http://dx.doi.org/10.3745/KIPSTA.2011.18A.6.281

AMI 네트워크 환경에 적합한 시뮬레이터 설계 및 구현에 관한 연구

양 일 권[†] · 정 남 준[††] · 이 상 호[†††]

요 약

본 논문에서는 NS - 2 시뮬레이션 소프트웨어를 적용하여 AMI 네트워크 환경의 네트웍 시뮬레이터를 설계 및 구현하였다. 데이터의 처리 시간, 네트웍 프로토콜 그리고 시스템 성능 등과 같은 다양하고 제한된 AMI 운영 환경에서 최적의 AMI 네트웍 구성을 구현하기 위하여 시뮬레이션 될 것이다. 따라서 본 AMI 네트웍 시뮬레이터는 실제 네트웍과 장비와 최대한 비슷한 환경에서 운영될 수 있도록 구현되었다. 본 시스템의 개발로 앞으로 AMI 네트웍 시뮬레이터는 유틸리티의 AMI 본격 전개시 AMI 운영 네트웍 설계에 적용됨으로써, 운영 자원의 낭비를 줄이고 AMI 운영의 효율성을 극대화 할 수 있을 것으로 기대한다.

키워드 : 에이엠아이, 첨단계량인프라, 네트웍 시뮬레이터

A study on the Design and Implementation of Simulator adapted in AMI Network environment

Il-Kwon Yang[†] · Nam-Joon Jung[††] · Sang-Ho Lee[†††]

ABSTRACT

In this paper, a simulator adapted in AMI network environment with NS-2 was designed and implemented. The limited AMI operation environment such as processing times of data, network protocol and system performance can be simulated to find out the optimal AMI formations. Consequently, it is simulated under conditions that are closely analogous to the actual networks and each device. In future, the result of simulation would be applied to AMI network, mitigated the waste of resources and much contributed towards the real optimal AMI deployments in utilities.

Keywords : AMI, Advanced Metering Infrastructure, Network Simulator, NS-2

1. 서 론

전 세계적인 저탄소 녹색 성장의 필요성에 따라, 스마트그리드에 대한 관심은 그 어느 때보다 크다. 스마트그리드는 기존 전력망에 정보통신 기술을 융합하여 소비자와 전력공급자 간에 실시간 에너지 정보를 양방향으로 주고받을 뿐 아니라, 전력 에너지 효율을 최적화하기 위한 차세대 전력망이다[1]. 첨단계량시스템(AMI, Advanced Metering Infrastructure)은 스마트그리드를 실현하기 위한 핵심인프라이자 필수적인 요소 기술이다. 하지만, AMI 네트워크 범위가 넓고 대상 가구가 1,800만 내외의 대단위 고객이 서비스 대상이 될 것으로 예상되기 때문에, AMI 도입 이전에 주요 장비 및 네트워크에 대한 적정 설계가 요구된다. 만약 주요 장비들이 적절한 성능을 고려하여, 구성·운영하지 못 한다면, 전반적인 시스템 전개에 있어서 운영의 어려움과 불필요한 자원의 낭비가 발생할 수 있다. 따라서 추진 중인 AMI 본격 구축 사업이 성공적으로 이루어지기 위해서는 시뮬레이션을 통한 현장의 네트워크 자원 및 시스템에 적합한 설계가 요구된다. 본 논문에서는 AMI 망의 성능 분석에 필요한 네트워크 시뮬레이터의 요구사항들을 도출하고, AMI 전력망에 적합한 시뮬레이터 개발을 위하여 NS-2 네트워크 시뮬레이터션 소프트웨어의 소스를 수정한 보완된 알고리즘을 제시하여 AMI 네트워크 환경에 적합한 AMI 네트워크 시뮬레이터를 설계하고 구현한 결과를 기술하였다.

† 정 회 원 : 한국전력공사 전력연구원 소프트웨어 센터장
†† 정 회 원 : 한국전력공사 전력연구원 책임연구원
††† 종신회원 : 충북대학교 전자정보대학 소프트웨어학과 교수
논문접수 : 2011년 9월 7일
수 정 일 : 1차 2011년 10월 25일, 2차 2011년 11월 24일
심사완료 : 2011년 12월 5일

2. 관련 연구

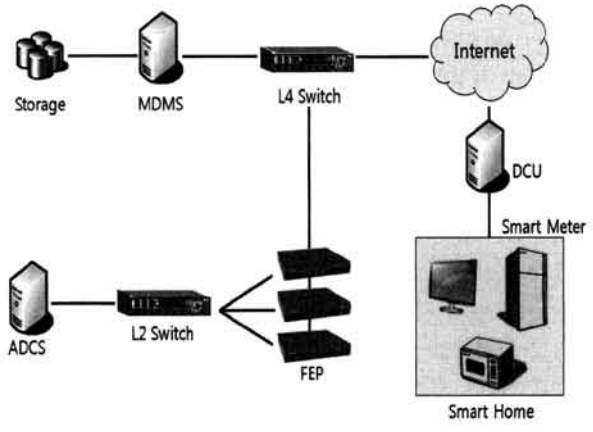
본 장에서는 AMI 시스템의 구성과 주요 동작을 살펴보고, 대표적인 분석방법인 시뮬레이션에 대해 간략히 설명한다. 그리고 AMI 네트워크의 성능 분석에 시뮬레이션 방법의 타당함을 증명한다. 또한 가장 널리 알려진 시뮬레이션 툴인 NS-2가 AMI 네트워크 환경에 대한 기반 시뮬레이션 툴로 지정한 근거를 제시하고 NS-2에 대해 소개한다.

2.1 AMI 시스템

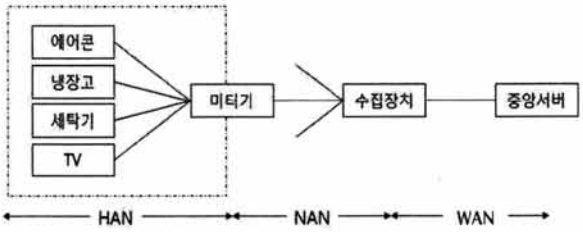
AMI는 AMR(Automatic Meter Reading)에서 한 단계 발전한 전력사용량 검침 기술로, 소비자 전력관리 및 서비스 인프라를 뜻한다. AMI는 Smart Meter와 양방향 통신을 기반으로 하기 때문에 미터의 원격제어와 수요관리 등 다양한 서비스가 가능하고, 정보교환을 통해 전력 소비량을 효율적으로 제어하여 전력의 소모량 및 비용을 줄일 수 있다[2]. AMI 시스템은 양방향 통신을 지원하고, 실시간 에너지 소비 정보 교환, 전력 품질 감시 기능, 자동 정산 기능 등의 서비스를 제공한다.

(그림 1)은 AMI 시스템의 전체적인 구성도를 나타낸다. AMI시스템은 데이터를 축적하고 분석하는 일을 담당하는 MDMS(Meter Data Management System), 데이터를 수집하는 일을 담당하는 ADCS(Automated Data Collection System), 데이터를 수집 및 전송하는 일을 담당하는 DCU(Data Collection Unit), 전력 사용량을 측정하는 Smart Meter와 같은 여러 장비로 구성된다. ADCS는 주기적으로 담당 도메인에 속한 모든 DCU로부터 검침 정보를 수신하게 되는데, 이 때 한꺼번에 트래픽이 발생하면서 부하가 생겨 검침 정보에 손실이 발생할 수 있다. 따라서 중간에 여러 대의 FEP(Front End Processor) 장치를 두어 부하를 분산시킴으로서 ADCS에 생기는 입출력 부하를 줄일 수 있다[3].

Smart Meter는 아날로그 수치를 디지털화 하는 기능을 가지고 있으며, 소비자와 공급자간의 통신 기능을 가지며 가정 내 홈 네트워크와 연동하는 기능도 갖는다[4].



(그림 1) AMI 네트워크 구성도



(그림 2) AMI 망구조

(그림 2)는 개략적인 AMI 망구조를 보여준다. AMI의 구현을 위해서는 기본적으로 각 가구에 설치된 미터기와 중앙서버가 네트워크로 연결되고 이들 간에는 양방향 통신이 요구된다. 로컬 수집 장치가 여러개 그룹의 미터로부터 데이터를 수집하고, 이를 WAN(간선망)으로 연결된 중앙서버로 전송하는 구조로, HAN(Home Area Network), NAN (Neighborhood Area Network), WAN(Wide Area Network) 등으로 구성된다[5].

2.2 네트워크 시뮬레이터 : NS-2

예상되는 구축 환경에서 각 장비들의 성능을 평가하고 이를 통하여 각 장비에 요구되는 적정한 성능 및 시스템 구성 방안을 도출해내기 위한 방법으로는 크게 시뮬레이션, 에뮬레이션, 그리고 수학적 계산을 통한 방법이 있다[6]. 본 논문에서는 성능 분석의 대상으로 삼고 있는 AMI 네트워크 환경이 국내에서 전력을 이용하는 1,800만 저압 모든 가구의 미터기와 이들 미터기로부터 데이터를 수집하는 수십만 대의 수집 장치를 포함하므로 실제 환경에서의 실험을 통하여 결과를 도출한다는 것은 현실적으로 불가능하다. 따라서 대규모 네트워크 환경에 적합한 시뮬레이션 방식을 기반으로 분석을 수행하는 것이 AMI 네트워크에 배치되는 각 장비에 대한 적정 성능을 도출하기에 적합하다. 네트워크 시뮬레이터는 사용자에게 실제와 같은 규모를 가진 가상의 네트워크를 구성할 수 있도록 여러 기능들을 제공하며, 구성된 네트워크가 동작하였을 때 발생하는 에러상황, 패킷 손실 등과 같은 문제들을 예상하여 상정 할 수 있다. 사용자는 네트워크 시뮬레이터의 실행 결과를 통해 이러한 문제를 사전에 예측할 수 있으며, 실제 구현 시 발생할 수 있는 시행착오를 줄 일 수 있다.

가장 대표적으로 사용되는 네트워크 시뮬레이터에는 Opnet, Qualnet, NS-2, NS-3 등이 있다. <표 1>에서는 3가지 네트워크 시뮬레이터들의 특징에 대해 비교하고 있다[7][8][9][10].

아래의 네트워크 시뮬레이터 중 NS-2는 연구용으로 개발된 오픈 소스 시뮬레이터이며 현재 연구용으로 가장 많이 사용되고 있다. NS-2는 실제의 네트워크 환경을 가상적으로 구현하고 이를 이용하여 시뮬레이션을 수행한다[7]. NS-2의 특징으로는 이산 사건 시뮬레이터(Discrete Event Simulator)로 이상 시간 포인트에서만 상태 변수가 변하는 시스템을 모델링하는 것을 말한다. 또한, 이벤트 기반의 시뮬레이션을 사용하며, 유무선 네트워크 모두를 지원한다.

NS-2의 장점으로 다양한 네트워크 컴포넌트들이 지원되며 업데이트 속도가 빠르다. 또한 네트워크의 기능과 프로토콜 등이 모듈화 되어 컴포넌트들을 추가하고 수정하기에 편리하여 사용자의 개발 자유도가 매우 높다는 장점이 있다. 따라서 본 논문에서는 AMI 시뮬레이터를 구현하기 위하여 오프소스이며 네트워크의 기능 및 프로토콜의 추가·수정하는 개발 자유도가 높은 NS-2를 활용하였다.

〈표 1〉 네트워크 시뮬레이터의 특징

네트워크 시뮬레이터	특 징
Opnet	- 현존하는 대부분의 프로토콜 지원 및 뛰어난 GUI 제공 - 높은 구매 비용(1억 4,784만 원)으로 사용이 어려움 - 개발언어 : C/C++ - 모델 구현 방식 : FSM과 Condition을 이용한 계층적 모델링 구조
Qualnet	- 혼합 플랫폼 네트워크 모델링 지원 & 병렬 CPU를 고려한 설계로 CPU 활용도가 높음 - 무선/유선/혼합 - 높은 구매 비용(1억 3,000만 원)으로 사용이 어려움 - 개발언어 : C/C++, PARSEC - 모델 구현 방식 : C/C++와 PARSEC으로 프로그래밍
NS-2	- 오픈 소스 기반의 무료 소프트웨어이며, 다양한 프로토콜 제공 - C++과 OTcl 2개의 객체지향 언어로 작성되었기 때문에 두 언어에 대한 이해가 필요 - 개발언어 : C/C++, Tcl - 모델 구현 방식 : C/C++와 Tcl으로 프로그래밍

2.3 AMI 네트워크 시뮬레이터 개발 사례

AMI 분야는 새로 구현되어 운영중인 기술로 북미 및 유럽을 중심으로 확산되는 추세이다. 이러한 AMI 네트워크 및 자원 설계를 위하여 효율적인 AMI 네트워크 시뮬레이터의 개발은 일반화되어 있지 않으며 구현된 제품은 SUM Global Technology의 'AMI Simulator Tool'이 유일한 제품이다. 2007년에 발표되었으며 네트워크의 신뢰성과 성능 그리고 네트워크 자원의 설계에 중점을 두고 개발되었다.[8] 본 소프트웨어를 활용한 AMI 네트워크 컨설팅 사업이 주요 비즈니스 모델이다.

3. AMI 네트워크 시뮬레이터 설계

본 논문에서 제시하는 시뮬레이터의 최종 목적은 출력된 결과 데이터를 통하여 AMI 네트워크 환경에서 MDMS, ADCS, FEP, DCU의 배치 적정 수준을 도출하기 위한 시스템 설계와 구현을 하는 것이다. 따라서 이것들을 도출하기 위하여 사용될 시뮬레이션 기능에 대한 정의가 필요하다. 또한 대규모 네트워크 환경을 고려하여 시뮬레이션 성능을 최대한 높이면서 필요한 결과를 정확하게 출력할 수 있도록 시뮬레이터를 구현해야 하며 이를 위한 입력 파라미터도 정의 되었다[11][12].

3.1 요구사항 정의

AMI 네트워크를 구성하는 각 시스템의 적정 성능을 도출하기 위해 필요한 요구사항에 대한 정의는 <표 2>와 같다.

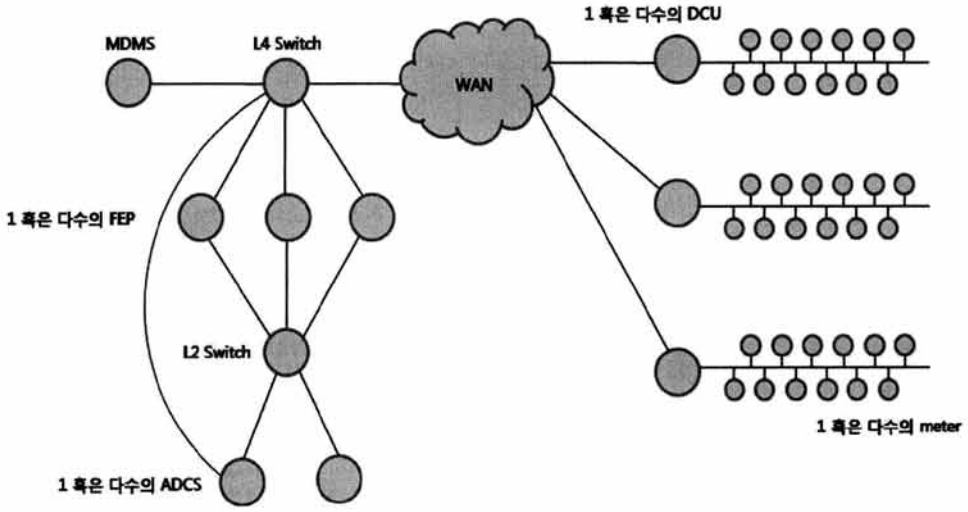
〈표 2〉 요구사항 정의

요구사항	설 명
REQ_1	시뮬레이션 수행 후 각 노드의 유입 트래픽 정보 및 손실 트래픽 정보를 상세히 파악할 수 있어야 한다.
REQ_2	시뮬레이션 수행 후 각 DCU의 미터 검침 데이터 수집 정보(수집한 데이터의 양, 수집 시간 등)를 검침 주기별로 상세히 파악할 수 있어야 한다.
REQ_3	시뮬레이션 수행 후 MDMS, ADCS, FEP의 주기적 유입 트래픽 량을 파악하여 트래픽 추이 그래프를 나타낼 수 있어야 한다.
REQ_4	AMI 환경에 구현된 검침 프로토콜에 의한 동작을 재현해야 한다.
REQ_5	각 시스템의 성능에 따른 TASK 처리시간이 반영되어야 한다.

가장 필요한 공통적인 정보는 유입 트래픽에 대한 트레이스 정보이다. 유입 트래픽의 양과 전송 실패한 트래픽의 양을 통해 장비의 성능이 얼마나 적정한지 판단 할 수 있다. MDMS와 ADCS, FEP의 경우 각 장비의 처리 역량과 유입 트래픽량 및 잔여 버퍼량, 손실된 패킷량을 비교하여 해당 시스템 성능의 지나침 또는 모자람 정도를 판단할 수 있으므로 AMI 네트워크 시뮬레이터에 대한 첫 번째 요구사항은 [REQ_1]로 정의된다. 그리고 [REQ_1]을 만족시키기 위하여 시뮬레이터 내 존재하는 각 노드로부터 추출해 내고자 하는 공통적인 정보를 <표 3>과 같이 정의한다. 또한 [REQ_2]를 위해서도 각 노드의 트레이스 정보들이 정의된다.

〈표 3〉 [REQ_1]을 위한 각 노드의 공통 트레이스 정보

이 름	내 용
Node ID	NS-2 내 노드 식별 번호
Total Traffic	노드 내에 유입된 전체 트래픽 크기
Peak Traffic	1초간 피크 트래픽 양
Average Traffic	초당 평균 트래픽 양
Peak Buffer	피크 버퍼 점유량
Average Buffer	평균 버퍼 점유량
Packet Drop	패킷 드롭량
Packet Drop Count	패킷 드롭 횟수
Collision Count	패킷 충돌 횟수
Retransmission Count	패킷 재전송 횟수



(그림 3) AMI 시뮬레이션 토폴로지

그리고 기존의 NS-2는 네트워크 프로토콜과 전송매체에 의한 패킷 전송 지연 시간은 반영되나 시스템 성능에 의한 데이터 처리 시간은 반영되지 않는다. 따라서 [REQ_5] 조건 만족을 위한 NS-2 노드 구성의 변경 및 재설계가 필요하다.

3.2 시뮬레이터 토폴로지 및 입력 파라미터 정의

본 논문에서 제안하는 시뮬레이터는 AMI 네트워크만을 대상으로 설계되었으며, 따라서 기본적으로 다음과 같은 토폴로지 구성을 따라야 한다.

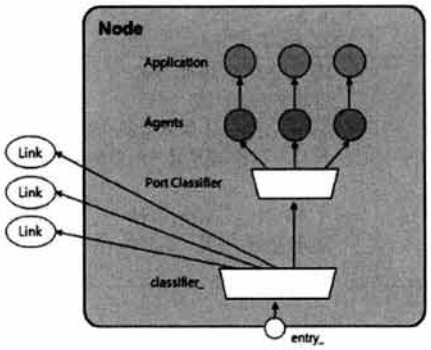
각 노드와 사용자 정의 통신 프로토콜, 검침 프로토콜은 각각 다음의 입력 파라미터를 가진다.

〈표 4〉 입력 파라미터 정의

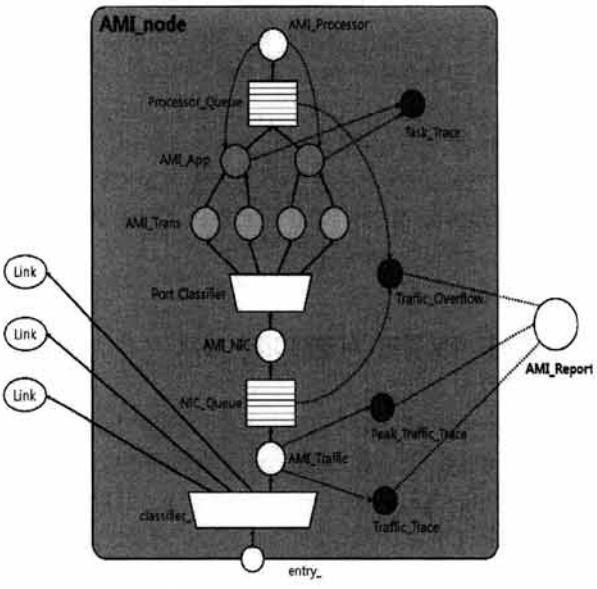
노드입력 파라미터	통신입력 파라미터	검침 프로토콜입력 파라미터
Bandwidth	Flow & Error control	App Type
Buffer	Frame size	최대 응답 지연 시간
Processor	Header size	재전송 횟수
Memory	Ack size	일반 검침 주기
		현재 검침 주기
		Ack Type
		Reqeust Type

3.3 NS-2 소스 수정을 통한 시뮬레이터 코어 설계

NS-2는 네트워크 프로토콜에 대한 시뮬레이터로서는 매우 큰 장점을 가지고 있으나, 각 노드에서 TASK 처리 시 위 <표 4>의 검침 프로토콜 입력 파라미터를 반영할 수 있는 시스템 성능에 따른 지연 시간을 실질적으로 반영하지 못하며, 각 장비의 적정도를 산출하는데 있어서 필요한 트레이스 데이터를 생성하지 않는다. 또한 NS-2에는 AMI 네트워크 환경에 특화된 검침 프로토콜을 수행하는 오브젝트가 구비되어 있지 않다. 따라서 본 논문에서는 (그림 4)과 같은 기존 NS-2 노드의 구조를 (그림 5)와 같이 변경하고, 필요한 오브젝트를 추가하였다.



(그림 4) NS-2 노드의 변경 전 (기존 NS-2의 노드 구성)



(그림 5) AMI 네트워크 시뮬레이터를 위한 NS-2 노드의 변경

〈표 5〉 NS-2 노드구성에 추가된 오브젝트

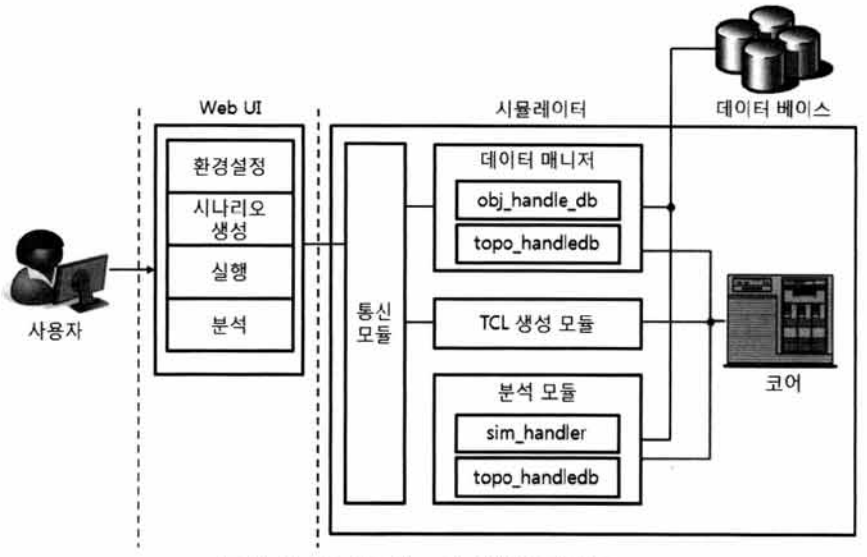
오브젝트	설명
AMI_Traffic	AMI_Traffic 클래스로 정의된 오브젝트로, 노드로의 유입 트래픽을 관찰하고 이로부터 트레이스 정보를 생성한다. 트래픽 정보 저장을 위하여 Linked List로 구성된 traffic_table을 가진다. traffic_table에는 항상 최근 1초 이내의 트래픽 정보만 남고, Peak Traffic 양을 구할 수 있다.
NIC_Queue	네트워크 인터페이스를 적용하기 위한 오브젝트로, 기존 Queue를 활용하여 NIC(Network Interface Card) 버퍼와 같은 역할을 수행한다. 즉, 노드가 가진 NIC의 버퍼 크기를 반영하여 패킷이 NIC 버퍼에 머무르는 상황을 재현한다.
AMI_NIC	네트워크 인터페이스를 적용하기 위한 오브젝트로, NIC의 성능과 패킷의 크기에 따른 시간 지연이 반영되어 패킷을 노드 내로 유입시키는 역할을 담당한다.
AMI_Trans	AMI_Transport 클래스를 부모로 하여 정의된 오브젝트로, 통신 프로토콜을 정의한다. TCP/UDP/Zigbee/PLC/사용자정의 통신 프로토콜 오브젝트가 존재한다.
AMI_App	AMI_Application 클래스를 부모로 하여 정의된 오브젝트로, 검침 Task 프로토콜을 정의한다.
Processor Queue / AMI Processor	시스템의 프로세서와 같은 역할을 하여 Task의 처리순서 및 속도를 반영하여 프로세서의 성능에 따른 프로세싱 딜레이를 적용한다.
Task_Trace	AMI_Trace 클래스로 정의된 오브젝트로, DCU 적정도 산출을 위해 필요한 트레이스 정보를 생성한다.
Traffic_Overflow	AMI_Trace 클래스로 정의된 오브젝트로, NIC 버퍼 또는 시스템 내 메모리 부족으로 손실된 데이터에 대한 트레이스 정보를 생성한다.
Traffic_Trace	AMI_Trace 클래스로 정의된 오브젝트로, 초당 유입 트래픽 량을 주기적으로 생성한다.
Peak_Traffic_Trace	AMI_Trace 클래스로 정의된 오브젝트로, 1초간 피크 트래픽 량에 대한 트레이스 정보를 주기적으로 생성한다.

<표 5>은 기존의 NS-2 노드 구성에 추가된 오브젝트에 대한 간략한 설명이다.

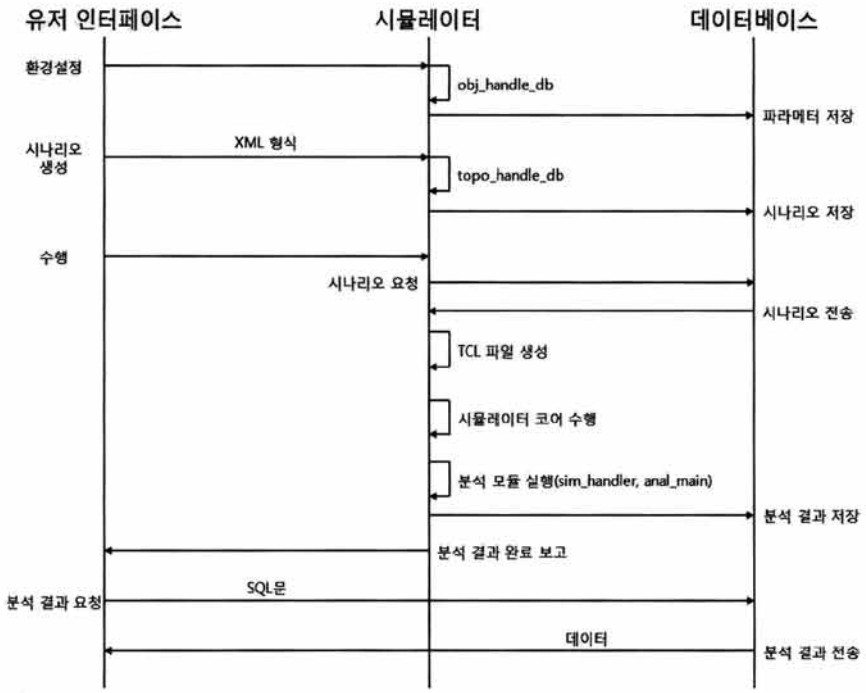
NS-2 노드 내에서 프로세서의 성능에 따른 프로세싱 지연(delay)을 적용하기 위하여 AMI Processor 오브젝트와 Processor Queue를 구현하였다. Processor Queue는 NS-2에서 제공하는 기존의 Queue 클래스를 활용하여 특정 Job이 프로세서에 의해 처리되기를 기다리는 대기 큐에 해당하는 역할을 수행하도록 설계되었다. AMI Processor는 프로세서의 성능과 Job의 크기에 따른 시간 지연이 반영되어 실제 Job 처리 시간 적용하는 역할을 담당하는 오브젝트이다. AMI Processor는 또한 프로세서의 Idle 시간과 프로세서 동작 시간을 지속적으로 계산하여 저장하고 있다가 주기적으로 AMI Report 오브젝트에 전달하여 CPU 처리율에 대한 트레이스 정보를 생성할 수 있게 한다.

3.4 AMI 네트워크 시뮬레이터 상세 설계

AMI 네트워크 시뮬레이터는 사용자가 웹 페이지를 통해 접근하여 가상적인 AMI 시스템을 구축하고, 시뮬레이션 할 수 있는 프로그램이다. (그림 6) AMI 네트워크 시뮬레이터의 전체 구성도에서 보는 바와 같이 사용자가 AMI 네트워



(그림 6) AMI 네트워크 시뮬레이터 전체 구조



(그림 7) AMI 네트워크 시뮬레이터 동작 순서 다이어그램

크 시뮬레이터에 접근하고 제어할 수 있는 인터페이스를 제공 유저 인터페이스(Web UI)는 시나리오를 위한 환경을 구성하고 시뮬레이션 수행 후,수행 결과를 조회하기 위한 모듈이며, 통신모듈은 유저 인터페이스를 통해 접근하는 사용자와 시뮬레이터 간 명령, 이벤트 등을 상호연결(Interconnection)하는 역할을 수행한다. 데이터 매니저 모듈은 사용자가 유저 인터페이스의 환경설정에서 구성한 오브젝트와 토폴로지의 구성을 데이터베이스에 저장, 수정 , 삭제 등의 기능을 수행한다.

TCL(Tool Command Language) 생성 모듈은 NS-2 시뮬레이터가 사용자가 설정한 오브젝트와 토폴로지를 통해 시뮬레이션을 수행하기 위해 실행하는 파일이며, 사용자가 시뮬레이션을 실행 시에 자동으로 생성하여 준다. 분석 모듈은 사용자가 작성한 시나리오를 실행하였을 때 발생하는 결과를 분석하고 사용자에게 보고하는 기능을 수행한다.

(그림 7)은 위 (그림 6)의 모듈들을 기반으로 AMI 네트워크 시뮬레이터의 전반적인 동작 순서 다이어그램이다.

3.4.1 유저 인터페이스 설계

AMI 네트워크 시뮬레이터의 유저 인터페이스(Web UI)는 시나리오 조건 설정을 위하여 시뮬레이션에 필요한 Object (장비)와 Protocol(통신/Task)을 생성 및 관리 장비를 추가하는 화면으로, AMI 구성 요소인 MDMS, ADCS, DCU, METER, FEP, SWITCH에 대한 장비 파라미터 정보와 통신 프로토콜을 정보로서 입력하여, 노드와 노드 간에 통신이 구성되는 PLC, Zigbee, TCP, UDP 등과 같은 설정 파라미터의 정의 그리고 어플리케이션의 메시지 송/수신 기법에 대한 정보 파라미터가 시스템에서 결정되도록 처리한다.

3.4.2 통신모듈 설계

유저 인터페이스를 통해 사용자의 control message를 전달하게되는데, control message에는 object, protocol, topology의 생성과 관리, 시뮬레이션 수행, 시뮬레이션 수행 결과를 출력하는 기능이 포함된다. NS-2 시뮬레이터는 이러한 메시지를 송/수신하기 위해 TLV(Type Length Value) 메시지 구조를 사용하여 통신한다. TLV 구조는 메시지 헤더와 페이로드 부분으로 구성되며, 통신에서 사용되는 헤더의 필드는 SubSystem ID의 Type 필드와 Action Type, Info, Error Code 등에 해당한다. Subsystem ID 필드에서는 UI인지, Main Simulator인지에 대해서 식별할 수 있도록 Type값이 저장되고, Action Type 필드에는 Action Type[0]에는 Object, Protocol, Topology의 Request, Reply에 대한 값이 들어가고, Action Type[1]에는 데이터 저장, 수정, 삭제와 시뮬레이션 시작, 중단 등의 action이 주어지도록 설계하였다. Info 필드에는 각 노드별 정보, 토폴로지 정보, Report하기 위한 상태 정보 등의 정보가 들어간다. Sequence Number 필드는 순서번호를 위한 필드이며, Error Code 필드에는 서버에서 내부 처리 과정에서 오류 발생 시 적절한 에러 처리를 할 수 있도록 설계되었다.

각 필드에서 사용되는 세부 정보는 다음 <표 6>과 같다.

3.4.3 시뮬레이션 데이터 메니져

시뮬레이션 데이터 매니저는 AMI 네트워크에서 장비, 프로토콜, 그리고 토폴로지의 데이터를 관리하는 기능을 제공한다. 사용자가 Web UI를 통해 장비, 프로토콜, 또는 토폴로지의 데이터를 저장, 수정하거나 삭제요청을 하면 해당 요청이 시뮬레이터로 전달되고 전달된 요청에 따라 시뮬레

〈표 6〉 헤더의 각 필드별 세부 정보

Field	Value	Type	Description
Subsystem ID_type	0x01	Web UI	UI
	0x02	AMI Network Simulator	Main Simulator
Action Type [0]	0x01	OBJ_REQ	Object 관련 Request
	0x02	OBJ_REP	Object 관련 Reply
	0x03	PRO_REQ	Protocol 관련 Request
	0x04	PRO_REP	Protocol 관련 Reply
	0x05	TOP_REQ	Topology 관련 Request
	0x06	TOP_REP	Topology 관련 Reply
	0x07	SIM_REQ	Simulation 요청
	0x08	SIM_REP	Simulation 응답
	0x09	SIM_REPORT	Simulation 과정에서 보고
	0x0A	LOG_REQ	Log 관련 Request
	0x0B	LOG_REP	Log 관련 Reply
Action Type [1]	0x01	STORE	데이터 저장
	0x02	DELETE	데이터 삭제
	0x03	REWRITE	데이터 수정
	0x04	SIM_START	Simulation 시작
	0x05	SIM_STOP	Simulation 중단
	0x06	TARGET	보고 받을 Target 설정
	0x07	REPORT	Simulation 중간 데이터 전달
	0x00	NO_ACTION	No Action
Info Field	0x01	MDMS_INFO	MDMS 정보
	0x02	ADCS_INFO	ADCS 정보
	0x03	SWITCH_INFO	SWITCH 정보
	0x04	DCU_INFO	DCU 정보
	0x05	METER_INFO	Meter 정보
	0x31	FEP_INFO	FEP 정보
	0x06	COMM_PRO_INFO	통신 프로토콜 정보
	0x07	APPLI_PRO_INFO	어플리케이션 프로토콜 정보
	0x08	TOPOLOGY_INFO	토폴로지 정보
	0x09	SIMUL_INFO	시뮬레이션 정보
	0x10	TOTAL_REPORT	전체 토폴로지의 상태 정보
	0x11	MDMS_REPORT	MDMS 노드의 상태 정보
	0x12	ADCS_REPORT	ADCS 노드의 상태 정보
	0x13	SWITCH_REPORT	스위치 노드의 상태 정보
	0x14	DCU_REPORT	DCU 노드의 상태정보
	0x17	EVENT_REPORT	시뮬레이션 이벤트 보고
	0x18	STATE_REPORT	시뮬레이션 상태 보고
ErrorCode			

이션 데이터 매니저가 동작하게 된다. 이러한 시뮬레이션 데이터 매니저에는 장비와 프로토콜의 데이터를 관리하는 obj_handle_db와 토폴로지 데이터를 관리하는 topo_handle_db로 설계되었다.

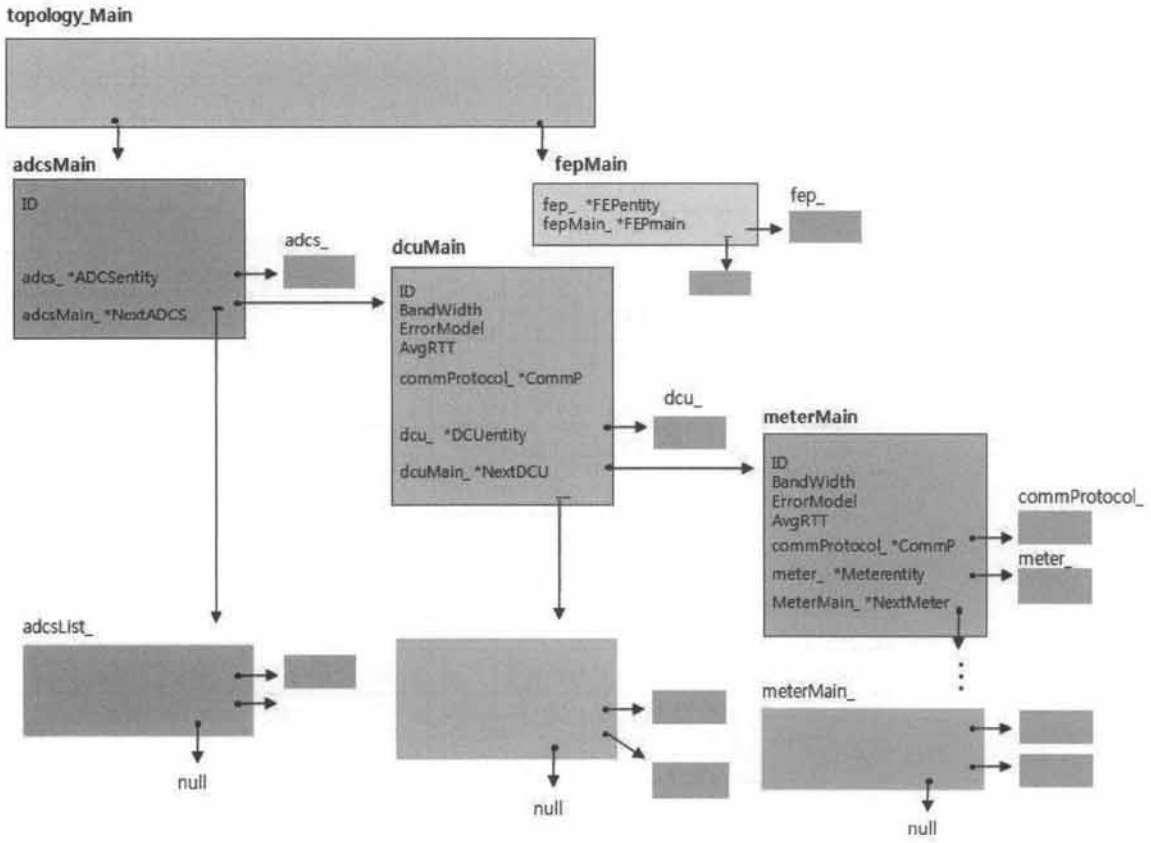
AMI 네트워크에서 모든 장비는 MDMS, ADCS, DCU, Meter, FEP, Switch L2, Switch L4 중 하나에 속한다. MDMS는 하나만 존재하는 데에 반해 ADCS, DCU, Meter, FEP는 하나 이상의 장비들이 존재한다. ADCS 하나는 단일 종에서 이종 이상의 DCU를 연결될 수 있으며 연결되는 DCU 개수도 하나에서 수십 개 이상이 될 수도 있다. ADCS와 마찬가지로, DCU 하나에 단일 종에서부터 이종이상의 Meter들이 불특정한 수로 연결될 수 있다. 이러한 구조적 특징은 AMI 네트워크에 변할 가능성에 매우 희박하여 이러한 특징을 기반으로 AMI 네트워크의 토폴로지 데이터를 표현할 수 있다. 토폴로지의 구성을 위해 설계된 데이터 구조는 다음 (그림 8)과 같다.

3.4.4 TCL 파일 생성 모듈

NS-2를 기반으로 하는 시뮬레이터가 사용자가 정한 시나리오에 따라 시뮬레이션을 수행하기 위해서는 시나리오에 따른 TCL 파일이 필요하다. TCL 파일은 네트워크 장비, 프로토콜 그리고 토폴로지 등 네트워크의 기본이 되는 모든 정보를 담고 있으며, 사용자가 UI를 통해 입력하는 입력 값을 반영하기 위해 TCL파일을 생성하기 위한 모듈이 존재한다. Web UI를 통해 입력하는 값이 시뮬레이션에 적용되어야 하고, 시뮬레이션에 적용되기 위해서는 입력 값에 따라 TCL 파일이 유동적으로 변해야 한다. TCL 파일은 장비 및 링크 관련 설정, Transport Agent 및 Task Agent 관련 작성 함수들을 가지고 TCL 파일이 완성되고, 이 파일이 시뮬레이터 코어의 입력이 되어 시뮬레이션이 수행된다.

3.4.5 트레이스 분석 모듈

트레이스는 NS-2 네트워크 시뮬레이터가 시뮬레이션를 수행하면서 발생한 이벤트들을 저장하는 파일이다. NS-2가 기본적으로 제공하는 트레이스 파일에는 시뮬레이션 중에 발생한 모든 네트워크 트래픽의 흐름이 저장되어 있으나, 가공되지 않은 데이터이기 때문에 직접적인 이용이 어렵다. 또한 AMI 네트워크 시뮬레이터에서 분석해야 하는 요소들을 추출하기 어렵기 때문에, AMI 네트워크 시뮬레이터 고유의 여러 트레이스 파일을 생성하여 분석에 사용한다. <표 7>은 시뮬레이터가 사용하는 여러 트레이스 파일 중 index_xx_final.trace 파일의 예시이다. xx는 시나리오 번호를 의미한다. 1번 필드는 NS-2 시뮬레이터내의 노드 수이다. 사용자가 구성한 시나리오에 존재하는 모든 노드의 수를 의미한다. 2번 필드는 FEP와 DCU의 예상 검침 수와 실제 검침 수에 대한 데이터를 유지한다. 3번 필드는 시뮬레이터가 시나리오를 실행한 전체 시뮬레이션 시간을 의미하며, 단위는 초(Second)이다. 4번 필드는 각 노드의 트래픽과



(그림 8) 토폴로지의 데이터 구조

〈표 7〉 index_xx_final.trace의 구조

내 용	필드번호
25	1
1872 1920 960 960	2
2763.902000	3
0 3328416.000000 24628.000000 1204.638436 1514.000000 103.563518 0.000000 0 -1 -1 1 4066920.000000 30370.000000 1471.921824 1514.000000 103.563518 0.000000 0 -1 -1 2 6656832.000000 49520.000000 2409.276873 2042.000000 139.680782 0.000000 0 -1 -1 3 3328416.000000 24892.000000 1204.638436 2042.000000 139.680782 0.000000 0 -1 -1 4 1355640.000000 11500.000000 490.640608 1514.000000 103.563518 0.000000 0 -1 -1 5 1355640.000000 11500.000000 490.640608 1514.000000 103.563518 0.000000 0 -1 -1 6 1355640.000000 11500.000000 490.640608 1514.000000 103.563518 0.000000 0 -1 -1 7 598656.000000 1396.000000 216.668838 628.000000 100.558813 0.000000 0 0 0 8 598850.000000 1396.000000 216.739052 522.000000 101.076366 0.000000 0 0 0	4

〈표 8〉 트레이스 파일 구성 및 저장 정보 설계

Trace	저장 정보
index_xx_final.trace	시뮬레이션에 대한 전반적인 정보를 유지
index_xx_final_(mdms, adcs, fep).trace	각 장치별 트래픽 처리량에 대한 데이터를 매초 마다 저장하기 위한 트레이스 파일
index_xx_final_(mdms, adcs, fep)_mem.trace	각 장치별 메모리 사용량을 매초 마다 저장하기 위한 트레이스 파일
index_xx_final_(mdms, adcs, fep)_occ.trace	각 장치별 프로세서 사용량을 매초 마다 저장하기 위한 트레이스 파일
index_xx_nt.trace	각 DCU의 검침 주기별 검침 정보를 나타내는 트레이스 화일

〈표 9〉 분석 결과 저장 구조체

구조체	내용
p_SIMULATION_MAIN	시뮬레이션의 토폴로지 인덱스와 분석 결과 인덱스, p_SIMULATION_RESULT 구조체 포인터의 주소 등을 저장하는 구조체
p_SIMULATION_RESULT	MDMS의 분석 결과, 시뮬레이션 전체 결과, 장치별 분석 결과 구조체 포인터를 저장하는 구조체
p_BUFFER_MDMS_ENTITY	MDMS의 버퍼에 존재한 트래픽량을 시간별 그래프로 표현하기 위한 구조체
p_RESULT_ADCS_ENTITY	ADCS의 분석 결과를 저장하는 구조체
p_ADCS_BUFFER_ENTITY	ADCS의 버퍼에 존재한 트래픽량을 시간별 그래프로 표현하기 위한 구조체
p_RESULT_FEP_ENTITY	FEP의 분석 결과를 저장하는 구조체
p_FEP_BUFFER_ENTITY	FEP의 버퍼에 존재한 트래픽량을 시간별 그래프로 표현하기 위한 구조체
p_RESULT_SWITCH_L2	L2 스위치의 분석 결과를 저장하는 구조체
p_RESULT_SWITCH_L4	L4 스위치의 분석 결과를 저장하는 구조체
p_RESULT_METER_ENTITY	DCU의 분석 결과를 저장하는 구조체
p_RESULT_METER_COLLECTING	라운드당 DCU가 미터의 정보를 수집하는데 경과된 시간과 수집한 양을 저장하는 구조체

버퍼에 대한 정보를 나타내며, 각 속성의 의미는 각각 NS2 내 노드 식별 번호, 노드 내에 유입된 전체 트래픽 크기(단위 : byte), 1초간 피크 트래픽 양(단위 : byte/sec), 초당 평균 트래픽 양(단위 : byte/sec), 피크 버퍼 점유량(단위 : byte), 평균 버퍼 점유량(단위 : byte), 패킷 드롭량(단위 : byte), 패킷 드롭 횟수(단위 : 개), 패킷 충돌 횟수(단위 : 개), 재전송 횟수(단위 : 개)를 나타낸다.

충분한 분석을 위하여 설계된 트레이스 파일은 시뮬레이션에 대한 전반적인 정보를 저장하는 index_xx_final.trace 등 총 11종이며 그 트레이스 파일에 대한 설명은 <표 8>과 같다.

3.4.6 분석 모듈

분석모듈은 트레이스 파일에 저장되어 있는 데이터를 이용하여 시뮬레이션 결과 분석에 필요한 데이터를 생성하는 모듈이다. sim_handler는 노드의 정보를 저장하는 NODE_INFORMATION 구조체와 시뮬레이션의 트래픽량, 수행 시

간 등의 정보를 저장하는 SIMULATION_INFORMATION 구조체를 전역 변수로 선언하여 메모리에 적재한다. 두 구조체는 이후 분석 모듈인 anal_main에서 사용되어 진다. 그 외에도 분석 모듈은 분석을 위한 함수 외에, 분석 결과를 데이터베이스에 저장하기 위해 각 장치별 전역 구조체를 갖는다. <표 9>는 각 장치들의 분석 데이터를 데이터베이스에 저장하기 위한 구조체들이다.

3.4.7 적정도 판정 기준 설정

MDMS, ADCS, FEP의 적정도는 두 가지 측면에서 판단하는데, 처리 역량 적정도와 메모리 적정도이다. 처리 역량 적정도는 시스템이 영구적으로 가용성을 유지할 수 있는지를 판단하는 척도이다. 시스템의 초당 트래픽 처리량을 R_{perf}, 초당 트래픽 유입량을 R_{traf} 라고 했을 때 처리 역량 적정도 P_{proc} 는 다음과 같은 식으로 구할 수 있다.

$$P_{proc} = 100 \times (R_{traf} / R_{perf})$$

주기 검침 이외에 발생하는 트래픽을 고려하였을 때 P_{proc} 가 80 이하이면 안정적인 트래픽 처리 역량을 지녔다고 판단한다. 참고로 P_{proc}가 100 이상이면 데이터가 메모리 내에 누적됨을 의미하므로 메모리 크기와 상관없이 일정 시간이 지남에 따라 데이터 오버플로우가 발생할 수 밖에 없다. R_{perf}는 프로세서의 $tpmC$ 를 이용하여 구할 수 있으며 다음의 식을 사용한다.

$$R_{perf} = (\text{1개 트래픽의 평균 크기}) / \{ 60 / (tpmC / F) \}$$

위 식에서 F는 1개 트래픽을 처리하는 데 걸리는 시간에 대한 실측 값을 바탕으로 구해진 상수이다.

메모리 적정도는 시스템이 트래픽이 몰리는 시점에 가용성을 유지할 수 있는 충분한 메모리 용량을 가지고 있는지 판단하는 척도이다. 시뮬레이션을 통하여 얻을 수 있는 $B_{\max}$는 메모리 버퍼에 상주하였던 최대의 데이터 량을 나타내며 M 메모리의 크기를 나타낸다고 했을 때, 메모리 적정도 P_{mem}은 다음과 같은 식으로 구할 수 있다.

$$P_{mem} = 100 \times (B_{\max} + R_{traf}) / M$$

4. AMI 네트워크 시뮬레이터 구현 및 검증

본 장에서는 앞서 설명하였던 설계를 바탕으로 구현된 AMI 네트워크 시뮬레이터를 다양한 조건으로 실행시켜봄으로서 실용성을 평가한다. 우선 시뮬레이션 정상 동작 테스트를 통하여 사용자가 설정한 토폴로지에 대한 시뮬레이션 기능이 정상적으로 수행되는지 확인하여 시뮬레이션 결과 데이터의 유효성을 판단한다. 또한 적정도 적합성 테스트를 통하여 도출된 분석결과를 현 운용 시스템의 경험 데이터와 비교시 타당한지 판단함으로써 본 시뮬레이터의 실질적인 활용 가능 여부를 분석한다.

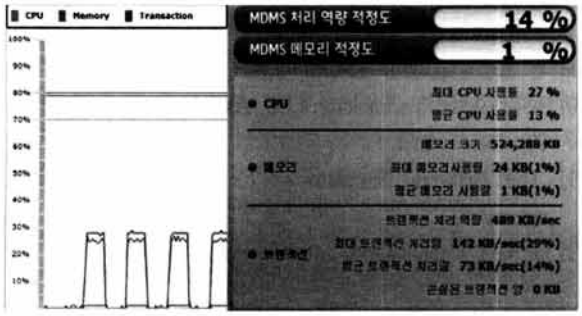
4.1 시뮬레이터의 기능 검증

시뮬레이션 동작환경을 검침 Task 정상 동작 환경과 각 시스템의 역량 부족 환경을 설정하여 시뮬레이션을 시행하여 보았다. 정상 환경은 현재의 시스템 운영 환경을 위하여 도입한 시스템의 운영 환경을 그대로 적용했으며, 시스템 역량 부족 환경은 현재의 서버 및 장비의 자원 역량을 축소하여 시뮬레이션함으로써 그 분석의 차이점을 확인하고자 하였다.

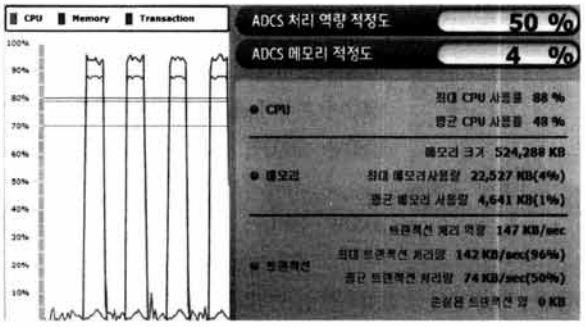
아래 <표 10>의 환경 구성과 시뮬레이션 결과에서 보는 바와 같이 조건을 다르게 함에 따라 Task 처리 적정도와 메모리의 사용율에 대한 결과가 적절하게 반영되어 검침 주기 별로 각각의 시스템에 예상했던 트래픽이 다르게 발생함을 확인하였고 이를 통해 실제 AMI 환경에서의 검침 Task가 시뮬레이션 시나리오에 올바르게 반영이 되었음을 알 수 있다. (그림 9) ~ (그림 11)은 시뮬레이터 결과를 사용자에게 리포팅하는 화면의 예 이다.

〈표 10〉 기능 검증 환경 구성 및 결과 데이터

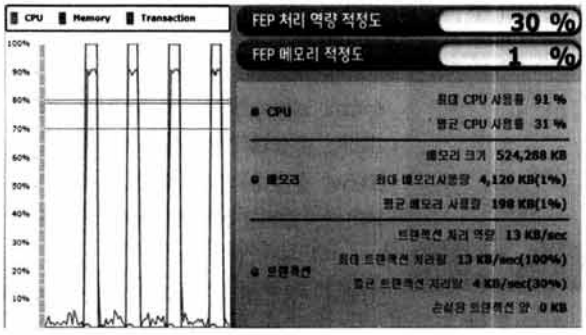
테스트 유형	시스템	CPU 성능 (tpmC)	메모리 버퍼	NIC 대역폭	결과	
					처리적정도	메모리적정도
실 운영 자원 환경	MDMS	360,014	512MB	1,024Mbps	14%	1%
	ADCS	108,435	512MB	1,024Mbps	50%	4%
	FEP	108,435	512MB	1,024Mbps	30%	1%
	Meter	-	-	24Mbps		
부족 자원 환경	MDMS	10,000	128MB	1,024Mbps	228%	100%
	ADCS	10,000	128MB	1,024Mbps	227%	100%
	FEP	65,543	1MB	1,024Mbps	125%	104%
	Meter	-	-	24Mbps		



(그림 9) 정상환경-MDMS



(그림 10) 정상환경-ADCS



(그림 11) 정상환경-FEP

4.2 적정도 적합성 검증

적정도 적합성 테스드는 시뮬레이터를 통하여 도출된 시스템 적정도 도출 값이 실제 운영 환경의 분석 값과 적합한지 확인하는 테스트로서, 서버 시스템 처리 역량 적정도 및 메모리 적정도 적합성 테스트와 DCU 적정도 적합성 테스트를 시행하였다. 아래 환경 구성에서와 같이 각 서버(MDMS, ADCS, FEP)와 같은 서버 환경은 고정하고 DCU와 Meter의 수량을 증가하면서 적정 시스템 운영을 시뮬레이션 하였다.

현재 국내 AMR 시스에서는 DCU 1대당 평균 30개 정도의 미터를 운영하고 있으며, FEP 1대당 314개 정도의 DCU가 설치 운영 중이다. 이런 실제 운영 환경과 비교시 위 <표 12>와 <표 13>의시뮬레이션 결과값과 운영 환경의 분석 데이터와는 유사한 결과를 가져왔으며 시뮬레이터의 적정성이 만족된다. 현재 AMR 운영 환경은 지속적으로 미터가 설치되는 상황이므로 현재의 FEP 운영은 DCU 수량이 720개 정도를 넘어서면 새로운 FEP를 추가함으로써 입출력 부하를 줄여주어야 함을 추측할 수 있는 것이다. FEP외의 서버 자원은 시스템의 자원을 증설함으로써 위와 같은 입출력 부하를 충분히 감당할 수 있지만 DCU의 경우는 모델이 확정된 자원이므로 새로운 성능의 DCU등의 개발을 통하여 수용 가능한 미터의 수량을 확대 할 수 있다고 분석가능하다.

〈표 11〉 적합도 검증 시스템 환경

테스트 식별자	시스템	CPU 성능 (tpmC)	메모리 버퍼	NIC 대역폭
T-PS1 T-PS2 T-PS3 T-PS4	MDMS	360,014	512MB	1,024Mbps
	ADCS	108,435	512MB	1,024Mbps
	FEP	108,435	512MB	1,024Mbps
	DCU	108,435	128MB	24Mbps
	Meter	-	-	24Mbps

〈표 12〉 테스트별 DCU, 미터 수량 변화와 FEP 적정도 분석 결과

시뮬레이션 자원 환경				FEP 적정도 결과	
테스트 식별자	DCU 개수	미터 개수	DCU당 평균 미터 수량	처리 적정도	메모리 적정도
T-PS1	240	8,400	35	30%	5%
T-PS2	480	16,800	35	61%	17%
T-PS3	720	25,200	35	84%	23%
T-PS4	960	33,600	35	100%	40%

〈표 13〉 DCU의 연결 미터 수량 변화와 적정도 분석 결과

결과			DCU 처리 적정도 결과
테스트 식별자	DCU 구분	미터 개수	
T-PD1	D1	20	41%
	D2	30	62%
	D3	40	82%
	D4	50	103%

5. 결 론

현재 국내에서는 국책 과제의 AMI 기술개발 사업과 제주 스마트그리드 실증 단지를 통한 AMI 시범망 구축 등 AMI 도입에 많은 노력을 쏟고 있다. 하지만, AMI 망은 그 전체 규모가 국내 모든 가구를 망라하여 광범위하므로, AMI 전면 도입에 있어서 주요 장비들의 적절한 성능 및 배치를 설계하지 못한다면 시스템 운영에 문제가 있고, 불필요한 자원의 낭비를 초래할 수 있다. 따라서 대규모 AMI 네트워크 및 자원 설계를 쉽게하고, 운영에 필요한 자원의 확보를 위해서는 각 장비의 배치 및 성능에 대한 평가가 가능한 도구가 필요하다. 이에 따라 본 논문에서는 NS-2를 기반으로 한 AMI 네트워크 시뮬레이터를 설계 및 구현을 통하여 실 환경의 적정 네트워크 설계를 위한 적정성을 분석하고자 하였다.