실제자원 시뮬레이터를 위한 E-GDMO 컴파일러 구현

*송병권, **김건웅, ***진명숙
*서경대학교 정보통신공학과,**목포해양대학교 해양전자통신공학부,***명지전문대학 정보통신과
bksong@skuniv.ac.kr, kgu@mail.mmu.ac.kr, msjin@mail.mjc.ac.kr

Implementation of Extended GDMO Compiler for Real Resource Simulator

*Byung-kwen Song, **Geonung Kim, ***Myung-sook Jin
*Dept. of Information & Comm. Eng., Seokyeong University, **Division of Comm. & Electronic Eng., Mokpo National Maritime University, ***Dept. of IT & Comm, Myongji College

요 약

본 논문에서는 실제 자원의 개발 전에도 망 관리 시스템의 개발 및 운용 테스트를 수행하도록 지원하는 실제 자원 시뮬레이터(RRS: Real Resource Simulator)를 위해 확장된 GDMO 문법과 이를 처리하는 E-GDMO 컴파일러를 소개한다. RRS에서는 사용자가 원하는 형태로 실제 자원의 동작을 시뮬레이션 해야 하므로, 기존 GDMO의 패키지 부분에 사용자가 동작 특성을 기술 할 수 있도록 문법을 확장하였다. 또한 E-GDMO 컴파일러는 기존의 GDMO 컴파일러 역할과 RRS의 구성 요소 중 사용자가 정의한 동작 특성을 유지하는 SDT(Simulation Data Table)의 내용을 초기화하는 역할을 동시에 수행한다.

1. 서론

TMN(Telecommunication Management Network)은 표준화된 개방형 방식의 총체적이고 일원화된 통신망 운용관리체제를 구축하기 위해 권고된 것으로 관리 정보 모델링을 위해 OSI(Open System Interconnection)에서 제안한 객체지향 모델링 기법을 채택하고 있고, 관리 정보의 접근 및 교환을 위하여 OSI의 CMIP(Common Management Information Protocol) /CMIS(Common Management Information Service) 프로토콜을 채택하고 있다[1][2][3][4][5][6][7][8].

TMN에서는 망 관리 역할에 따라 관리자(manager)와 에이전트(agent)로 구분하고 있다. 관리자는 에이전트에게 관리 요청을 전달하고 에이전트로부터 관리 요청에 대한 수행 결과를 반환 받거나, 특별한 사건이 발생했음을 알리는 통고(notification) 메시지를 수신한다. 또한 에이전트는 관리자로부터 관리요청을 수신한 다음 실제 자원(real resource)에 접근하여 해당 값을 가져온 후, 그것을 관리자에게 반환하거나 실제 자원에서 발생한 통고를 전달한다.

망 관리 시스템에서 실제 자원은 전기통신망을 구성하는 교환기 및 각종 유무선 통신 장비 등 실제 관리하고자 하는 하드웨어 또는 소프트웨어적인 요소들이다. 일반적으로 실제 자원을 관리하기 위한 망 관리 시스템은 실제 자원이 완성된 후에 개발을 시작하거나, 실제 자원 개발자로부터 사전에 관리 정보를 제공받아 실제 자원 개발과 병행해서 개발하는 방식을 취하는데, 두 방식 모두 비용과 시간이 많이 소요된다.

RRS(Real Resource Simulator)는 실제 자원 개발 전에도 사용자가 지정하는 형태로 속성(attribute) 값들이나 사건(event)을 생성하여 에이전트에게 돌려주고, 또한 실제 자원 없이도 운용 환경을 꾸밀 수 있도록 제안되었다[9][10]. 따라서 이를 도입하면 실제 자원과 망 관리시스템의 개발을 병행할 수 있으며, 또한 가상의 망 환경에서 망 관리시스템을 평가할 수도 있다. 본 논문은 이러한 RRS를 위해 제안되었던 확장된 GDMO (Guidelines for the Definition of Managed Object) 문법과 이를 처리하기 위한 E-GDMO 컴파일러[11]를 구현한 결과를 소개한다. RRS의 구현 결과는 다른 논문에서 보인다.

본 논문에서는 먼저 2장에서는 RRS 구성과 E-GDMO의 역할을 설명하고, 3장에서는 E-GDMO 문법을 소개한다. 다음 4장에서는 E-GDMO 컴파일러의 구현 결과를 보이고, 5장에서 결론을 맺는다.

2. RRS의 구성과 E-GDMO의 역할

RRS는 그림 1과 같이, 에이전트와의 다양한 통신 방식을 지원할 통신 모듈, 각 관리 객체 별로 사용자가 지정한 속성 값 및 통고의 생성 방식 등을 담고 있는 SDT(Simulation Data Table), 현재 생성된 관리 객체들의 정보를 담고 있는 MOT(Managed Object Table), 이들을 바탕으로 속성 값을 변화시키거나 통고를 발생시킬 커널 메인(kernel main)과 이때 이용할 지원 함수(support function)코드와 스케줄링 테이블, SDT 또는 객체 테이블 내의 값 변경, 통고의 발생 등을 직접 수행할 수 있도록 하는 GUI 인터페이스로 이루어져 있다.

여기서 커널 메인은 ① 에이전트로부터 요청 받은 속성 값을 MOT에서 찾아 반환하고, ② 사용자가 지정한 자원함수에 따라 사건을 발생시키고, 이에 대한 통고 메시지를 생성, 에이전트에게 전달하는 기능을 수행한다. 따라서 커널 메인은 RRS내의 모든 자원들을 관리하게 되는데, 이에 관련된 관리 객체의 정보들이 MOT에 존재하게 되고, 각 자원마다 사용자가 지정한 시뮬레이션 동작 특성이 SDT에 존재하게 된다.

지원함수에는 커널 메인에서 이용할 랜덤 값을 생성하는 함수들이 존재한다. 이러한 지원함수는 크게 두가지 용도로 쓰이게 되는데, 상태 값 자체의 생성과 사건이 일어나는 시간 간격 결정에 이용된다.

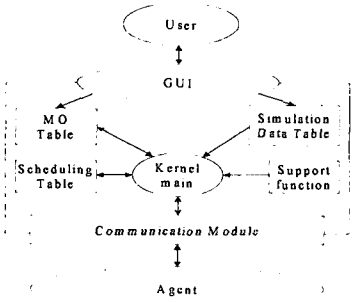

그림 1. RRS 전체 시스템 구조

SDT에는 각 관리 객체의 속성 값 및 통고들을 생성할 때 이용할 정보들을 담고 있다. 관리 객체마다 여러 속성과 사건 발생이 가능하므로, 관리 객체 식별자와 속성 식별자, 사건 식별자를 이용, 구분하도록 하고, 랜덤 값을 생성하는데 이용할 지원함수, 사건 발생 간격을 결정할 지원 함수, 이들 지원 함수들의 관련 매개 변수들을 담고 있다.

MOT에는 실제 자원의 현재 값들이 저장된다. 실제 자원 상태는 바로 전 상태에 종속적인 것과 독립적인 것이 있을 수 있는데, 종속적인 경우 바로 전 상태 값을 바탕으로 새로운 상태 값이 생성되어야 한다. 따라서 이를 위해 현재 상태 값들을 저장할 MOT가 있다. 또한 GUI를 통해 커널 동작 전에 SDT의 값들을 초기화하거나, 동작 중에도 사용자의 뜻에 따라 MOT에 저장된 정보들을 변경할 수 있도록 되어 있다.

사용자는 SDT 정보들을 생성하기 위해서 기존의 관리 객체 정의와 별개로 각 관리 객체의 동작을 기술할 수 있어야 한다. 이러한 SDT 정보를 기술하고 처리하는 방안은 여러 가지가 있는데, 그 중 하나는 기존의 GDMO 문법을 확장하여 이를 기술하고, 또한 기존 GDMO 컴파일러를 확장시켜 처리하는 방법이다. 본 논문에서는 이러한 확장된 GDMO를 E-GDMO라 칭하고 이를 처리하는 확장된 GDMO 컴파일러를 E-GDMO 컴파일러라 칭한다.

3. E-GDMO

사용자는 표준 GDMO 문법에 시뮬레이션 신택스(syntax)을 포함한, 확장된 GDMO 문법을 이용하여 시뮬레이션하고자 하는 실제 자원의 행동을 표현할 수 있다. 원래 GDMO 문법은 관리되는 객체를 9개의 템플릿(template)으로 나누어 정의하도록 하였다. 이중에서 관리 객체 템플릿은 관리 객체의 구성 요소를 정의하며, 이때 정의되는 구성 요소는 부모 클래스(class)와 관리 객체에 포함되는 패키지(package)들이다. 패키지 템플릿은 관리 객체 템플릿에서 참조하는 패키지의 구성 요소들을 정의하고, 여기에는 속성과 속성 그룹(attribute group), 동작(action), 통고 등이 있다. 이외에 속성 템플릿, 속성 그룹 템플릿, 동작 템플릿, 통고 템플릿, 파라미터(parameter) 템플릿, 네임바인딩(namebinding) 템플릿, 행동(behaviour) 템플릿이 있다.

RRS에서는 수행할 시뮬레이션 대상을 관리 객체의 속성과 통고로 제한하였다. 따라서, E-GDMO에서는 관리 객체에 포함되는 속성과 통고를 시뮬레이션할 수 있도록, 관리 객체의 속성과 통고를 기술하는 패키지 템플릿에 시뮬레이션을 위한 문법을 추가하였다. 그림 2는 E-GDMO의 패키지 템플릿에 대한 문법을 나타낸다. 여기서 굵은 글자체로 나타낸 것이 새로 추가된 문법이다.

```
<package-label> PACKAGE
        [BEHAVIOUR <behaviour-definition-label>
                        [, <behaviour-definition-label>]* ; ]
        [ATTRIBUTES     <attribute-label> propertylist [<parameter-label>]*
                        [, <attribute-label> propertylist [<parameter-label>]* ]* ; ]
        [ATTRIBUTE GROUPS   <group-label> [<attribute-label>]*
                        [<group-label> [<attribute-label>]*]* ; ]
        [ACTIONS    <action-label> [<parameter-label>]* [, <action-label>
                        [<parameter-label>]*]* ; ]
        [NOTIFICATIONS
                        <notification-label> [period-definition <parameter-label>]*
                        [, <notification-label> [period-definition <parameter-label>]*]* ; ]
[REGISTERED AS object-identifier] ;
supporting productions
propertylist    →   [REPLACE-WITH-DEFAULT]
                    [DEFAULT VALUE      value-specifier]
                    [INITIAL VALUE      value-specifier]
                    [PERMITTED VALUES   type-reference]
                    [REQUIRED VALUES    type-reference]
                    [RANDOM VALUES          [random-value-specifier] ]
                    [get-replace]
                    [add-remove]
period-definition   →   RANDOM VALUES [random-function-definition]
value-specifier     →   value-reference |
                        DERIVATION RULE <behaviour-definition-label>
random-value-specifier  →   period-function-specifier
                            generate-function-specifier
get-replace     →   GET | REPLACE | GET-REPLACE
add-remove      →   ADD | REMOVE | ADD-REMOVE
period-function-specifier       → PERIOD random-function-definition
generate-function-specifier     → GENERATE random-function-definition
random-function-definition  → EXPONENTIAL <real-value>
                            | CONTINUOUS UNIFORM <real-value> <real-value>
                            | DISCRETE UNIFORM <integer-value> <integer-value>
                            | BINOMIAL <integer-value> <real-value>
                            | POISSON <real-value>
                            | GAUSSIAN <real-value> <real-value>
```

그림 2. 확장된 GDMO의 패키지 템플릿 문법

이 중 패키지 템플릿에서 속성과 통고를 실제 값이 아닌, 시뮬레이션 값을 사용한다는 것을 명시할 수 있다. 따라서, 사용자는 패키지 템플릿을 기술하면서 속성과 통고를 시뮬레이션한다는 것과, 시뮬레이션에 사용하는 주기 및 생성 함수의 종류와 각각에 대한 파라미터를 정의할 수 있다. 또한 주기 및 생성 함수를 기술하지 않은 경우는 RRS 운용 중에 GUI를 통해 사용자가 직접 지정할 수 있다.

4. E-GDMO 컴파일러

기존의 GDMO 컴파일러는 사용자가 기술한 GDMO를 입력으로 하여 에이전트에서 이용할 관리 객체 코드를 생성한다. 그러나 RRS에서는 실제 자원을 대신하여 시뮬레이션 하는 동작에 관련된 정보 역시 생성되어야 한다. 따라서 E-GDMO 컴파일러는 그림 3과 같이 사용자의 입력에서 추가된 문법을 분리하고, 이를 이용하여 SDT 정보를 생성한다.

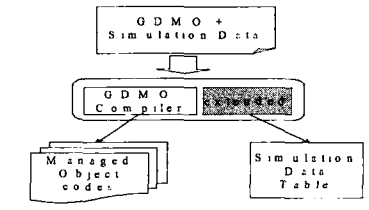

그림 3. E-GDMO 컴파일러의 동작

실제로 시뮬레이션에 관련된 부분은 SDT 정보뿐이지만, 원래의 GDMO 컴파일러의 출력인 관리 객체 코드도 같이 이용

할 수도 있으며, 이런 경우에는 망 관리시스템의 에이전트 생성도 동시에 수행할 수 있다. E-GDMO 컴파일러는 사용자가 기술한 문서를 받아들여 원래의 GDMO 부분과 시뮬레이션 관련 부분을 분리하여 내부 클래스에 저장한다. 이와 같이 분리된 데이터를 바탕으로 관리객체 코드와 SDT 파일을 생성하게 된다.

그림 4는 E-GDMO 컴파일러의 전체적인 구조를 보이고 있다. 여기에는 내부적으로 전체적인 동작을 수행하는 Main 루틴과 초기화 루틴, GDMO 문서 인식 루틴, 내부 데이터 클래스, 외부 파일 출력 루틴이 있다.

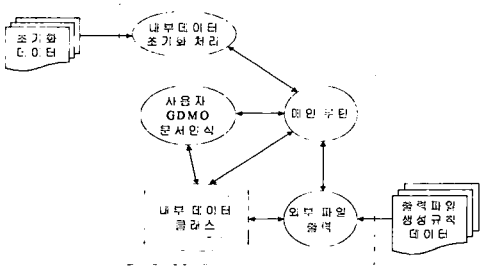

그림 4. E-GDMO 컴파일러의 구조

Main 루틴에서는 확장된 GDMO 컴파일러의 전체 동작을 관장하며, 각각의 내부 요소들과 상호 동작한다. Main 루틴의 주요한 동작은 ① 내부 데이터의 초기화와 ② 각각의 내부 요소의 동작 순서 유지, ③ 각 내부 요소에서 발생한 에러 처리 등이다. 내부 데이터 초기화 루틴에서는 E-GDMO 컴파일러가 수행하기 위해 필요한 초기화 파일들을 읽고, 그 정보를 Main 루틴에 전달하는 역할을 수행하고, GDMO 문서 인식 루틴에서는 ① 사용자가 기술한 문서를 읽어서 이를 해석하는 역할과 ② 해석된 데이터를 내부 데이터 클래스에 저장하는 역할, ③ 에러 발생 시 Main 루틴에 보고하는 역할, 그리고 ④ 사용자 문서에 대한 처리가 끝났을 때, 처리 결과를 Main 루틴에 보고하는 역할을 수행한다. 내부 데이터 클래스에는 사용자의 GDMO 문서에서 인식한 데이터들을 저장하고, 외부 파일 출력 루틴에서는 사용자 문서에서 인식된 결과를 이용하여 필요한 관리 객체 클래스 코드와 시뮬레이션 데이터를 생성하는 역할을 수행한다. 또한 초기화 데이터에서는 컴파일러를 동작시키기 위해 필요한 초기화 정보를 유지하는데, 여기에는 기존에 정의된 관리 객체, 속성, 신택스의 정보등이 포함된다. 마지막으로 출력 파일 생성 규칙 데이터에는 내부 데이터 클래스에 저장된 정보를 이용하여, 관리 객체 클래스와 시뮬레이션 데이터를 생성하는 규칙을 저장한다.

확장된 GDMO 컴파일러의 수행 과정은 다음과 같다. ① 확장된 GDMO 컴파일러는 초기화 데이터 파일을 읽고, 초기화 데이터 파일의 정보를 이용하여 내부 데이터를 초기화 한다. ② 컴파일러는 사용자가 작성한 GDMO 문서를 읽고, 사용자 문서에 포함된 정보들을 인식한다. ③ 사용자의 문서에서 인식된 정보를 이용하여 내부 데이터 구조에 해당 정보들을 저장한다. ④ 확장된 GDMO 컴파일러는 내부 데이터 구조에 저장된 정보를 이용하여, 출력 파일 생성 규칙에 따라 관리 객체 클래스 파일과 SDT를 생성한다.

다음 그림 5는 E-GDMO 컴파일러를 이용하여 만든 실험용 RRS의 동작 화면을 보이고 있다.

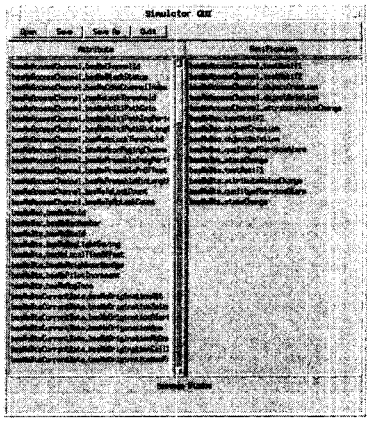

그림 5. E-GDMO 컴파일러를 이용하여 생성한 RRS의 동작 화면

5. 결론

본 논문에서는 실제 자원의 역할을 대신할 수 있는 RRS를 지원하기 위한 E-GDMO와 E-GDMO 컴파일러를 소개하였다. 구현된 RRS와 E-GDMO 컴파일러는 공개된 CMIP/CMIS 개발툴인 OSIMIS[12]를 기반으로 하여 구현하였다. 본 연구에서 개발한 E-GDMO 컴파일러와 RRS는 실제 자원의 다양성을 고려하여 실제 자원이 없이 에이전트를 개발하거나, 망 관리 시스템을 테스트할 수 있는 환경을 제공한다. 따라서, 이를 활용하면 망관리 시스템 개발 기간이 단축될 것이며, 망 관리 시스템 동작 평가와 성능 평가 역시 용이해질 것으로 기대된다.

참고문헌

[1] ITU-T M.3010, "Principles for a Telecommunication Management Network"

[2] ISO7498-4/ITU-T X.700, "OSI Basic Reference Model Part 4: Management Framework"

[3] ISO10040/ITU-T X.701, "Systems Management Overview"

[4] ISO9595/ITU-T X.710, "Common Management Information Service Definition"

[5] ISO9596-1/ITU-T X.711, "Common Management Information Protocol Specification"

[6] ISO10165-1/ITU-T X.720, "Management Information Model"

[7] ISO10165-2/ITU-T X.721, "Definition of Management Information"

[8] ISO10165-4/ITU-T X.722, "Guidelines for the Definition of Managed Objects"

[9] 송병권, 김건웅, 진명숙, "망관리시스템을 위한 테스트베드 설계", 한국정보처리학회, 13회 춘계학술대회 논문집, 2000

[10] 송병권, 김건웅, 진명숙, "TMN을 위한 실제 자원 시뮬레이터 설계", 한국통신학회 논문지 26권 10A호 pp.1727-1736, 2001

[11] 송병권, 김건웅, 진명숙, "망관리시스템 테스트베드를 위한 GDMO 컴파일러의 확장", 한국정보처리학회, 13회 춘계학술대회 논문집, 2000

[12] George Pavlou, "The OSIMIS Platform: Making OSI Management Simple", Integrated Network Management IV, 1995