
   - Terminal expandability
2. Wide range and rigorous tests
   - Terminal operating procedure, transmission and signalling
   - Many concurrent tests
   - A number of terminal circuits
3. Low development cost for tests
   - High test program productivity
   - On site compiling and running
   - Continuous test operation
4. Environmental simulator
   - Random call origination or continuous test sequence
   - Traffic evaluation data such as number counts and response time distribution
   - Easy tool for system tuning

3.2. Design Philosophy

The approach taken here is to list all requirements for the tester to cover, tear down all these requirements to a fractional requirement and think out the solution to fulfill OR-set of these fractional requirements. Although variations might be many, the basic requirements are not so many. Systematic procedure such as structured design and concurrent operating system are powerful keys to solve the problem. The design philosophy is described here.

1. One machine concept by stored program control
   - Embedded system program for respective switch.
   - Rather smaller system fitted for various tests and use many of them for applying heavier loads.
2. Multidimensional hierarchical modular structure of function and matching software
   - Hierarchical software structure with standardized interfaces.
   - User aspect; test programs (at the top), signaling protocols (in the middle) and hardware access (at the lower).
   - Implementation aspect; administration, test subsystem and operating system.
   - Hierarchical data structure, communication, and language.
   - Hierarchical hardware structure.
3. Distributed state transition method (Ref. 2)
   - The tester is, like a switching system, a Finite State Machine. The system is hierarchically broken down to a composite system of various Finite State Machines. Transition from states to others form structurally partitioned programs.
   - Conventional structured design is applicable hereafter.
4. SDL machine
   - State, event and task expressions of CCITT-SDL (Specification and Description Language) enable to express Finite State Machines.
   - Direct execution of state transition macros (CCITT-SDL).
   - Built-in SDL-compiler for instantaneous programming and running.
5. Event driven and concurrent operating system
   - State transition control.
   - Four layered hierarchical sequential process.
   - SDL processor (interpretively executed).
3.3. Architecture

Requirements and design philosophy resulted in the tester architecture. This section describes them with figures. Functional partitioning of both software and hardware will be shown.

Figure 3 shows the system. As shown in the Figure, the tester is externally connected to the system to be tested. Any external function tests are applicable. Simulators (telephones, trunk circuits and attendants) operate according to test procedures such as first off hook, then dial.... Tests are loaded from the system disc of a minicomputer.

Figure 4 shows hierarchy of switching related software together with associated signals. All interfaces are standardized and chosen to be as logical as possible in order to make layered software independent of each other. In a layer, programs are partitioned to be monofunctional. These hierarchical structures enable the software to accomodates various applications easily. Top and middle layer softwares are all sequential and run by programs written by "State Transition Macros." This is an important provision for programing a sequential machine easily.

Figure 5 explains the principle of operation. A test is made using several terminals (telephone, trunk circuit or attendant). They form a Test Group. Users describe operations of terminals for a test (Test Scenario). SDL-compiler converts Test Scenario to object programs and they then are loaded to each simulator. ( in the Figure) Operator's commands such as "Start Test (Test Group i)" are converted to standardized packet and transmitted to Test Control (TGi). Then, it broadcasts "Start Test (TGi)" to Supervisors (in three processors). Thus triggered Supervisors send start commands to the simulator circuits belonging to TGi. ( in the Figure). Telephonic signals are detected by Signal Detectors and activate Test Programs through Signaling Sequences. ( in the Figure) Mark in boxes denotes that SDL is used. Signal Detectors and Signalling Sequences are provided for each version of switching system. Due to the hierachical monofunctional structure and logical interface, differences between various switching systems are minimized. All timing values are specified by hierachical data. These provisions lead to easy adaptation to various switching systems.
1. On Hook Tel. A and B.
2. OffHook Tel. A. Check Dial Tone.
3. Dial xxxx.
4. Check that Tel B is ringing.
5. Off Hook Tel. B. Check answer is received.

FIGURE 6

FIGURE 7
3.4. Operation

Figure 6 shows the operational flow. A test at the left is converted to operations of terminals named "Test Scenario" as shown at the center bottom. Each box in the flow chart corresponds to Scenario Macro described later. A completed flow chart with abnormal routes, is then converted to a test program comprising of State Transition Macro code through SDL-compiler.

Control of tests such as start, stop, repeat are commanded by the operator. These commands may be catalogued and an automatic operation by this catalogue is possible. For the purpose of the environmental simulation, a member of tests (currently hardware limited to max. 48, but software limit 255) can run at a time.

In order to test, user must define the operation of each terminal. This makes the test program design a little more difficult. But for the system function test and the environmental simulation, it is indispensable. These state transition programs are rather tedious for users. Therefore, a higher level language "Senario Macro" is provided. They correspond to elemental test procedures. Each macro corresponds to a fraction of state transition programs.

They are standardized to have standardized exits; normal, busy, no answer, abandoned and abnormal as shown in Figure 7.a. If function tests only are considered, two exits (normal and abnormal) are enough. This is a special feature to cover both function testers and an environmental simulator. Scenario Macros in a test program are linked together with symbolic addresses as shown in Figure 7.b and they are then converted to a bigger state transition program as a whole as shown in Figure 7.c. State transition programs are sequential and still complicated. Therefore they are expressed in macros named "State Transition Macro".

For traffic evaluation, several macros are provided. Hatched box "Number count" in Figure 8, is one of them. It increments a counter as the program passes through. Thus user can count number of call origination, busy, completed etc. as he needs. Other macros for traffic evaluation are provided.

For measuring response delay distribution, start of timing is recorded by "Mark Timing". On receiving the response, "Classify" measures time difference and increments counter in the corresponding column. Number counters and delay distribution tables are read out and displayed.

![FIGURE 8 Macros for traffic evaluation](image-url)
4. OPERATING EXPERIENCE

As the first application is a PBX development, it provides 16 telephone simula­
tors, 16 trunk circuit simulators and 16 attendant simulators. Up to now, three (domestic, export and integrated) versions have been developed.
Adaptation of each system has been very simple.

As for the preparation of tests, programming itself is very easy. Overall pro­
ductivity is the best in feature software designers and the next inspectors.
(Situation is like in using GPSS, where clear understanding of the model to be
simulated and exact knowledge of the language is indispensable.) As users gain
experiences, they begin to mix State Transition Macros in their program to test
more details. Mixed use of two macros is successful.

For tests, the tester proved its rigorous checking ability. Repeated tests
find out software and hardware instability. Regression tests find degradations
easily. It proved to bring about a lot of man-power saving and usefulness
tool for quality assurance.

For traffic evaluations, environmental simulations are usually attained by
selecting some tests from those already done and run several of them at a time.
Thus apparently no extra development cost is needed for environmental simula­
tions. Most of the efforts are spent in finding what is wrong and in tuning.
It welcomes that checking ability in depth and the same environment is appli­
cable repeatedly.

5. CONCLUSION

Summarizing, an environmental simulation tester has been developed success­
fully. Presentday situation of electronic switching system development
requires one machine concept which is applicable to any tests as well as
environmental simulation tests for any switches. The target has been achieved
by structuring. A wide range of objectives are broken down to a set of elemen­
tary requirements. These requirements are implemented by hierarchical software
and data structure with standardized logical interfaces. This system may be
converted to a prototyper as indicated in Figure 6. These approaches, such as
very high level language and SDL together with distributed state transition
method are assumed to be also applicable to electronic switching system deve­
lopment. (Ref. 3)

ACKNOWLEDGEMENTS

The authors express their deep appreciation to those who participated in this
system development. Also their sincere thanks to users for their contribution.

REFERENCES

(1) Koono, Z., Kimura, T., Iwamoto, M. and Soga, M., A Stored Program
Controlled Environmental Function Tester Based On FMM/SDL Design,
International Switching Symposium 1987 B9.5.1
(2) Hiyama, K., Mizuhara, N., Mochizuki, K. and Koono, Z., A Software System For
Electronic Switching System Using Distributed State Transition Method,
International Conference on Communication 1982 5C.3.1
(3) Aoyama, T., Tokita, Y., Inokuma, Y., Mizuhara, N. and Koono, Z., Unified
Switching Software Architecture Using SDL Design, Proceedings of
International Telecommunication Symposium 1986 10. 3