프로토콜 적합성 시험 언어 TTCN

하 수 철
대전대학교 공과대학 컴퓨터공학과

요약

ISO 9646에서 정의된 TTCN(Tree and Tabula Combined Notation)은 프로토콜에 대한 규격 준수 여부, 즉 적합성을 확인하기 위해 사용되는 표기법으로서, 트리 구조와 테이블 형식을 사용하여 시험을 기술한다. 적합성 시험은 구현물의 정적 능력과 동적 행위를 검사하는 것이다. 이것은 구현 제품 공급자에 의해 만들어진 주장을 검증하는 역할을 한다. 이러한 주장은 프로토콜 표준 또는 기능적 프로파일을 참조하여 판단할 수 있다. 본 논문에서는 프로토콜의 적합성 시험 기술 언어 TTCN와 그의 확장에 대해 기본 개념을 논한다.

1. 서 론

TTCN(Tree and Tabular Combined Notation)은 ISO에서 개발한 표기법으로 프로토콜 적합성 시험(conformance test)을 위한 표준화된 시험 사례(test case)작성을 위해 고안되었으며[1], TTCN이라는 이름은 트리형 구조를 사용하여 시험을 작성하고 이 시험을 테이블 형식안에 기술하기 위한 표기법에서 유래되었다.

ITU-T 및 ISO/IEC 9646의 권고 안을 기준으로 한 TTCN의 사용은 적합성 시험 환경에 매우 중요하다. TTCN이 어떠한 프로토콜과 계층에도 적용되는 포괄적인 표기법이기 때문에 시험도구가 단일 시험 규격(test suite)이상을 처리하도록 작성될 수 있다. 이것은 시험 제공자에게 비용절감 효과를 줄 수 있는데, 프로토콜 구현자가 시험결과의 향상된 비교가능성으로부터 이득을 얻을 수 있다는 의미에서 다른 구현 가능성을 감소시키는 것이다.

일반적으로 무한개의 사건이 PCO(Point of Observation and Control)에서 발생할 수 있기 때문에 소모적인 적합성 시험 대신에 시험의 적용범위와 실행 비용간의 절충이 필요하다. 비 적합 구현물은 주어진 환경에서 적합성 시험을 통과할 수 있지만 다른 환경에서는 실패할 수 있다. 적합성 시험을 다음과 같이 구분한다.

(1) 기본 상호접속 시험(interconnection test): 첫적한 시험을 수행하지 않고 상호접속에 대한 충분한 적합성을 달성하기 위한 주요 특징의 제한적인 시험이다. 이러한 시험은 구현물의 공급자에 의한 적합성 주장의 기본으로 사용될 수 없다.

(2) 가용 시험(capability test): 정적 적합성 요구사항의 각각에 대한 시험이다. IUT (Implementation Under Test)와 PICS(Protocol Implementation Conformance Statement)의 일관성을 철저히 대조한다.

(3) 행위 시험(behavior test): 가능한 한 동적 적합성 요구사항의 전 범위에 대해 이 체력 있는 시험으로 시험기에 의해 전송된 유효 무효 부적당한 PDUs(Protocol Data Unit)에 반응하여 IUT에 의한 행위를 시험한다.

2. 적합성 시험과 TTCN

2.1 추상 시험 규격과 TTCN

추상 시험 규격(ATS: Abstract Test Suite)는 각 시험 항목이 식별된 목적을 가진 다수의 시험 항목으로 이루어진다. 시험기와 SUT(System Under Test)에 에워 쌓인 환경을 고려하지 않기 때문에 추상적이다. 따라서 시험 운용이 시작되기 전에 프로그래밍 환경을 고려한 ETS(Executable Test Suite)로 전환되어야 한다.

ATS의 설계는 PCOs(Point of Control and Observation) 위치와 선택된 ATM(Abstract Test Method)에 의존한다. 일반적으로 각 적용 가능한 ATM에 대해 하나 이상의 ATS가 존재한다.

모든 시험 항목은 시험 목적(purpose)을 가진다. ATS의 시험 항목은 시험 그룹으로 집단화 될 수 있다. 필요시 시험 그룹은 시험 그룹 내에 내포될 수도 있다. 시험 항목을 그룹화 하는 기준은 다음과 같다[2].

- 능력 시험은 의무적 특성, 선택적 특성을 기준으로 구룹화 될 수 있다.
- 행위 시험은 유효한 행위의 시험, 무효 행위의 시험, 부적당 행위의 시험으로 구룹화 될 수 있다.
- 행위 서브 그룹은 접속 설치 단계, 데이터 전송단계, 접속 해제 단계로 세분화 될 수 있다.
- 데이터 전송 단계는 IUT에 전송된 PDUs와 IUT로부터 수신된 PDUs에 초점을 두고 그룹화 될 수 있다.

ATS의 추상적 성격 때문에 시험 몸체는 형식 표기법으로 기술되어야 한다. ISO/IEC 9646-3은 이 목적으로 TTCN을 정의하고 있다. TTCN은 PDUs 교환하기와 관련된 행위 및 동작의 트리를 기술하기 위한 명세 언어이다.

2.2 TTCN의 정의

TTCN은 다수의 보충적인 방식으로 기술된다. 표준의 몸체는 예제에 의해 확장되는 문장형태의 기술로 이루어진다. 문서의 규범적인 부록은 TTCN에 대한 BNF(Backus-Naur Form)을 갖고 있으며, 별개의 규범적인 부록은 TTCN의 연산적인 의미론 정의를 포함하고 있다.

TTCN 시험은 “TTCN.GR"(Graphical)이라 부르는 페이퍼 버젼, 또는 "TTCN.MP"(Machine Processable)이라 부르는 기계 판독 버전 중의 하나로 작성될

수 있다.

TTCN에 대한 BNF는 LALR(1) 파싱도구에 의해 처리되도록 설계되었으며, TTCN 문법의 BNF 기술과 TTCN의 문장적 기술은 동일하다. 만일 텍스트가 BNF에 의해 지정된 것과 다르게 해석된다면, TTCN 문장에서 오류로 간주되며 표준 교정과정을 통해 교정되어야 한다. 이러한 과정이 완료될 때까지 BNF 명세는 문장보다 우선권(precedence)을 갖는다. 이것은 논리적인 우선 순위로서, TTCN 시험 사례의 처리를 구현하는 시험도구는 엄격하게 BNF를 구현하도록 요구되기 때문이다[3].

TTCN 시험 규격은 다음과 같은 4 부분으로 구성된다.

① 개요(overview)

시험 규격이 어떻게 다른 시험 사례들로 구성되는지가 표 형식(tabular form)으로 기술된다.

② 선언(declarations)

어떤 객체가 시험 사례 기술에서 발생하는지에 관하여 선언된 테이블을 포함한다.

선언을 위한 테이블로는

. 제어점과 관찰점
. 시험 규격 변수, 매개변수와 상수
. 시험 사례 변수
. 타이머

가 있다. 그리고 정의를 위한 테이블로는

. 메시지, 변수, 매개변수와 상수의 형
. 시험 규격 연산

등이 있다.

③ 제약조건(constraints)

시험기와 SUT 간에 교환되는 메시지들의 정확한 기술이 있는 테이블을 갖는다.

④ 행위(behaviors)

실제 시험 사례 기술 즉, 트리 구조와 관련된 판별을 포함한다. 트리 기술 역시 테이블로 주어진다.

TTCN의 BNF와 연산적 의미론 기술(operational semantic description)은 원래의 표기 명세 일부가 아니었다. TTCN 정의의 확장을 위해 몇 년이 지나는 동안 추가되었다. TTCN 연산적 의미론은 어떻게 TTCN 특징이 시험의 실행 중에 기능 하도록 의도되는지의 애매 모호한 해석을 제거하기 위한 시도로 제공된다. 의미론의 2가지 대안적인 형태는 Pascal 프로그래밍 언어와 유사한 의사 코드 버전(pseudo code)과 자연어 문장 버전으로 제공된다. 의미상 정보의 2가지 형태는 동일하다. 만일 그렇지 않다면 그것은 표준 교정 과정에 의해 교정되어야만 하는 오류이다. 의사코드에 의해 표현된 요구 사항이 이러한 오류가 교정될 때까지 우선권을 갖는다.

3. TTCN과 그의 확장

3.1 기본 개념

적합성 표준의 기본 전제는 IUT(Implementation Under Test)라 부르는 프로토콜의 구현이 블랙 박스라는 것이다.

IUT의 적합성에 관하여 그러 볼 수 있는 것은 IUT의 하위(lower) 및 상위(upper) 서비스 인터페이스에서 발생하는 사건을 관찰하고 제어함으로써 만들어진다. ISO/IEC 9646 항목에서 이러한 상호작용은 PCO(Points of Control Observation)에서 발생하며 ASP(Abstract Service Primitives)에 내장된 PDU(Protocol Data Unit)의 견지에서 표현된다.

IUT는 시험 시스템에 의해 시험된다. TTCN에서 시험 시스템의 다른 부분을 시험 컴퍼넌트(TC: test component)라 부른다.

하위 인터페이스에서 PCO들을 통해 IUT와 통신하는 시험 컴퍼넌트들을 묶어서 하위 시험기(LT)라 부르며, 상위 인터페이스에서 PCO를 통해 IUT와 통신하는 시험 컴프넌트들을 상위 시험기(UT)라 부른다. 적어도 하나의 시험 컴프넌트는 항상 시험 시스템에 나타나야 한다. 이것을 MTC(Main Test Component)라 부르는데, 시험을 조정하고 제어하는데 책임이 있으며 시험의 최종 판결을 내린다.

LT에서 시험 컴퍼넌트들 간의 통신은 CP(Coordination Points)를 통해 달성된다. 마찬가지로 UT 시험 컴퍼넌트는 CP를 통하여 각각 다른 것과 통신할 수 있다. LT와 UT간의 조정은 TCP(Test Coordination Procedures)에 의해 달성된다. 하위 시험기는 송신하고 수신하는 ASP들에 내장된 PDU의 제어와 관찰에 책임이 있어 2개의 컴퍼넌트 중에서 더 복잡하다. 실제로 LT는 시험 항목을 실행할 때 주어진 시간내에 적절한 프로토콜을 구현하고 있는 것이다.

IUT를 시험하기 위하여 시험 시스템을 제어하고 관찰하기를 바라는 상호작용의 연속순서(sequence) 또는 시험 사건(event)을 지정할 필요가 있다. 완전한 시험 목적(test purpose)을 지정하는 사건들의 연속 순서를 시험 항목(test case)이라 하며, 특정 프로토콜에 대한 시험 사례의 집합은 시험 규격(test suite)라 부른다.

시험 항목이 궁극적으로 운용될 실제 시험 시스템의 구조로부터 추상화되는 어떤 레벨에서 TTCN은 시험 항목의 명세를 위해 개발된 표기법인 것이다

ISO 적합성 표준은 시험이 (N-1)-layer ASP들, (N)-layer ASP들 및 (N)-layer PDU들에 따라 지정되도록 요구한다. 이러한 요구 사항을 만족시키기 위해 TTCN이 구비해야하는 최소의 기능은 다음과 같다.

- 시험 시스템에 의해 송신되거나 수신되는 ASP들을 지정하는 능력
- ASP들에 내장된 PDU들을 지정하는 능력

- 특정 PCO들에서 ASP들이 송신되거나 수신되는 순서의 명세 능력

기존의 TTCN 언어는 동적 행위 표(시험항목 단계)에서 병렬 행위 트리의 명세를 허용하지 않았다.

병렬 트리는 하나 이상의 병행접속 또는 관련성이 완전한 시험을 위해 요구되는 multi-party testing(MPyT)에서 요구된다. 비병렬인 경우는 single-party testing(SPyT)에서 사용되어 왔다.

다수 접속을 설정하는 간단한 경우, 그들의 각각이 데이터의 100 패킷을 전송하고 접속 해제하는 경우를 고려하자. 만일 다수의 병렬 시험 프로세스가 가능하다면 시험 항목의 명세는 훨씬 간단할 것이다. 시험 항목 트리의 별개 인스턴스는 각 접속에 대해 실행될 것이다. 이 경우 모든 요구된 사건이 수행되는 시간 또는 순서는 중요하지 않을 것이다. 예를 들어 첫 번째 접속이 100 패킷을 완성하는 두 번째보다 오래 걸린다면 상관없을 것이다. 각 접속은 그 자신의 시간 내에 완료하고 모든 시험으로부터의 결론은 완료 발생의 순서에 무관하게 모든 것이 완료된 후 분석이 가능하게 될 것이다.

다수의 병렬 시험 프로세스에 의한 시험은 이점이 있다. 기존의 SPyT에서 모든 사건은 단일 트리 자체 내부에서 순서화 되었다. 사건의 엄격한 순서가 중요하지 않거나 비결정적일 때 이러한 시험을 적절히 표현하는 방법은 병렬로 실행할 수 있는 다수의 별도 시험 프로세스(트리)를 통해서이다.

다른 다수의 경우 서로 다른 프로세스 사이에 어떤 사건의 순서는 개별 사건 레벨에서가 아니라 어떤 상위의 레벨에서 매우 중요하다. 예를 들어 어떤 포인트까지 독립적으로 실행될 단일 트리 내부의 사건 진행을 가질 수 있다. 이 포인트에서 첫 번째 트리의 실행이 계속되기 전에 다른 트리에서 사건이 다른 포인트에 도착하는 것을 보장할 필요가 있다. 이러한 경우 프로세스간 통신과 동기화 메커니즘이 필요하다. 이러한 메커니즘은 프로세스들(트리들)이 다른 것을 기다림을 보장하는데 필수적이며 선택된 점에서 상호 관심의 정보를 교환할 수 있다[4].

이러한 능력을 제공하기 위해 ISO9646의 part 3에서 병렬 트리와 연관된 특성, 병행으로 실행될 트리의 의미를 지정하는 방법을 효과적으로 정의하도록 제안되었다.

3.2 concurrent TTCN

통신 네트워크 분야에서 새로운 기술의 진화는 다양한 프로토콜과 다중 접속(multiple connection)을 포함하는 새로운 서비스의 정의를 유지 보수할 수 있게 만들고 있다. 이러한 프로토콜에 대한 시험환경은 더욱 복잡하여 기존의 OSI[ISO 9646-1]의 문맥에서 정의된 시험 방법을 벗어나고 있다. 이것은 TTCN의 확장인 병행(concurrent) TTCN의 정의를 제정하게 하였다.

TTCN과 달리 병행 TTCN은 시험 사례의 실행에 하나 이상의 활성화된 시험 컴퍼넌트가 참여할 수 있다. 모든 시험 컴퍼넌트는 병렬로 운용되고 조정 메시지(coordination message)를 교환함으로써 그들의 행위를 조정한다. 병행 TTCN의 잠재적인 이점은 복잡한 시험환경에 대한 시험 사례 기술이 훨씬 용이해 진다는데 있다. 반면 시험 항목의 유도(derivation)와 확인(validation)이 더욱 어려워진다는 단점이 있다.

ISO/IEC 9646 - 3의 TTCN 확장 개정안에 대한 주요 내용은 다음과 같다.

① 소개의 추가

추상 시험 항목의 명세에서 Multi-party 시험과 Single-party 시험 양자에 병행성(concurrency)을 사용하는 능력을 제공한다.

② 범위의 추가

- 하나 이상의 행위 기술이 병렬로 기동되는 시험 항목의 명세는 병행성을 고려하여 TTCN의 확장으로 다루어진다.
- TTCN에서의 병행성은 다음과 같은 시험 항목의 명세에 적용 가능하다.

a) Multi-party 시험 문맥에서
b) Single-party 또는 Multi-party 시험 문맥 중에서 Multiplexing과 Demultiplexing을 다루는 곳에서
c) Single-party 또는 Multi-party 시험 문맥에서 Splitting과 Recombining을 다루는 곳에서
d) IUT에 의해 조작되는 프로토콜 또는 프로토콜 집합의 복잡성이 병행성은 시험 항목의 명세를 단순화시킬 수 있는 그런 것 일 때의 single-party 시험문맥에서

시험기는 MTC(Main Test Component)와 0이상의 PTC(Parallel Test Component)로 구성된다 비병행성 TTCN에서 MTC를 선언할 필요가 없는 것은 단지 하나의 시험 컴퍼넌트만 존재하고 잠정치가 MTC이기 때문이다.

시험 컴퍼넌트는 시험 컴퍼넌트 선언표에서 선언된다. 시험 컴퍼넌트는 이 표에서 하나 또는 그 이상의 PCOs로 결박될 수 있다. 시험 컴퍼넌트는 조정점(Coordination Point)을 통해 조정 메시지를 교환함으로써 서로 교신할 수 있다. 시험 컴퍼넌트 구성 선언표는 시험 컴퍼넌트의 추상 구성을 상술하는데 사용된다. 이러한 선언은 시험 컴퍼넌트에 의해 사용되는 CPs가 어떤 것인지를 보여준다. 서로에 대한 CPs의 정확한 관계는 CP 선언에 포함된다.

조정메시지(Coordination Message)는 PDUs를 상술하는데 사용된 것과 유사한 방식으로 지정된다. ASN.1이 CM 명세를 위해 사용될 수 있다. CM 제약조건은 PDU 제약 조건과 매우 유사하다. 특별 Proformas가 CMs과 CM 제약조건의 정의/선언에 제공된다. 또, CMs과 CM 제약조건의 정의/선언에 제공된다. CMs은

정규적인 TTCN SEND와 RECEIVE문을 사용하여 전송 및 수신된다.

요약하면, 병행 TTCN은 다음과 같은 Proformas를 사용한다.

a) 시험 컴퍼넌트 선언(Test Component Declarations)
b) 시험 컴퍼넌트 구성 선언(Test Component Configuration Declarations)
c) 시험 컴퍼넌트 변수 선언(Test Component Variable Declarations)
d) CP 정의
e) CM형 정의
f) ASN.1 CM형 정의
g) CM 제약조건 선언
h) ASN.1 CM 제약조건 선언

4. 연산적 의미

4.1 기본 개념

TTCN 의미론의 정의는 3단계 접근을 따른다. 첫 단계에서 구체적인 TTCN 문장은 구문과 동적 의미가 대조된다. 이것은 두 번째 및 세 번째 단계 수행 동안에 조작될 수 있는 각 시험 항목에 대한 행위 트리로 귀결된다. 두 번째 및 세 번째 단계는 모두 한번에 대안들(alternatives)의 한 레벨씩 TTCN 시험 항목을 확장하고 해석하는 TTCN 기계의 기술로 간주될 수 있다.

첫 번째 단계는 다음의 두 단계로 분리될 수 있다.

a) 구문 정의
어떠한 의미론도 고정된 문법(context free)을 갖지 않는 언어에 할당될 수 없다.

b) 정적 의미론 정의
정적 의미론은 문맥 자유 문법에 의해 생성된 문장들이 의미가 있음을 기술한다. (예를 들면, 사용된 PCOs는 선언되어야 하고, 행위 부에서 참조된 제약 조건들은 시험 규격의 제약 조건부에 존재해야 한다 등이 그것이다.)

두 번째 및 세 번째 단계에서 추상 TTCN 기계가 정의된다.

이것은 기술된 구조로 기계가 구축될 것이라는 것과는 다르다. 구현 문제는 동적 또는 연산적 의미론의 정의에서 고려되지 않는다. TTCN 기계의 중심부분은 추상 평가 트리를 해석하는 EVALUATE_TEST_CASE라 부르는 프로세스이다. 추상 평가 트리는 이 프로세스의 매개변수이다.

두 번째 단계는 대안들의 집합이 완전히 처리되도록 하기 위해 REPEAT 구조, 잠정치 트리 및 Attach 구조를 확장함으로써 대안의 첫 번째 또는 현재 레벨을 확장한다.

세 번째 단계는 다음을 처리하기 위해 대안들의 후속 레벨을 발견하는 확장된 레벨을 평가한다.

TTCN에 대한 연산적 의미론은 개인적인 선호도에 기반 하여 방법의 선택을 독자에게 제공하기 위해 의사코드 및 자연어 2가지 선택적 표기법을 지원한다. 이들 표기법이 중복되는 곳에서 두 가지가 동일함을 알 수 있다. 만일 의사 코드와 자연어가 충돌되면, 이것은 오류이며 결함 보고서를 통해 표준화 기구에 다시 보고되어야 한다. 그렇지만 이러한 경우 의사 코드가 표준화 기구에 의해 교정중인 자연어 문장보다 우선적으로 처리된다.

4.2 세부 모델

TTCN 행위 트리는 한번에 대안들의 한 레벨이 평가된다. 각 레벨에서, REPEAT 구조는 확장되고, 잠정치는 첨부되며 Attach 및 RETURN 구문이 확장된다. 이것은 하나가 성공적으로 부합되는가를 발견하기 위해 평가될 대안 집합을 산출하며 이것으로 인해 다음에 진행될 대안 집합을 결정한다. 하나의 TTCN 문장에 대한 하나의 부합을 구성하는 것이 무엇인가에 대한 요구사항은 행위 라인에 코딩된 것과 이 의미론 문장에 기술된 것에 의존한다.

각 TC 실행의 연산적 의미론은 ISO/IEC 9646-3의 부록 B에 기술된 시험 항목 실행의 연산적 의미론과 동등하다. 다음이 추가된다.

a) 스냅샷은 TC의 CP큐에 정보를 유지할 수 있다.
b) CM에 대해 SEND 사건이 작동될 때, CM은 SEND 사건에서 지정된 CP의 큐에 놓이게 된다. 시험 항목 실행의 연산적 의미론은 TC들의 병렬 실행이다. TC들은 TC가 CP에 의해 접속되어 있는 모든 다른 TC에 만일 multiple TC들이 시험 항목에서 사용된다면, 타이머는 TC의 영역을 가진다. 즉, 각 TC는 선언된 타이머의 고유한 인스턴스를 가진다.
   모든 TC의 실행은 시험 판정이 MTC에 의해 할당될 때 정지된다.

TTCN 의미론은 TTCN 행위 트리와 그 트리에서 노드를 형성하는 컴퍼넌트들의 실행을 설명하는 단순한 기능적 접근을 사용하여 정의된다. 이 기능은 TTCN 의미론을 이해하기 위해 고안된 것이지 특정 실행 모델 또는 고급 프로그래밍 언어와 관련지으려 의도된 것은 아니다. 이것들은 TTCN 실행에 대한 직접적인 방법을 의미하지 않는다.

TTCN 시험 항목의 실행은 EVALUATE_TEST_CASE를 호출함으로 시작한다.

· procedure EVALUATE_TEST_CASE(TestCaseId BehaviourTree)

(* 레벨은 EVALUATE_TEST_CASE에 지역적인 변수이다.
만일 A가 현재의 행위 트리에서 m개의 대안들의 한 대안이라면 레벨은 각 A가 사건, 의사 사건 또는 구조 중의 하나인 순서집합(A_1 , A_2 ,…,Am)이다. 트리는 전체로 확장되기 때문에 나타날 수 있는 TTCN 구조는 GOTO와 ACTIVATE임에 유의한다.
TestCaseId, DefRefList는 TestCaseId에 의해 식별된 시험 항목의 헤더로부터 잠정치 참조를 추출한다.*)

```
begin
  Level :=FIRST_LEVEL (BehaviourTree);
  EVALUATE_LEVEL(TestcaseId, BehviourTree, Level, TestCaseId, DefRefList)
end
```

· procedure EVALUATE_LEVEL(TestCaseOrStep, Tree, Level, Defaults)

(* 이 함수는 트리의 대안들의 첫 번째 레벨이며, 평가되도록 유지된 TestCaseOrStep의 부분인 Level을 확장하고 평가한다. 또 Defaults는 잠정 트리 참조의 현재 활성 목록을 부여한다. Level에 포함된 대안 A_1 … Am이 발생된 순서대로 처리된다. TTCN 연산적 의미론은 대안 집합의 처리가 동시적이다. 즉, 모든 사건의 상태는 대조 프로세스 실행 중에는 변경할 수 없다. 레벨은 적합한 문장 함수에 의해 다음 레벨로 갱신된다 *)

```
begin
  EXPAND LEVEL(TestCaseOrStep, Tree, Level, Defaults);
  repeat
    TAKE_SNAPSHOT;
```

(* 진입하는 PCO와 CP 큐(들)의 스냅셧, 적절한 타임아웃 목록, 그리고 모든 시험 컴퍼넌트의 종료 상태가 택해진다. 스냅셧을 택하는 행동은 모든 PCO, CP 또는 타임아웃 목록으로부터 사건을 제거하지 않는다.*)

```
    if EVALUATE_EVENT_LINE(A1 , Level, Defaults) then
          EALUATE_LEVEL(TestCaseOrStep, Tree, Level, Defaults);
    if EVALUATE_EVENT_LINE(A2 , Level, Defaults)then
          EVALUATE_LEVEL(TestCaseOrStep, Tree, Level, Defaults);
    if      …
            …                                                    …

    if EVALUATE_EVENT_LINE(Am, Level, Defaults)then
          EVALUATE_LEVEL(TestCaseOrStep, Tree, Level, Defaults);
  until     SNAPSHOT_FIXED(LEVEL);
  stop      (TestCaseError)
```

(* 모든 적절한 PCO와 CP큐(들)이 그들에 대한 어떤 사건을 가지며, 모든 적절한 타이머가 소멸된다면 SNAPSHOT_FIXED(Level)은 TRUE를 반환하고, 그렇지 않으면 FALSE를 반환.*)

```
  end
```

5. 결론

시험 항목 설계, 명세, 시험 프로세스 구현을 단순화하기 위하여 병렬성이 요구된다는데 의심의 여지가 없다. 수백 개의 동시접속을 설정하는 경우, 이러한 시험 항목을 작성하는 것은 이전의 TTCN 구조로는 거의 불가능하다. 시험 규격의 크기와 작성의 지루함은 별개로 하더라도 다양한 접속간의 사건들 사이의 관계성의 비결정적 성격은 모든 가능한 조합을 기술함 없이 명세를 만드는 것이 불가능하다. 병렬성의 기본 원리는 다수의 실행트리를 독립적으로 설정하는 문제이기 때문에 상대적으로 간단하다.

또한, 이러한 트리는 독립적으로 실행되더라도 제어 메커니즘의 다양성에 의해 서로 관련될 수 있어 명쾌하다. 예를 들어 다른 트리의 하나로서의 시간이 다른 행동집합을 수행할 때까지 트리가 어떤 또 다른 행동의 집합을 수행하기를 바랄 수 있다. 그리고 그후 첫 번째 트리는 실행을 재개할 것이다. 그 결과 언제 무엇이 정확히 발생했는지를 보증할 필요가 있는 경우를 제외하고는 동기화 요구사항이 컴퓨터 시스템에서의 병렬처리 사용과 매우 유사하다.

그러나, 일련의 병렬트리를 가짐으로부터 유발되는 다음과 같은 문제들의 정교한 해결이 요구된다.

① 다수의 트리가 어떻게 한번에 시작되는가?
② 어떻게 시험 항목이 종료되는가? 즉, 언제 하나 또는 모든 트리가 종료되는가?
③ 판정의 다중성을 어떻게 단일 판정으로 전환하는가?
④ 어떻게 다중 트리를 제어하는가?
⑤ 어떻게 정보를 하나의 트리에서 다른 트리로 전달하는가?
⑥ 다중 트리가 정보를 공유할 수 있는가?
⑦ 어떻게 트리들이 요구된 상호 독립성의 집합에 따라 서로 동기화 되는가?

참고문헌

[1]. ISO/IEC, Information processing systems - OSI Conformance Testing Methodology and Framework, ISO/IEC 9646 1-7, 1994.

[2]. A.Tang, S.Scoggins, Open Networking with OSI, PTR Prentice-Hall, 1992.

[3]. 김기영, 하수철,“프로토콜 적합성 시험과 Concurrent TTCN 컴파일러”, 한국정보처리학회지 1996.9.

[4]. K. G. Knightson, OSI Protocol Conformance Testing: IS 9646 Explained, McGraw-Hill, 1993.