9

Design and Implementation of an Interconnectability Testing System - AICTS

Ken'ichi Takahashi, Shinji Suzuki, Katsuya Sawai,
Masatoshi Hatafuku, Kunihiro Gotoh, Keiichi Kazama

Advanced Intelligent Communication System Laboratories (AIC)
6-6-3, Minami Yoshinari, Aoba-ku, Sendai, 989-32 Japan

Telephone : +81-22-279-3310 Facsimile : +81-22-279-3640
email: ktaka@aic.co.jp kaze@aic.co.jp suzuki@aic.co.jp

Abstract

The OSI (Open Systems Interconnection) has been increasingly recognized and used for interconnection among different communications systems. Usually, conformance testing or interoperability testing is done to confirm the interconnectability of those different systems. But it is usually recognized that only these types of testing are inadequate for interconnectability confirmation. We have developed an interconnectability testing architecture by extending the architecture currently used for conformance and interoperability testing. With this architecture as the base, we are making efforts to develop an interconnectability testing system named AICTS (AIC's Interconnectability Testing System). Testing with this system is intended to complement ordinary conformance and interoperability tests. So far, we have completed the design of the system and developed part of its functions. In this paper, we discuss our interconnectability testing architecture as an extension of the existing architecture for conformance and interoperability testing. Also, this paper explains the structure of our testing system and presents example of test execution.

Keywords: conformance testing, interoperability testing, testing and debugging system

1. Introduction

The use of the OSI standards has been spreading as a means to connect distributed computer resources with each other through communications networks[1]. However, the OSI standards allow users to select options or parameters. Sometimes, these standards do not decide the details for enhanced extensibility and flexibility of communications systems. This means that the OSI standards leave the possibility that the functions implemented in the OSI systems by individual users may be inconsistent with each other. The communications systems implemented based on the OSI standards do not necessarily establish interconnectability. A lot of work has been done to design and develop the conformance and interoperability testing systems to check for inconsistencies of the implementation to enhance their interconnectability [2],[3],[4],[5]. But it seems that testing with these systems may involve the following problems: The conformance testing system deals with an IUT (Implementation Under Test) that made up the OSI reference model layers [Fig.1],[2],[3]. This testing is intended just to detect if

T. Mizuno et al. (eds.), *Protocol Test Systems*
© Springer Science+Business Media Dordrecht 1995

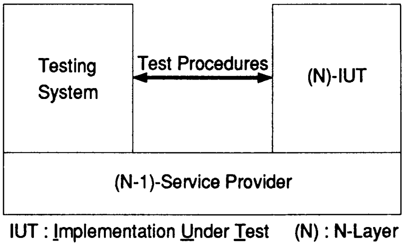


Figure 1 : An architecture of conformance testing

the IUT works according to the standard: it does not confirm the IUT for connectability with other systems. It may fail to detect any functional inconsistency of the IUTs to be interconnected.

Hence, the interoperability testing that is able to detect the inconsistency functions between the IUTs is necessary. This testing can be done either with or without placing a testing system between the IUTs to be connected[5],[6]. Usually, the latter method is used. In this case, the testing can be done most easily by connecting products each made up of all OSI layers and having their behavior controlled and observed by human testers or testing application software[Fig.2]. However, this method does not have any point to observe the behavior of each layer, it is difficult to obtain detailed test results comparable with those obtainable from conformance testing. Also, when conducting the interoperability testing on a layer-by-layer basis instead of between products made up of all the OSI layers, this testing is done by focusing only on the interworking of services provided by each IUT to upper layers without giving heed to services exchanged with lower layers. This type of testing may fail to observe the acts of the IUTs against abnormal behavior of lower layers. The interoperability testing of this type does not go beyond checking IUTs for normal interworking.

There are a number of the reports about the interoperability testing using a testing system between IUTs to be connected[6],[7]. Unlike the case of conformance testing, there is no established method for interoperability testing. Consequently, a variety of methods are employed, producing diverse test results. We think the existing methods for interoperability testing are inadequate to establish interconnection for a variety of communications systems.

To resolve the issue, we intend to establish a standard interoperability testing architecture as with conformance testing. We also plan to establish a testing system configuration capable

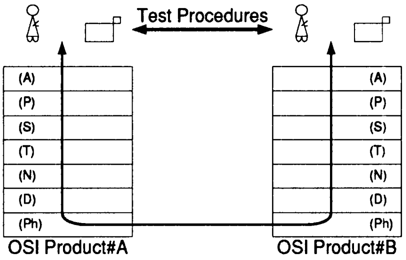


Figure 2 : An architecture of interoperability testing

of covering the shortcomings of the conventional conformance and interoperability testing methods by extending them. Further, we will build an integrated testing environment in which conformance and interoperability tests can be conducted.

Section 2 discusses the basic strategy of our proposed interconnectability testing architecture and the model of the testing system. Section 3 explains our testing system configuration, testing procedure and sample of the testing. In Section 4, we summarize our study and give our future plans.

2. Basic Strategy

2.1 Interconnectability testing

Because of the possible use of different parameter, option and other settings as stated above, OSI products may be made up of layers implementing mutually inconsistent functions. Conformance testing is insufficient for the purpose of sorting out any inconsistent implemented functions between connected IUTs, this testing is performed to detect the illegal behavior against the OSI standards. Ideally, conformance testing should be done by exhausting all the possible states of each IUT. However, this is practically difficult in terms of time and cost because a very large number of IUT states are needed. To find inconsistencies or illegalities that may be overlooked by conformance testing, interoperability testing, as shown in Figure 2, is intended to confirm the interconnectability of the OSI products made up of all the seven layers of the OSI reference model with respect to their behavior. It seems that this usual interoperability testing involves the following problems because it does not use any testing system similar to that used with conformance testing.

(1) When abnormal behavior in interworking between the OSI products arises, it is difficult to identify the functions or the layers that caused the abnormal behavior because it does not control or observe layer-by-layer behavior.

(2) When focusing on the implementation to be connected at a certain layer, it is impossible to observe the interworking with respect to any abnormal behavior of the lower layers. With this interoperability testing, it is impossible to check whether there is a means to solve an abnormal behavior with the lower layer of each implementation.

It seems that these problems can be resolved by placing a testing system between single-layer IUTs and providing PCOs (Points of Control and Observation) at the upper and lower levels of both IUTs. This is because the provision of the upper and lower PCOs makes it possible to control and observe the use of lower services of both IUTs, the status of services to be provided to the upper layers of both IUTs, and the protocol-level behavior of each IUTs with respect to the other. Since this test method gives data which is as detailed as that obtained by conformance testing, it is possible to determine interconnectability more accurately. Also, since the testing system can give the lower PCO stimulus that simulates an abnormal condition, it is possible to observe how both IUTs coordinate with each other in the event of a lower layer abnormal condition. We are making efforts to establish this test method and also develop a testing system which will be an extension of the usual conformance and interoperability testing system configurations. Testing based on this extended configuration can be regarded as variation of interoperability testing. In this paper, we call it ***interconnectability testing*** in order to distinguish it from interoperability testing. Figure 3 is a conceptual view of interconnectability testing. We are now developing an interconnectability testing system based on the interconnectability testing method. Table 1 gives the purposes of individual tests. Figure 4 illustrates testing stages for checking interconnectability between the OSI products. Stage 1, conformance testing is intended to see if the IUTs involved are implemented in conformity with the OSI standards. In stage 2, IUTs which qualified the conformance testing are connected each other and their interworking is controlled and observed by the interconnectability testing system placed between them. Finally, the OSI products made up of the IUTs that passed these tests are mutually connected and are put under interoperability testing to detect their overall product interconnectability. We believe addition of this type of interconnectability testing to the conventional testing would contribute toward enhancing the interconnectability of the OSI communications systems.

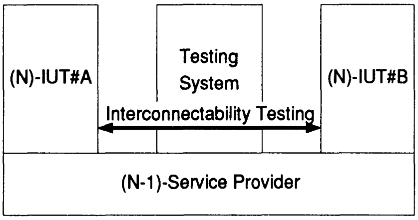


Figure 3 : Method of interconnectability testing

Table 1 : Purposes of testing

Testing	Purposes of testing
CT	protocol behavior of an IUT
ICT	interconnection between IUTs
IOT	interconnection between OSI products

CT : Conformance Testing
ICT : InterConnectability Testing
IOT : InterOperability Testing

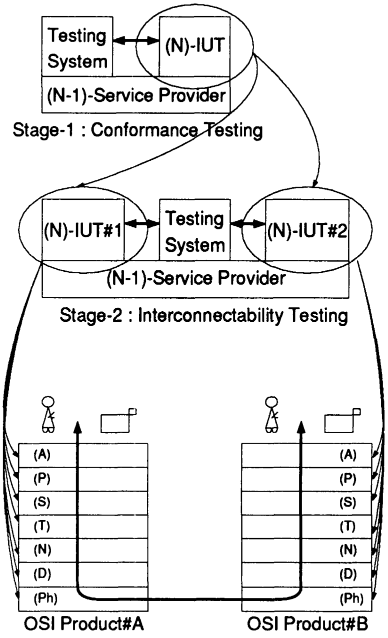
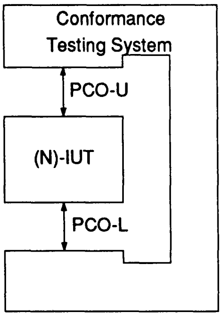


Figure 4 : Testing stages

2.2 Interconnectability testing model

Figure 5 and Figure 6 are models of conformance testing and interconnectability testing, respectively. In the conformance testing, the testing system deals with an IUT and analyzes its response for conformity with the standards. In this manner, the conformance testing determines whether the functions of an IUT follows the OSI standards. In Figure 5, for example, the conformance testing system gives a stimulus at the upper PCO (PCO-U), observes its response at the lower PCO (PCO-L), analyzes response with the stimulus and thereby determines whether the implemented IUT functions are in conformity with the standards. The interconnectability



PCO-U,L : Point of Control and Observation - Upper,Lower

Figure 5 : Model of conformance testing

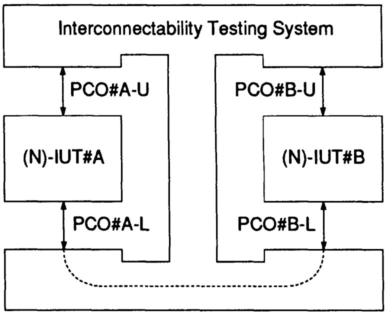


Figure 6 : Model of interconnectability testing

testing system observes and analyzes how one IUT reacts with respect to the stimulus given to the other IUT and thereby determines the interconnectability of the IUTs. Figure 6 shows a model of interconnectability testing. The interconnectability testing system gives stimulus at PCO#A-U, observes its response at PCO#A-L, gives the response to PCO#B-L, and observes the response as the result of the interworking at PCO#B-U. At this time, if the interworking of the mutually connected IUTs is in conformity with the standards, the response observed at PCO#B-U should be as expected in advance. If either IUT is in an abnormal condition, the response observed would be different from the expected one. For effective, reliable interconnectability testing, we have provided our testing system with the following features described in the following sections.

2.3 The lower tester, test channel and test management channel

Our assumption is to perform interconnectability testing for remote IUTs. We have, therefore, employed in our testing system configuration the distributed testing method of conformance testing defined by ISO9646[2]. The testers used for the distributed testing include a lower tester and upper tester, which correspond to the lower PCO and upper PCO, respectively. The upper tester, in combination with the IUT, makes up an SUT (System Under Test). These testers perform testing through the coordination of work based on the testing coordination procedure. The coordination of work is realized through the transmission and receiving of test management protocol data units (TM-PDUs) defined by the test management protocol (TMP). TM-PDU transmission and receiving take place on the test management channel provided by the lower layer service. Also, transmission and receiving of PDUs for testing between the IUT and lower tester take place on the lower layer connection under test. This configuration requires two channels to establish the connection of test management and connection under test. But, the behavior of both connections do not affect each other, it is expected to bring an improvement in the reliability of testing and the degree of freedom of testing operations.

The testing environment should simulate the actual IUT operation environment as closely as possible. It is undesirable that the testing environment is different from actual IUT operation environment for testing (or testing system) reasons. The operating environment is given primarily by the service provider of the lower layer used for actual IUT operation. The use of lower services of the IUT is controlled and observed by the lower tester at the lower PCO. In

Figure 6, the lower PCOs are placed immediately below the IUTs. In usual distributed testing, it is difficult to place the lower tester immediately below the IUT and control and observe the lower PCO directly. The lower tester is needed to perfectly simulate the behavior of the lower service provider used for IUT operation, this is difficult to realize when a variety of testing systems are made. For example, there is a case in which the lower layer is a ordinarily used public communications network. In such a case, it is difficult to place the lower tester in the public network. Therefore, the lower tester for our interconnectability testing is designed to control and observe the lower PCO indirectly by means of the lower-layer service used by the IUT. Figure 7 shows a configuration of our interconnectability testing system consisting of connection of test management, connection under test, lower testers, upper testers and IUT.

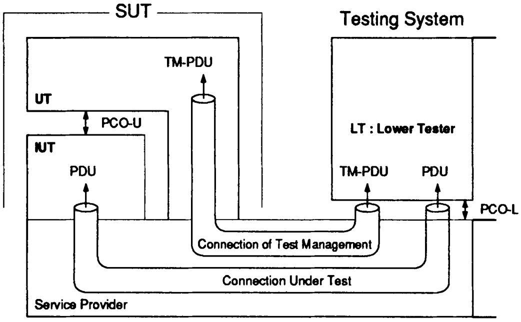


Figure 7 : Lower tester, SUT and channels

2.4 Upper tester

The upper tester controls and observes the upper PCO in coordination with the lower tester. There is a usual method which uses the upper tester working passive as instructed by the lower tester. All the testing behavior of the passive upper tester depend on instructions given by the lower tester. That is, an instruction to control the upper PCO is sent from the lower tester to the upper tester, which informs the lower tester of the receipt and execution of the instruction. The lower tester is informed of the responses observed at the upper PCO one by one. Examples are the astride test method and passive ferry test method[6][7].

We have developed an actively working upper tester by extending the functions of the passive upper tester. The testing behavior of the active upper tester is decided by the tester itself. The active upper tester knows the sequence of controlling and observing the IUT. The active upper tester gives stimulus to the upper PCO of the IUT and informs the lower tester of the responses observed at the upper PCO one by one.

Testing by using the active upper tester is advantageous over testing with the passive upper tester in that the former can employ a simplified test management protocol to be used in the process of testing to have the behavior of the upper and lower testers coordinated.

The passive upper tester is simple in structure because it has no control information for its own testing behavior. Since, however, the need for controlling the behavior of the upper tester involves a complex test management control, increasing the traffic of test management data. This inevitably allows more errors to erase in the data handled, lowering the reliability of testing. The increased complexity of the test management protocol also results in a lowered degree of freedom of testing. By contrast, while the active upper tester becomes complex in structure because it controls its own testing behavior, its test management protocol can be made simple, improving the reliability and the degree of the freedom of testing. In the case of interconnectability testing, it is advisable to use the active upper tester to prevent workload from concentrating in the lower tester, because each IUT is assigned an upper tester.

2.5 Executable test case

A test suite in which test cases are described is prepared for each protocol specification. The test suit is made up of several test cases. ISO9646 stipulates that TTCN be used to describe a test case in an abstract manner. The abstract test suite is not intended to actually cause a testing system to perform testing behavior. Instead, it is used to make a documentation of test specifications. To do testing, it is necessary to realize the testing behavior described in the abstract test suite based on the specification of each testing system.

We decided to perform the testing by using the lower tester and upper tester to process each executable test case (ETC: Executable Test Case), which is a file describing the testing behavior. In our executable test case, testing behavior is expressed as a state transition table. One method to describe a testing procedure in an executable test case is describing the contents to be controlled and observed at each PCO *collectively* for processing in the sequence of description. By contrast, our executable test case describes the procedure for exchanging service primitives at the upper PCOs, the procedure for exchanging PDUs according to the protocol and the procedure for exchanging service primitives at the lower PCOs *individually* to allow controlling and observing operations to take place in parallel at each PCO. Figure 8 shows the structure of our executable test case. A test case of this structure allow the active upper tester and lower tester to work in parallel. Besides, it has the following two characteristics:

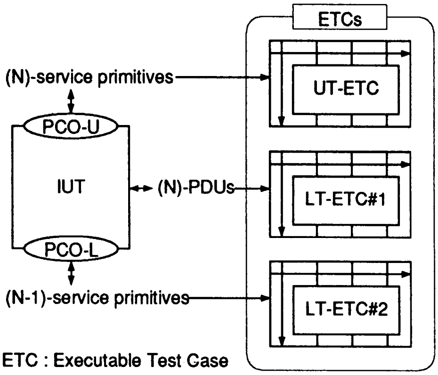


Figure 8 : Structure of our executable test case

(1) Simple description: This executable test case can simplify the contents of description. To describe all testing operations at individual PCOs collectively by relating them to each other, it is necessary to take into consideration all the combinations of testing behaviors at each PCO. This involves increasing the number of testing steps, inevitably resulting in an increased complexity of the contents of the test case. Figure 9 shows the flow of testing the establishment of transport connection for IUTs that implement the OSI transport layer protocol. To convert this

flow into an executable test case, describing the procedure collectively requires including all the transitions. But if the procedure is described step by step as with the structure of our executable test case, it is unnecessary to pay attention to the transition of any combination. This reduces the number of test steps described in the test case and thereby simplifying its structure, leading to improved processing efficiency, maintenance, understanding and reliability.

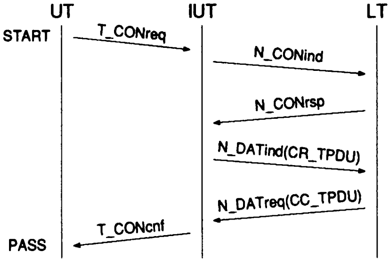


Figure 9 : An example of OSI TP0 protocol

(2) Extensibility: Since the executable test case is simple in contents, it can be easily modified for adaptation to other test case, it is also possible to use an executable test case for an upper tester used with a certain test item to another test item. For example, when testing a feature that shows the same test behavior at the upper PCO but works differently at the protocol level, it is possible to use different executable test cases for the upper and lower testers.

To describe specific testing behaviors to make testing executable, it is possible to describe the necessary procedure sequentially by using the C language, for example. But we decided to describe the test procedure by means of a state transition table by taking into consideration the expression of test events that may arise in parallel, the structure of the test case processing system, and ease of maintenance. Figure 10 shows part of our executable test case. The executable test case is made up chiefly of a 'current state' column, an 'action' column to be executed at that state, and an 'event' column. Since this executable test case was prepared according to relatively simple rules, we think it is relatively easy even to make a large number of test cases required to make a variety of tests. The processing system, an interpreter, was realized by means of compact, machine-independent functions.

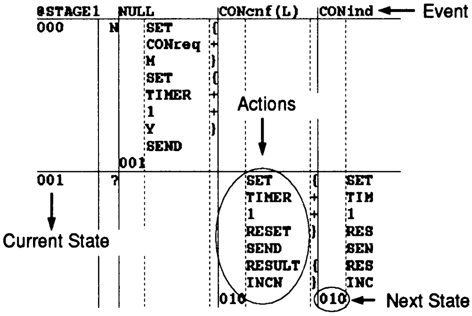


Figure 10 : Form of our executable test case

3.Interconnectability Testing System

3.1 AICTS

On the basis of the above discussion, we developed an interconnectability testing system named AICTS (AIC's Interconnectability Testing System), the structure of which is shown in Figure 11. The AICTS consists of a test manager (TM) responsible for controlling the behaviors of the system as a whole including SUTs, lower testers and SUTs. Each SUT comprises an IUT and an upper tester, both of which function in a coordinated manner. The test manager, which includes two lower testers, and the SUTs are implemented in a remote environment and connected through services provided by the lower layer. In this structure, channels of test management are established between the lower and upper testers and are used for their coordinated working. The channels under test are independent of the channels under test and the upper testers are active. This allows the AICTS to deliver the following testing functions:

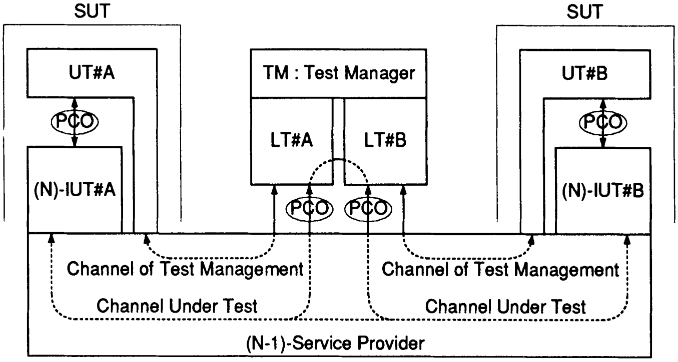


Figure 11 : Structure of AICTS

(1) Conformance testing function: This structure uses an upper tester and lower tester for each IUT. This is , in effect, the same as the structure of the typical conformance testing system. It is, therefore, possible to conduct conformance testing by using only the part of the necessary functions while not using the rest as shown in Figure 12.

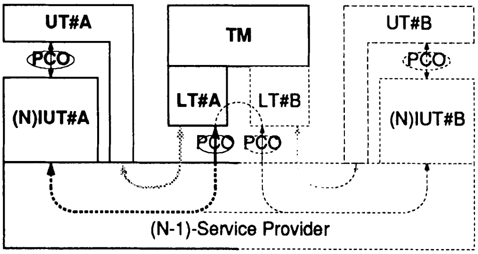


Figure 12 : AICTS - Conformance testing

(2) Extended interoperability testing function: In this structure, as shown in Figure 13, a connection under test is established between the IUTs, making it possible to do testing through control and observation at the upper PCOs without using the lower PCOs. The AICTS is capable of performing control and observation in the highest layer between mutually connected OSI products as with ordinary interoperability testing. It is also possible to perform an extended version of the conventional interoperability testing focusing on an intermediate single layer or consecutive layers. This testing is characteristic in that since it addresses only the services provided by the IUTs for the upper layers, control and observation are done only at the upper PCOs of the IUTs. This makes it possible to make a test environment most approximate to the actual operating environment of the IUTs.

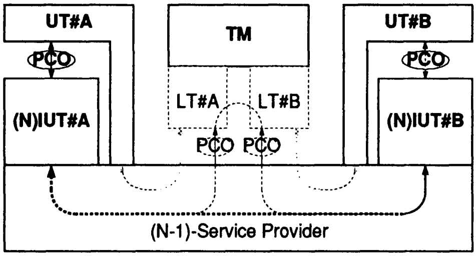


Figure 13 : AICTS - Extended interoperability testing

(3) Interconnectability testing function: This structure, as shown in Figure 14, allows interconnectability testing to be done by establishing a connection under test through the lower testers and providing control and observation at the upper and lower PCOs. In this testing, even if the IUTs fail to establish interconnection, it is possible to identify the cause because both upper and lower PCO are employed. With this testing, it is also possible to make the lower tester simulate errors of the lower layer and thereby observe the responses of both IUTs.

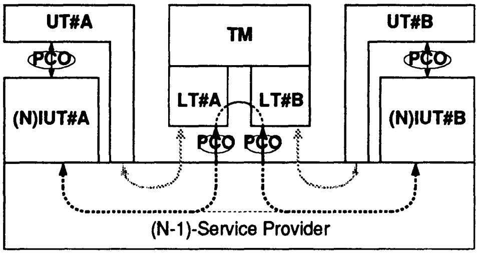


Figure 14 : AICTS - Interconnectability testing

3.2 Testing

We think that it is possible to effectively check the interconnectability of single-layer IUTs by conducting conformance testing, extended interoperability testing and interconnectability testing by means of the AICTS testing functions according to the procedure shown in Figure 15 as described below:

Step 1 - conformance testing: This testing, using the configuration shown in Figure 12, checks IUT to be mutually connected for their conformance. This step is to check each IUT to see if its behavior conforms with the OSI standards.

Step 2 - interconnectability function check: This testing, using the configuration shown in Figure 13, performs extended interoperability testing for IUTs to be mutually connected. This step is intended only to see if both IUTs can respond normally to stimulus received from the upper PCOs and establish interconnection between them.

Step 3 - interconnectability function analysis: This step is executed if any abnormal condition has been found in step 2 to identify the cause by using both upper and lower PCOs. In this step, interconnectability testing is done for the mutually connected IUTs by using the configuration shown in Figure 14. If the cause of the abnormal condition has been identified and has been fixed in this step, Step 2 is done again for confirmation.

Step 4 - abnormal environment testing: configuration shown in Figure 14, checks IUTs to be mutually connected. In this testing, the lower testers generate errors and other abnormal states of the lower layer on purpose to examine the responses of the IUTs. If their interworking goes wrong, checking is done to see which IUT is responsible for the malfunctioning. After elimination of the cause, the testing is conducted again for confirmation.

Step 5 - interoperability testing: This testing, done with the configuration shown in Figure 13 or the usually employed interoperability testing configuration, is to perform the final checking of the interconnectability of the IUTs involved.

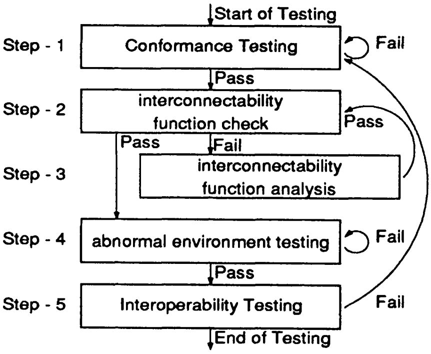


Figure 15 : AICTS - Steps of testing

3.3 System implementation

The AICTS interconnectability testing system incorporating the functions discussed above in now used as a prototype for our development work involving the following conditions:
- Layer under test: OSI transport layer
- Protocol under test: Transport protocol Class 0, 2
- Lower layer in test environment: Network layer with X.25 protocol
- IUT interface: IUT upper and lower interface specification

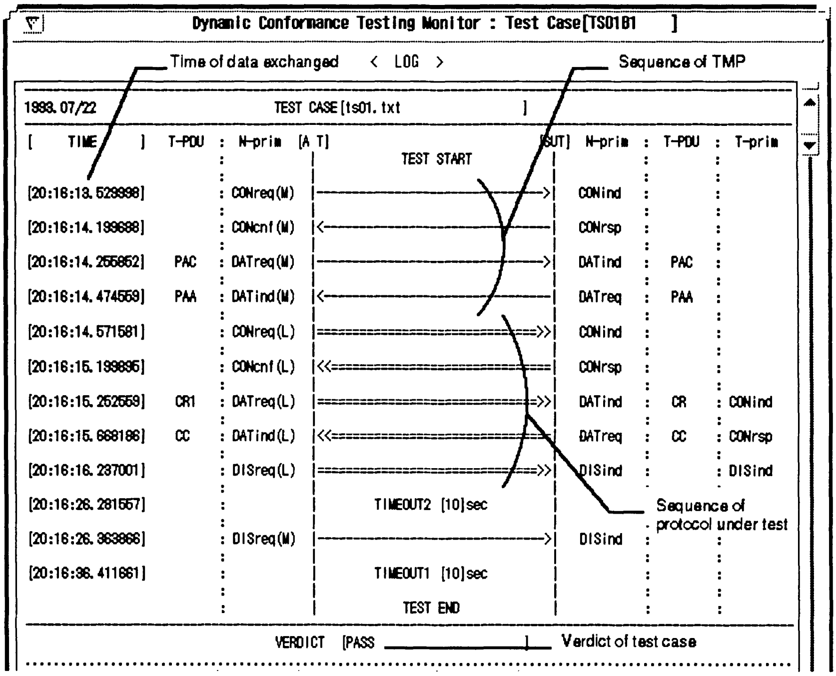
We have also decided to develop a system that is capable of:
- establishing a consistent testing environment including the processes of making preparations for and executing testing and preparing a report;
- establishing static and dynamic testing functions;
- establishing an integrating testing environment for conformance testing, interconnectability testing and extended interoperability testing; and
- providing an effective user interface for testing.

The following are the system development steps we employed:
(1) Basis system: This system was developed to deliver the conformance testing functions necessary to develop the parts making up the AICTS system and check the function of each function component.
(2) Extended interoperability testing system: This system was developed by extending the basis system for the purpose of delivering functions to check the interconnectability of IUTs.
(3) Interconnectability testing system: This system, to be developed by expanding the basis system and also the extended interoperability system, is designed to deliver functions to analyze the detailed behaviors of the IUTs and simulate any abnormal; environmental condition by using both upper and lower PCOs.

So far, we have developed the testing functions for the basis system and the extended interoperability testing system. The following are examples of execution of these systems:

3.4 Execution of testing

ISO9646 stipulates that in conformance testing, both static testing and dynamic testing be done in that sequence: the former is to review the implementation functions on paper and the latter to perform actual data transmission. With the AICTS, interconnectability testing and extended interoperability testing are done in the same procedure as with conformance testing. Figure 16 shows a window displayed when dynamic conformance testing is done with the basis system. In this window, the PDUs, service primitives and results exchanged between the test system and SUT during the testing are displayed and recorded. The information facilitates test result analysis and report preparation. In our testing, we used our internally developed IUT with test system debugging capability. This IUT can be set to simulate abnormal, as well as normal, behaviors. Our testing of the IUT for various behaviors with the AICTS proved that the AICTS can make verdicts properly for passes and failures for both normal and abnormal IUTs. In our testing, we executed a test suite of 40 cases, whose contents are equivalent with those employed by OSTC (Open Systems Testing Consortium)[8] and found that all the results were passed. Figure 17 is the structure of the conformance testing suite with the AICTS. Table 2 gives the results of the execution of this suite.



```
1993.07/22                     TEST CASE[ts01.txt            ]
[    TIME      ]  T-PDU : N-prim  [A T]                        [SUT] N-prim  : T-PDU : T-prim
                        :             |        TEST START          |         :       :
                        :             |                            |         :       :
[20:16:13.529998]       : CONreq(M)   |--------------------------->|  CONind :       :
                        :             |                            |         :       :
[20:16:14.199688]       : CONcnf(M)   |<---------------------------|  CONrsp :       :
                        :             |                            |         :       :
[20:16:14.255952]  PAC  : DATreq(M)   |--------------------------->|  DATind :  PAC  :
                        :             |                            |         :       :
[20:16:14.474559]  PAA  : DATind(M)   |<---------------------------|  DATreq :  PAA  :
                        :             |                            |         :       :
[20:16:14.571581]       : CONreq(L)   |==========================>>|  CONind :       :
                        :             |                            |         :       :
[20:16:15.199895]       : CONcnf(L)   |<<==========================|  CONrsp :       :
                        :             |                            |         :       :
[20:16:15.252559]  CR1  : DATreq(L)   |==========================>>|  DATind :  CR   : CONind
                        :             |                            |         :       :
[20:16:15.668186]  CC   : DATind(L)   |<<==========================|  DATreq :  CC   : CONrsp
                        :             |                            |         :       :
[20:16:16.237001]       : DISreq(L)   |==========================>>|  DISind :       : DISind
                        :             |                            |         :       :
[20:16:26.281567]       :             |     TIMEOUT2  [10]sec      |         :       :
                        :             |                            |         :       :
[20:16:26.363866]       : DISreq(M)   |--------------------------->|  DISind :       :
                        :             |                            |         :       :
[20:16:36.411661]       :             |     TIMEOUT1  [10]sec      |         :       :
                        :             |                            |         :       :
                        :             |         TEST END           |         :       :
                           VERDICT  [PASS                ]
```

Figure 16 : Executing of conformance testing

Table 2 : AICTS - Result of conformance testing

Total Time of Testing		1255 sec.
Average Time to Process an ETC		31 sec.
Maximum Time to Process an ETC		92 sec.
Minimum Time to Process an ETC		17 sec.
Maximum Line Number of an ETC		589 lines
Minimum Line Number of an ETC		117 lines
Average Line Number of an ETC		202 lines
Result of Testing	Pass	40 cases
	Fail	0 case
	Inconclusive	0 case
	Abort	0 case

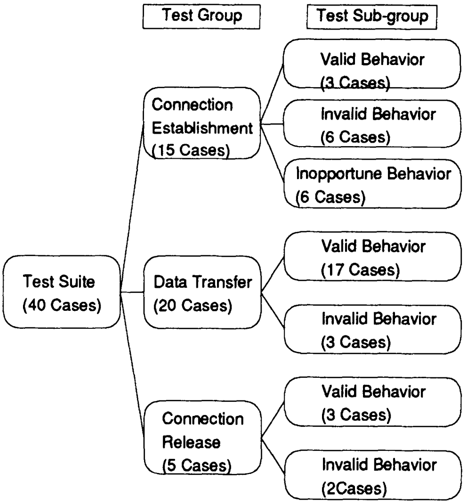
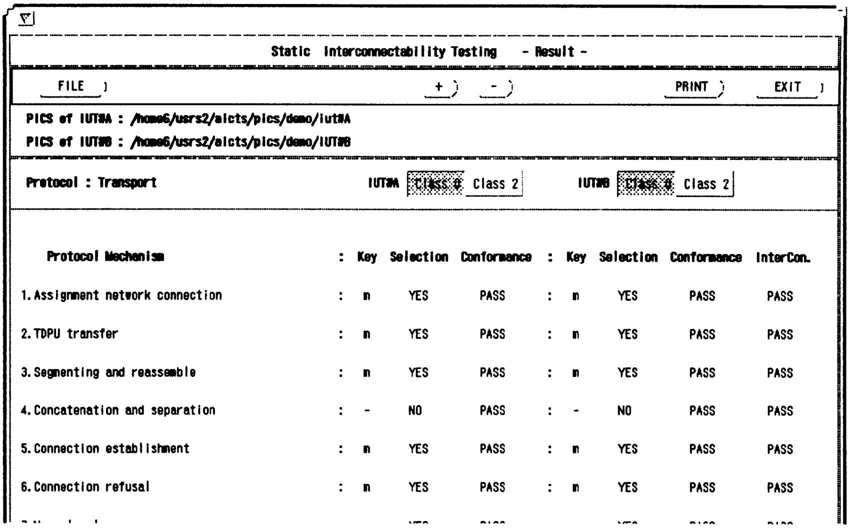


Figure 17 : AICTS - Structure of the conformance test suite



Static Interconnectability Testing - Result -

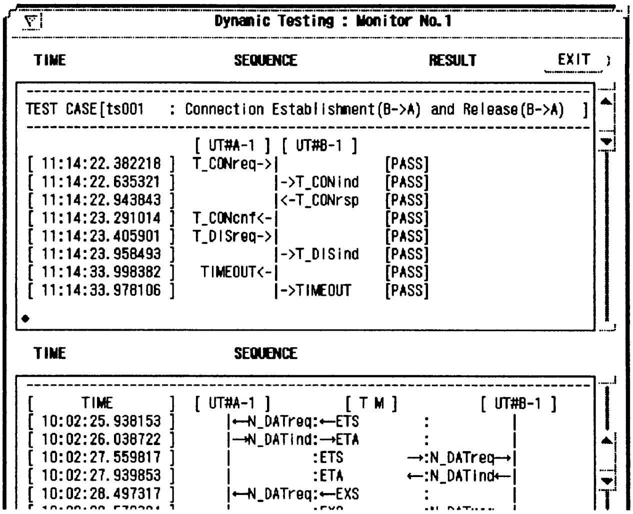
PICS of IUT#A : /home6/usrs2/aicts/pics/demo/iut#A
PICS of IUT#B : /home6/usrs2/aicts/pics/demo/IUT#B

Protocol : Transport   IUT#A Class 0 Class 2   IUT#B Class 0 Class 2

Protocol Mechanism	:	Key	Selection	Conformance	:	Key	Selection	Conformance	InterCon.
1. Assignment network connection	:	m	YES	PASS	:	m	YES	PASS	PASS
2. TDPU transfer	:	m	YES	PASS	:	m	YES	PASS	PASS
3. Segmenting and reassemble	:	m	YES	PASS	:	m	YES	PASS	PASS
4. Concatenation and separation	:	-	NO	PASS	:	-	NO	PASS	PASS
5. Connection establishment	:	m	YES	PASS	:	m	YES	PASS	PASS
6. Connection refusal	:	m	YES	PASS	:	m	YES	PASS	PASS

Figure 18 - AICTS - Static interconnectability testing

Figure 18 shows an example of static interconnectability testing with the extended interoperability testing system to compare the implementation functions of the IUTs to be connected with each other. The window automatically displays the status of the implementation functions of each IUT and the result of their comparison. Figure 19 illustrates a sample window displayed



Dynamic Testing : Monitor No.1

```
TIME                    SEQUENCE                    RESULT

-----------------------------------------------------------------------
TEST CASE[ts001   : Connection Establishment(B->A) and Release(B->A)  ]
-----------------------------------------------------------------------
                     [ UT#A-1 ] [ UT#B-1 ]
[ 11:14:22.382218 ]  T_CONreq->|                 [PASS]
[ 11:14:22.635321 ]            |->T_CONind       [PASS]
[ 11:14:22.943843 ]            |<-T_CONrsp       [PASS]
[ 11:14:23.291014 ]  T_CONcnf<-|                 [PASS]
[ 11:14:23.405901 ]  T_DISreq->|                 [PASS]
[ 11:14:23.958493 ]            |->T_DISind       [PASS]
[ 11:14:33.998382 ]   TIMEOUT<-|                 [PASS]
[ 11:14:33.978106 ]            |->TIMEOUT        [PASS]
◆

TIME                    SEQUENCE

-----------------------------------------------------------------------
[      TIME       ]  [ UT#A-1 ]          [ T M ]          [ UT#B-1 ]
[ 10:02:25.938153 ]    |←N_DATreq:←ETS      :                |
[ 10:02:26.038722 ]    |→N_DATind:→ETA      :                |
[ 10:02:27.559817 ]    |          :ETS     →:N_DATreq→       |
[ 10:02:27.939853 ]    |          :ETA     ←:N_DATind←       |
[ 10:02:28.497317 ]    |←N_DATreq:←EXS      :                |
```

Figure 19 : AICTS - Dynamic interconnectability testing

when interconnectability function checking was done for the IUTs by means of the extended interoperability testing system. The window displays and records the transport service primitives and TM-PDUs exchanged between the upper testers and the IUTs in a time series manner. The window also displays test results every test case.

3.5 Execution environment

Our hardware environment was designed for remote IUT testing and for testing in the same premises with the possibility of debugging in mind. Figure 20 shows a sample hardware configuration for AICTS testing. TM that includes two lower testers and SUTs are installed on different workstations. These workstations are connected by generally available communications software. The network configuration for remote IUT testing is made up of NTT's DDXP (Digital Data Exchange Network-Packet) and DSUs (Digital Service Units), the network for testing IUTs in the same premises is made up of pseudo ISDN switchboards and terminal adapters.

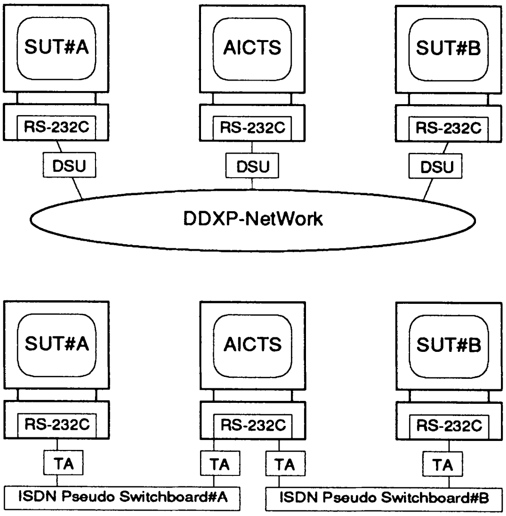


Figure 20 : Testing environment

4. Summary and Future Plans

In this paper we discussed our interconnectability testing architecture we developed by extending the existing testing method by solving problems with the general conformance testing and interoperability testing. We completed some of the functions of the AICTS testing system, which is capable of executing conformance testing, extended interoperability testing and interconnectability testing. We explained the configuration of our testing system and presented sample executions done with our testing system. Our future plans include:

(1) Developing test cases: We will further develop test cases and improve our testing environment. Our test cases were made manually based on our experience. It is difficult to

logically prove the validity of each of these test cases. To meet this, we are doing research on automatic test case generation from protocol specifications. So far, we have developed a prototype automatic test case generation tool. We will make efforts to complete this tool and demonstrate that test cases generated by using this tool include part of our current test cases. This will prove the validity of the tool itself and of the test cases used so far.

(2) Development of interconnectability testing functions: We have so far completed the functions necessary to realize extended interoperability testing. On the basis of these functions, we will further develop a testing system capable of conducting interconnectability testing. This system will also provide an environment for conformance testing through combinations of various functions.

(3) Adaptability to upper layers: Our testing system was developed to deal with OSI transport layer protocols. We will examine system configurations capable of testing with the upper three layers of OSI by applying the architecture of this testing system.

Acknowledgments

Our thanks go to Professor S. Noguchi, Nihon University, and Professor N. Shiratori, Tohoku University for their valuable advice given in the course of our research, to Mr. Ogata, a director, and Mr. Kobayashi, a chief researcher. AIC, for giving the opportunity of conducting this work, and to staff members of the Sendai Branch of Fujitsu BSC for providing cooperation in program development.

Reference

[1] ISO/IEC DIS 7498, “Information Technology - Open Systems Interconnection Reference Model - Part 1: Basic Reference Model,” (1992).
[2] ISO/IEC 9646 “Information Technology - Open System Interconnection - Conformance Testing methodology and Framework - Parts 1-5,” (1991).
[3] D.Rayner, “OSI Conformance Testing,” Computer Networks and ISDN Systems ,Vol.14,pp.79-98,North-Holland(1987).
[4] Jay Gadre,Chris Rohrer, Catherine Summers and Susan Symington, “A COS Study of OSI Interoperability,” Computer Standards & Interface,Vol.9,pp.217-237,North-Holland(1990).
[5] L.Lenzini and F.Zoccolini, “Interoperability tests on OSI products in the framework of the OSIRIDE-Interest initiative, “ Computer Networks and ISDN Systems, Vol.24, pp. 65-79, North-Holland(1992).
[6] S.T.Chanson,B.P.Lee,N.J.Parakh and H.X.Zeng,“Design and Implementation of a Ferry Clip Test System,” Protocol Specification,Testing and Verification, IX, North-Holland(1990).
[7] O.Rafiq and R.Castanet,“From conformance testing to interoperability testing,” Protocol Test Systems III,North-Holland(1991).
[8] OSTC TRANSPORT & SESSION TECHNICAL AREA TEST SPECIFICATION COMMITTEE, "OSTC ABSTARCT TEST SUITE TRANSPORT CLASS 0" (Jan. 1991).