J. Internet Comput. Serv.
ISSN 1598-0170 (Print) / ISSN 2287-1136 (Online)
http://www.jics.or.kr

OMNeT++을 이용한 네크워크 시뮬레이션 기초 가이드

A Basic Guide to Network Simulation Using OMNeT++

박 수 연[1*]
Sooyeon Park

요 약

OMNeT++(Objective Modular Network Testbed in C++)는 네트워크 시뮬레이터를 구축하기 위한 확장 가능하고 모듈화된 C++ 시뮬레이션 라이브러리 및 프레임워크이다. OMNeT++는 센서 네트워크, 인터넷 프로토콜 등 다양한 분야에서 독립적으로 개발된 시뮬레이션 모델을 제공한다. 이를 통해 연구자들은 원하는 시뮬레이션에 필요한 도구와 기능을 사용할 수 있다. OMNeT++는 NED(Network Description) 언어를 사용하여 노드와 네트워크 토폴로지 등을 정의하고, C++ 언어를 통해 정의된 네트워크 객체의 생성과 동작을 구현할 수 있다. 더욱이, INET 프레임워크는 OMNeT++ 시뮬레이션 환경을 위한 오픈 소스 모델 라이브러리로, 다양한 네트워킹 프로토콜과 구성요소에 대한 모델을 포함하고 있어 새로운 네트워크 프로토콜의 설계와 검증에 용이하다. 본 논문은 기초 연구자들을 위해 OMNeT++의 개념과 INET 프레임워크를 활용한 네트워크 시뮬레이션 절차를 설명하여, 이를 통해 다양한 네트워크 시나리오를 모델링하고 분석하는 데 도움을 주고자 한다.

☞ 주제어 : OMNet++ (Objective Modular Network Testbed in C++), NED (Network Description) 언어, INET, 시뮬레이션

ABSTRACT

OMNeT++ (Objective Modular Network Testbed in C++) is an extensible and modular C++ simulation library and framework for building network simulators. OMNeT++ provides simulation models independently developed for various fields, including sensor networks, and Internet protocols. This enables researchers to use the tools and features required for their desired simulations. OMNeT++ uses NED (Network Description) Language to define nodes and network topologies, and it is able to implement the creation and behavior of defined network objects in C++. Moreover, the INET framework is an open-source model library for the OMNeT++ simulation environment, containing models for various networking protocols and components, making it convenient for designing and validating new network protocols. This paper aims to explain the concepts of OMNeT++ and the procedures for network simulation using the INET framework to assist novice researchers in modeling and analyzing various network scenarios.

☞ keyword : OMNet++ (Objective Modular Network Testbed in C++), NED (Network Description) Language, INET, Simulation

1. 서 론

OMNeT++(Objective Modular Network Testbed in C++)는 다양한 네트워크, 멀티 프로세서 및 기타 분산 시스템 환경에서의 시뮬레이션을 쉽게 수행하기 위해 1997년부터 헝가리의 OpenSim사(社)에서 지속적으로 개발되고 있다. OMNeT++는 공개 소스 소프트웨어로, 비영리 용도로 무료로 사용할 수 있는 Academic Public License에 따라 제공된다[1-2].

OMNeT++는 네트워크 시뮬레이터를 구축하기 위한 확장 가능하고 모듈화된 C++ 시뮬레이션 라이이브리 및 프레임워크이다. OMNeT++는 기본적인 INET 프레임워크로 다양한 네트워킹 프로토콜과 구성요소에 대한 모델을 활용할 수 있으며, 다양한 신규 모델을 지속적으로 확장하여 연구자들이 새로운 네트워크 프로토콜의 설계와 검증을 용이하게 시도할 수 있다[3-4].

본 논문에서는 OMNeT++에서 사용되는 언어인 C++와 NED(Network Description)에 대해 개략적으로 설명하고, 간단한 예제를 통해 네트워크 시뮬레이션의 각 단계를 설명한다. 이어서 앞서 보여준 예제에 INET 프레임워크를 활용하여 UDP 환경 등을 적용하는 시뮬레이션 과정을 보여준다. 이러한 내용을 통해 기초 연구자들이 네트워크 시뮬레이션을 효과적으로 활용하고, 다양한 네트워크 환경에서 발생할 수 있는 문제를 탐구하며 해결하는 데 유용한 도구로 삼을 수 있을 것이다.

1 Dept. of IT Convergence Software, Seoul Theological University, Bucheon, 14754, Korea.
* Corresponding author (anisoo@stu.ac.kr)
[Received 24 July 2024, Reviewed 26 July 2024, Accepted 03 August 2024]

본 논문의 구성은 다음과 같다. 2장에서는 OMNeT++의 개요를 소개하고, 3장에서는 OMNeT++의 모델링 언어 및 4장에서는 INET 프레임워크를 다룬다. 마지막으로, 5장에서는 결론을 맺는다.

2. OMNeT++ 개요

본 섹션에서는 OMNeT++의 주요 특징과 구성 요소를 간단히 설명한다. 그리고, 다양한 시뮬레이션 모델 소개 및 간단한 시뮬레이션 수행 과정을 설명한다.

2.1 주요 특징

OMNeT++는 네트워크를 구성 요소 기반으로 모델링하는 방식을 사용한다. 각 구성 요소는 네트워크의 기능을 나타내는 C++ 클래스로 정의되어 실제 네트워크와 유사한 방식으로 시뮬레이션 모델을 구축할 수 있도록 한다. OMNeT++는 모듈러 설계를 통해 사용자가 필요에 따라 기능을 추가하거나 변경할 수 있으며 TCP/IP, UDP, 802.11, LTE 등 다양한 네트워크 프로토콜을 기본적으로 지원한다. 또한 사용자 정의 네트워크 프로토콜 및 모델 정의와 시뮬레이션 결과 시각화 기능을 제공한다.

2.2 구성 요소

OMNeT++는 시뮬레이션 커널 라이브러리(C++), NED, Eclipse 플랫폼 기반 시뮬레이션 IDE, 대화형 시뮬레이션 런타임 GUI(Qtenv; Qt Environment), 시뮬레이션 실행을 위한 명령 프롬프트(Cmdenv; Command-line Environment) 등으로 구성된다[2, 5].

시뮬레이션 커널 라이브러리는 OMNeT++의 핵심 구성 요소로서 시뮬레이션 모델 실행, 이벤트 처리, 데이터 전송, 통계 수집 등 다양한 기능을 제공한다. 이 라이브러리는 사용자가 효율적으로 시뮬레이션 모델을 구현하고 실행하도록 지원한다.

NED는 네트워크 토폴로지를 정의하는 데 사용되는 XML 기반 언어로 네트워크 요소, 연결, 속성 등을 명확하게 정의할 수 있다. NED는 계층적 구조를 통해 복잡한 네트워크 모델을 효율적으로 설계하고 구현할 수 있다.

Eclipse 플랫폼 기반 시뮬레이션 IDE는 OMNeT++를 사용하여 네트워크 시뮬레이션 모델을 설계, 구현, 실행 및 분석하는 통합 개발 환경이다. 이는 다양한 플러그인을 지원하며 시뮬레이션 모델 편집, 코드 완성, 디버깅, 시뮬레이션 실행, 결과 분석 등을 위한 다양한 기능을 제공한다.

Qtenv는 시뮬레이션 모델을 대화형 방식으로 실행하고 분석하는 데 사용되는 GUI이다. 이를 통해 시뮬레이션 모델을 직관적으로 제어하고 결과를 실시간으로 확인할 수 있어 시뮬레이션 모델 개발 및 디버깅 과정을 효율적으로 진행할 수 있다.

Cmdenv는 시뮬레이션 모델을 명령줄을 통해 실행하는 도구이다. 시뮬레이션 결과를 파일로 저장하고 옵션을 설정할 수 있어 시뮬레이션 실행 과정을 자동화할 수 있다.

2.3 시뮬레이션 모델

OMNeT++는 다양한 시뮬레이션 모델 유형을 지원한다[4]. 연구자는 시뮬레이션 모델을 개발할 때 모델링하려는 시스템의 특성에 따라 적절한 시뮬레이션 모델 유형을 선택 할 수 있으며, 여기에서는 주요 대표 모델을 설명한다.

INET 프레임워크[6]는 OMNeT++ 시뮬레이션 환경을 위한 오픈 소스 모델 라이브러리로서 INET에 인터넷 스택, 유무선 링크 계층 프로토콜, 이동성 지원, MANET 프로토콜에 대한 모델 등을 제공한다.

Simu5G(Simulator for 5G New Radio Networks)[7]는 OMNeT++ 및 INET 프레임워크를 위한 5G NewRadio 및 LTE/LTE-A 네트워킹 모델이다. Veins[8]는 차량 네트워크 시뮬레이션을 위한 오픈 소스 프레임워크로, 이벤트 기반 네트워크 시뮬레이터인 OMNeT++와 도로 교통 시뮬레이터인 SUMO[9]를 기반으로 한다. 이 프레임워크는 OMNeT++에 의해 실행되면서 SUMO와 상호작용하여 차량 네트워크와 도로 교통을 시뮬레이션할 수 있다.

2.4 네트워크 시뮬레이션 수행 과정

OMNeT++를 이용한 네트워크 시뮬레이션을 수행하는 주요 과정은 시뮬레이션 모델 정의, 시뮬레이션 실행, 결과 분석의 단계로 진행된다.

첫째, OMNeT++와 함께 제공되는 이클립스 기반의 IDE는 NED 언어를 사용하여 네트워크 토폴로지, 네트워크 객체, 객체의 속성 및 객체 간의 통신 등의 네트워크 모델을 정의한다. 네트워크 객체의 동작인 네트워크 객체의 메시지 전송, 데이터 처리, 이벤트 처리 등을 구현하기 위해 C++ 코드를 활용한다.

둘째, OMNeT++ 시뮬레이션 모델을 실행하기 위한 두 가지 주요 도구는 Cmdenv와 Qtenv이다. Cmdenv는 시뮬

레이션 모델의 이름과 실행 옵션을 입력하여 시뮬레이션을 실행할 수 있으며, Qtenv는 GUI를 통해 실행할 수 있다.

마지막으로, 시뮬레이션 결과 파일은 OMNeT++ IDE를 통해 간단히 살펴볼 수 있다. 그러나 자세한 분석이 필요하다면 Python 기반의 분석 패키지(NumPy, Pandas, Matplotlib)를 다양하게 활용할 수 있다[10].

3. OMNeT++ 모델링 언어

3.1 종류

OMNeT++의 모델링 언어는 원하는 네트워크 모델을 시뮬레이션하는 데 필요한 모든 기능을 제공하기 위해 NED와 C++의 조합으로 구현한다. 예를 들어, 라우터 및 스위치와 같은 다양한 네트워크 객체를 NED로 정의할 수 있다. 여기에, C++는 정의한 네트워크 객체가 메시지를 어떻게 전송하고 처리하는지, 이벤트를 어떻게 처리하는지 등의 동작을 정의할 수 있다. 또한, 시뮬레이션 모델에서 사용되는 데이터 구조를 정의하고, 시뮬레이션 모델에서 사용되는 알고리즘을 구현할 수 있다.

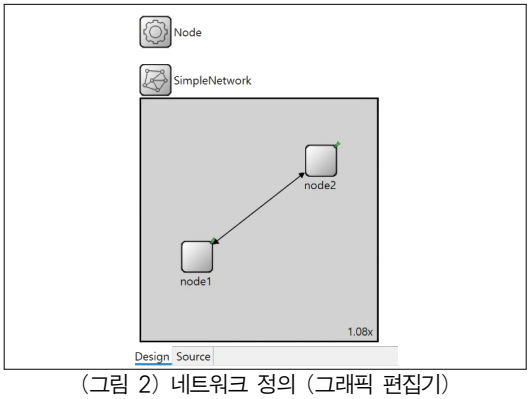
3.2 NED 예제

NED 편집기는 텍스트 편집기(Source)와 그래픽 편집기(Design) 가 있다. 텍스트 편집기에서 작성한 NED 소스는 그림과 아이콘 형태로 그래픽 편집기에 표시된다. 그림 1은 텍스트 편집기에서 NED를 이용하여 네트워크 객체를 만드는 과정이다. SimpleNetwork라는 네트워크를 정의하고, 이 네트워크는 node1과 node2라는 두 개의 노드를 포함하고 있다. 여기에서 노드의 속성은 간단히 입/출력 포트만을 표현하였으나, 진행하는 연구에 적절하게 객체의 특성 등을 추가할 수 있다.

그림 1은 node1의 출력 포트를 node2의 입력 포트에 연결하고, node2의 출력 포트를 node1의 입력 포트에 연결한다. 다음 절에서는 연결된 두 개의 노드를 이용하여 데이터를 송수신할 것이다. 그림 1에서 정의된 내용을 그림 2와 같이 그래픽 편집기에서도 확인할 수 있다. Node에 대한 아이콘으로 속성을 살펴볼 수 있고, 정의된 SimpleNetwork 환경인 node1과 node2의 유선 연결을 직접 확인할 수 있다. 반대로 그래픽 편집기에서의 작업으로 텍스트 편집기로의 반영 또한 가능하다.

```
simple Node
{
    gates:
        input in;
        output out;
}
network SimpleNetwork
{
    submodules:
        node1: Node;
        node2: Node;
    connections:
        node1.out --> node2.in;
        node1.in <-- node2.out;
}
```

(그림 1) 네트워크 정의 (텍스트 편집기)
(Figure 1) Network Definition (Source)



(그림 2) 네트워크 정의 (그래픽 편집기)
(Figure 2) Network Definition (Design)

3.3 C++ 예제

그림 1에서 정의된 노드를 활용하여, 그림 3 및 4는 두 노드 간의 메시지를 주고받는 간단한 C++ 코드의 예를 보여준다. 그림 3은 Node.h 헤더 파일이고, 그림 4는 Node.cpp 소스 파일이다. 그림 3은 Node라는 클래스를 정의하고 있으며, 그림 4는 Define_Module(Node)를 이용하여 NED로 정의한 Node를 OMNeT++의 모듈로 등록한다.

그림 4의 initialize() 함수에서는 Node의 초기화 작업이 수행된다. 시뮬레이션이 실행되면, 모든 Node(node1, node2)들에서 초기화가 수행되며, 여기에서는 node1이 메시지를 전송한다. 즉, 노드의 이름이 node1이 확인되는 Node는 node1이고, 그 후 TestMessage를 생성하고 node1의 out 포트로 메시지를 전송한다.

```
#include <omnetpp.h>
#include <string.h>
using namespace omnetpp;
class Node : public cSimpleModule
{
protected:
  virtual void initialize() override;
  virtual void handleMessage(cMessage *msg) override;
};
```

(그림 3) Node.h
(Figure 3) Node.h

```
#include "Node.h"          // Fig. 3
Define_Module(Node);       // Fig. 1
void Node::initialize()
{
   if (strcmp("node1", getName()) == 0) {
      // node1에서 메시지 생성 및 전송
      cMessage *msg =
                  new cMessage("TestMessage");
      send(msg, "out");
    }
}
void Node::handleMessage(cMessage *msg)
{
    // 메시지 처리 코드
    EV << "Received message: " << msg->getName()
       << " at " << getName() << endl;
    send(msg, "out");
}
```

(그림 4) Node.cpp
(Figure 4) Node.cpp

```
[General]
network = SimpleNetwork
```

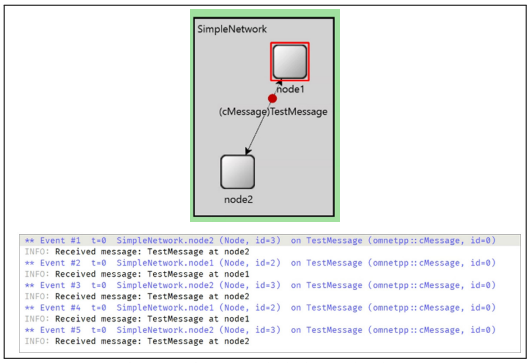
(그림 5) 첫 번째 시뮬레이션을 위한 omnetpp.ini
(Figure 5) omnetpp.ini for 1st Simulation

시뮬레이션을 실행하기 위해 그림 5와 같이 omnetpp.ini 파일을 생성하며, 그림 1에서 정의한 SimpleNetwork를 등록한다. NED로 node1과 node2의 연결을 정의하였으므로, 전송된 메시지는 node2로 향하게 된다.

그림 4의 handleMessage 함수에서는 수신한 메시지가 처리된다. node1에서 전송된 메시지는 node2에서 수신하게 되며, 여기에서는 그 즉시 out 포트로 전송하게 설계하였다. 즉, node2에서 수신된 메시지는 node1으로 전송된다. 정리하면, node1과 node2가 서로 메시지를 주고받는 모습을 시뮬레이션으로 디자인하였다.

3.4 시뮬레이션 실행 모습

그림 1 내지 5를 작성하고, IDE의 메뉴에서 실행(Run)을 수행하면, 시뮬레이션이 실행되는 그림 6의 모습을 관찰할 수 있다. 계획한 바와 같이 두 개의 노드 node1과 node2가 메시지를 서로 주고받으며, 각 노드는 메시지를 수신할 때마다 그림 4에서 코딩한 문자열을 출력한다.



(그림 6) 시뮬레이션 실행 결과
(Figure 6) Simulation Execution Results

4. INET 프레임워크 소개

INET 프레임워크는 OMNeT++ 시뮬레이션 환경에 대한 오픈 소스 모델 라이브러리이다. INET 프레임워크는 유·무선 및 모바일 네트워크를 포함한 다양한 네트워크 시나리오를 모델링하고 시뮬레이션하는 데 사용된다. 또한 INET 프레임워크를 확장하여 연구하고자 하는 네트워크 모델 설계 및 다양한 기능을 제공할 수 있다[11-12].

그림 6과 같이 동작하는 네트워크 시뮬레이션에 특정 프로토콜(Ethernet, UDP)을 추가하기 위해서 그림 7 및 8처럼 INET 프레임워크를 사용할 수 있다. 그림 1의 SimpleNetwork와 같은 환경을 정의하기 위해 그림 7에서는 WiredNetwork로 셋팅한다. 일반적으로 @display()를 사용하여 그래픽 환경에 렌더링하기 위한 다양한 속성들을 활용할 수 있다[13]. 네트워크의 크기를 가로 638m, 세로 256m로 설정하고, 두 개의 노드(node1, node2) 위치를 각각 독립적으로 지정할 수 있다. 각 노드는 StandardHost의 기능을 갖도록 하여 TCP, SCTP, UDP 등의 통신 프로토콜을 설정할 수 있다[14]. Ipv4NetworkConfigurator를 통해 node1과 node2에 각각 IPv4 주소가 할당되며, 두 노드 간의 통신을 위해 정적 라우팅이 설정된다. 이더넷으로 통신하기 위해, 각 노드의 이더넷 인터페이스를 증가시키

고(ethg++), 10Mbps의 속도를 제공하는 이더넷 링크(Eth10M)로 연결한다.

```
import inet.node.inet.StandardHost;
import inet.networklayer.configurator.ipv4.Ipv4NetworkConfigurator;
import inet.node.ethernet.Eth10M;

network WiredNetwork
{
    @display("bgb=638, 256");
    submodules:
        node1: StandardHost {
            @display("p=111, 155");
        }
        node2: StandardHost {
            @display("p=480, 128");
        }
        configurator: Ipv4NetworkConfigurator {
            @display("is=s; p=80,69"); // is: icon size
        }
    connections:
        node1.ethg++ <--> Eth10M <--> node2.ethg++;
}
```

(그림 7) InetNetwork.ned (텍스트 편집기)
(Figure 7) InetNetwork.ned (Source)

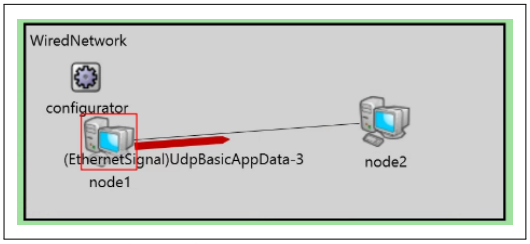
시뮬레이션을 실행하기 위해 그림 8과 같이 omnetpp.ini 파일을 생성하며, 그림 7에서 정의한 Wired Network를 등록한다. '**'는 import한 StandardHost의 계층구조를 표현하는 것으로 두 노드의 속성을 지정할 수 있다. numApps로 1개의 UDP 기반 어플리케이션(app[0])으로 한정한다. 송신자 node1은 UdpBasicApp, 수신자 node2는 UdpSink로 지정하며, 이들은 inet.applications. udpapp 패키지에 정의되어 있다. node1의 목적지는 node2로 설정하고, 송신할 메시지의 크기는 100B, 전송 분포는 고정값 100ms이다. 수신측의 포트는 5001번으로 송·수신시 모두 지정한다.

```
[General]
network = WiredNetwork

**.node1.numApps = 1
**.node1.app[0].typename = "UdpBasicApp"
**.node1.app[0].destAddresses = "node2"
**.node1.app[0].messageLength = 100B
**.node1.app[0].sendInterval = 100ms
**.node1.app[0].destPort = 5001

**.node2.numApps = 1
**.node2.app[0].typename = "UdpSink"
**.node2.app[0].localPort = 5001
```

(그림 8) 두 번째 시뮬레이션을 위한 omnetpp.ini
(Figure 8) omnetpp.ini for 2nd Simulation



(그림 9) 시뮬레이션 실행 결과
(Figure 9) 시뮬레이션 실행 결과

그림 7 및 8을 작성하고, IDE의 메뉴에서 실행(Run)을 수행하면, 시뮬레이션이 실행되는 그림 9의 모습을 관찰할 수 있다. 계획된 바와 같이 node1에서 UDP 기반의 어플리케이션이 메시지를 생성한 후, 이더넷 링크를 통해 node2로 전송한다. 상기 메시지는 node2의 UdpSink 애플리케이션에 수신된다. (사실 그림 9는 단방향 통신이며, 그림 6과 같이 양방향 통신을 위한 모델은 예제의 간결함을 위해 생략하였다.)

지금까지 간단한 UDP 통신 예제를 INET 프레임워크를 통해 시뮬레이션 해 보았다. 이러한 방법으로 INET 프레임워크를 이용하여 다양한 통신 모델을 설정 또는 확장 시킬 수 있다.

5. 결 론

OMNeT++는 네트워크 시뮬레이션을 위한 오픈 소스 시뮬레이션 환경으로, 다양한 네트워크 시나리오를 모델링하고 시뮬레이션하는 데 사용된다. NED 모델링 언어를 사용하여 네트워크 모델을 정의하고, OMNeT++ 시뮬레이션 엔진을 통해 모델을 실행하고 결과를 분석할 수 있다. OMNeT++는 유·무선, 모바일 네트워크를 포함한 여러 네트워크 환경을 지원한다.

OMNeT++는 INET 프레임워크를 활용하여 다양한 네트워크 프로토콜, 단말, 어플리케이션 등을 모델링하는 강력한 기능을 제공하며, 사용자 정의 모델을 개발하고 기존 모델을 확장할 수 있는 유연성을 갖추고 있다. 또한, 다양한 플랫폼에서 실행될 수 있으며, 다른 시뮬레이션 도구 및 라이브러리와 연동될 수 있다.

본 연구에서는 연구자를 위하여 OMNeT++ 시뮬레이션의 기초 소개 및 간단한 예제를 제공하였다. 향후에는 INET 프레임워크를 활용한 다양한 모델에 대해 소개할 예정이다.

참고문헌(Reference)

[1] A. Varga and R. Hornig, “An Overview of the OMNeT++ Simulation Environment,” ICST, 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops, Article No. 60, pp. 1-10, March 2008. https://dl.acm.org/doi/10.5555/1416222.1416290

[2] Retrieved from https://omnetpp.org/intro (August 2024)

[3] Virdis and M. Kirsche, “Recent Advances in Network Simulation: The OMNeT++ Environment and Its Ecosystem,” p. 55, Switzerland: Springer, 2019. https://link.springer.com/book/10.1007/978-3-030-12842-5

[4] Retrieved from https://omnetpp.org/download/models-and-tools (August 2024)

[5] Retrieved from https://omnetpp.org/documentation/ (August 2024)

[6] Retrieved from https://inet.omnetpp.org/Introduction.html (August 2024)

[7] Retrieved from https://simu5g.org/ (August 2024)

[8] Retrieved from https://veins.car2x.org/documentation/ (August 2024)

[9] Retrieved from https://eclipse.dev/sumo/ (August 2024)

[10] Retrieved from https://docs.omnetpp.org/tutorials/pandas/ (August 2024)

[11] Retrieved from https://omnetpp.org/download-items/INET.html (August 2024)

[12] Retrieved from https://inet.omnetpp.org/ (August 2024)

[13] Retrieved from https://doc.omnetpp.org/omnetpp/manual/ (August 2024)

[14] Retrieved from https://doc.omnetpp.org/inet/api-current/neddoc/inet.node.inet.StandardHost.html (August 2024)

◐ 저 자 소 개 ◑

박 수 연(Sooyeon Park)

2004년 한국공학대학교 컴퓨터공학과 (공학사)
2007년 한국공학대학교 컴퓨터공학과 (공학석사)
2011년 한국공학대학교 컴퓨터공학과 (공학박사)
2011년~2015년 전자부품연구원 연구원
2024년~현재 서울신학대학교 IT융합소프트웨어학과 초빙교수
관심분야 : 지능형 모바일 센서 네트워킹, 머신러닝 및 인공지능, 시뮬레이션, 데이터베이스 등
E-mail : anisoo@stu.ac.kr