PDF Download
2048416.2048429.pdf
24 February 2026
Total Citations: 0
Total Downloads: 168

Published: 03 April 2011

Citation in BibTeX format

SpringSim '11: 2011 Spring Simulation Multi-conference
April 3 - 7, 2011
Massachusetts, Boston

RESEARCH-ARTICLE

HMNToolSuite: tool support for mobility management of mobile devices in heterogeneous mobile networks

JOONMYUNG KANG, Pohang University of Science and Technology, Pohang, Kyongsangbuk-do, South Korea

SINSEOK SEO, Pohang University of Science and Technology, Pohang, Kyongsangbuk-do, South Korea

JOHN C STRASSNER, Pohang University of Science and Technology, Pohang, Kyongsangbuk-do, South Korea

JAMES WON KI HONG, Pohang University of Science and Technology, Pohang, Kyongsangbuk-do, South Korea

HMNToolSuite: Tool Support for Mobility Management of Mobile Devices in Heterogeneous Mobile Networks

Joon-Myung Kang and Sin-seok Seo
Dept. of Computer Science and Engineering
Pohang University of Science and Technology
(POSTECH), Pohang, South Korea
{eliot, sesise}@postech.ac.kr

John Strassner and James Won-Ki Hong
Div. of IT Convergence Engineering
Pohang University of Science and Technology
(POSTECH), Pohang, South Korea
{johns, jwkhong}@postech.ac.kr

Keywords: Mobile Device Management, Mobility Management, Heterogeneous Mobile Networks

Abstract

This paper presents tool support for testing mobility of mobile devices in heterogeneous mobile networks. As mobile devices are growing and networks are becoming heterogeneous, their mobility management in heterogeneous mobile networks has become important. Nonetheless, previous network simulators have focused on handover protocols at layer 2 or layer 3, but have not focused on handover decisions at layer 7. This tool suite allows the user to create multiple types of mobile networks, mobile nodes, and network servers for testing mobility of mobile devices. Moreover, it also allows the user to create simulation scenarios and generates testing results based on users' demands. This paper presents the requirements, design, and implementation of the tool suite. We show the feasibility of our tool using a case study of context-aware handover decision management.

1. INTRODUCTION

Recently, the evolution of various mobile networking technologies and network interfaces for supporting them has been leading services using multiple networks in mobile devices [1]. In the future, these networks are likely to support flexible and personalized handover decisions by dedicated devices to satisfy the demands of the end user based on context information [2–5]. When we develop handover decision algorithms for mobile devices in Heterogeneous Mobile Networks (HMNs), we need emulation and simulation techniques to validate them because it is still difficult to deploy these devices in such networks.

Existing network simulators focus on layer 2 or layer 3 network protocol implementation, but do not address handover decisions for mobility management in HMNs. While the performance of such low level protocols, such as mobile IP or Media Independent Handover (MIH) [6], is important, we need to focus on higher-level aspects such as handover decisions or access network selection. Therefore, we need to develop a more flexible and user-friendly tool in order to test mobility of mobile devices in HMNs.

In this paper, we present a tool suite for emulating and simulating mobile devices in HMNs in order to test mobility. We call it *HMNToolSuite*, which stands for Heterogeneous Mobile Network Tool Suite. It allows the user to create HMNs, different types of mobile nodes, and network servers with their own characteristics. It also allows the user to create various simulation scenarios using customized network maps. To show the feasibility of the tool suite, we present a cast study of context-aware handover decision management of mobile devices in HMNs. This suite is currently available as an *open source project* [7] to encourage other researchers to use our tool on testing mobility management functions.

The remainder of this paper is organized as follows. We describe existing network simulators and their limitations in Section 2. We present the requirements for developing new simulation tools in Section 3. We then describe the design and implementation of our HMNToolSuite in Section 4. We validate our tool with a case study in Section 5. Finally, we draw conclusions and discuss future work in Section 6.

2. RELATED WORK

NetSim [8] is a network simulation tool used by the academic community for teaching, network lab experimentation and research. It supports various technologies such as Ethernet, Wireless LAN, Wi Max, TCP, IP, etc. However, it cannot be extended to address modern networks because it has not been extended since the initial version. NS-2 [9] is a discrete event network simulator, which is widely used in the simulation of routing protocols, among others, and is heavily used in ad-hoc networking research. It supports popular network protocols. It was built in C++ and provides a simulation interface through OTcl. NS-3 [10] is a discrete-event network simulator for Internet systems, targeted primarily for research and educational use. It is intended as an eventual replacement for the popular NS-2 simulator. OPNET Modeler

(Optimized Network Engineering Tool) [11] provides a comprehensive development environment for the specification, simulation and performance analysis of communication networks. It supports the modeling and simulation cycles, hierarchical modeling, specialized in communication networks, and automatic simulation generation. OMNet++ [12] is a C++-based discrete event simulator for modeling communication networks, multiprocessors and other distributed or parallel systems. It was a powerful open-source discrete event simulation tool that can be used by academic, educational and research-oriented commercial institutions for the simulation of computer networks and distributed or parallel systems. J-Sim [13] is a component-based, compositional simulation environment, implemented in Java. J-Sim is similar to OMNet++ in that simulation models are hierarchical and built from self-contained components, but the approach of assembling components into models is more like NS-2 [12].

However, all these network simulators focus on layer 2 or layer 3 network protocols, network traffic, mobile nodes, and MAC issues. Our HMNToolSuite focuses on management of mobile devices in HMNs at layer 7 such as network interface selection.

3. REQUIREMENTS

HMNToolSuite was designed to support In this section, we present requirements for developing our HMNToolSuite. We surveyed the characteristics of mobile devices and HMNs and general functions of emulators and simulators. We also surveyed management issues related to mobile devices in HMNs, especially mobility management. We summarized the requirements as follows:

- **Heterogeneous Networks Modeling:** The tool should provide to create, modify, and delete multiple kinds of mobile networks, mobile nodes, and network servers. It should provide to specify their characteristics.
- **Mobile Device:** The tool should provide to create, modify, and delete mobile devices with their own functions.
- **Network Traffic:** The tool should provide a time-series network traffic for each application between a mobile node and an application server via mobile networks.
- **Handover Decisions:** The tool should provide a framework for implementing handover decision algorithms for mobility management in mobile nodes and show how to use context information.
- **Scalability:** The tool should provide to run simulations with a large number of nodes in a reasonable amount of time.
- **Flexibility:** The user should provide to specify relevant simulation parameters, policies, network statuses, and services in a human readable configuration file or a GUI-based configuration manager.
- **Reuse of Simulation Code:** The provided implementation of handover decisions should be reusable for real network applications enabling researchers to validate the simulation results by comparing them to the results from real-world test networks.
- **Statistics:** The tool should collect statistical data. The output should be in a format that is easy to post-process (e.g., for generating gnuplot-based graphs).
- **Interactive Visualizer:** In order to validate and debug new or existing management functions, the tool should provide a GUI, which can visualize the network map, access networks, servers, and mobile nodes in a customizable way.

4. DESIGN AND IMPLEMENTATION

In this section, we present the design and implementation of the HMNToolSuite. First, we describe the overall structure of our tool. Then, we present the design of mobile devices. Finally, we present the implementation details.

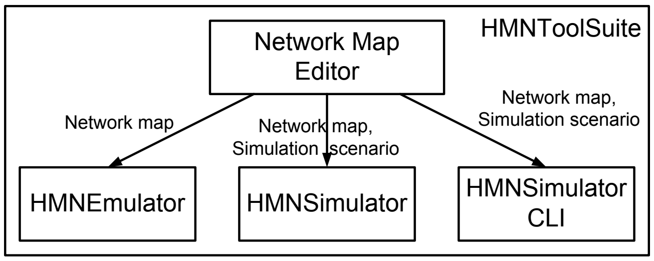


Figure 1: Overall architecture of the HMNToolSuite

4.1. Design of the HMNToolSuite

Figure 1 shows an overall architecture of the HMNToolSuite which is composed of Network Map Editor, HMNEmulator, HMNSimulator, and HMNSimulatorCLI. Network Map Editor provides to create a network map for emulating and simulating. We create mobile networks, mobile nodes, and network servers with customized parameters. We also specify the moving path and velocity of each mobile node and time-series network traffic information of each application traffic between a network server and a mobile node via a mobile network. A network map is composed of mobile networks, mobile nodes, and network servers. It includes coordinations and time-series parameters of mobile networks and mobile nodes for testing mobility. We define each mobile network with following parameters: a name, an access device, a coverage, a bandwidth, a delay, a jitter, a bit error rate, a

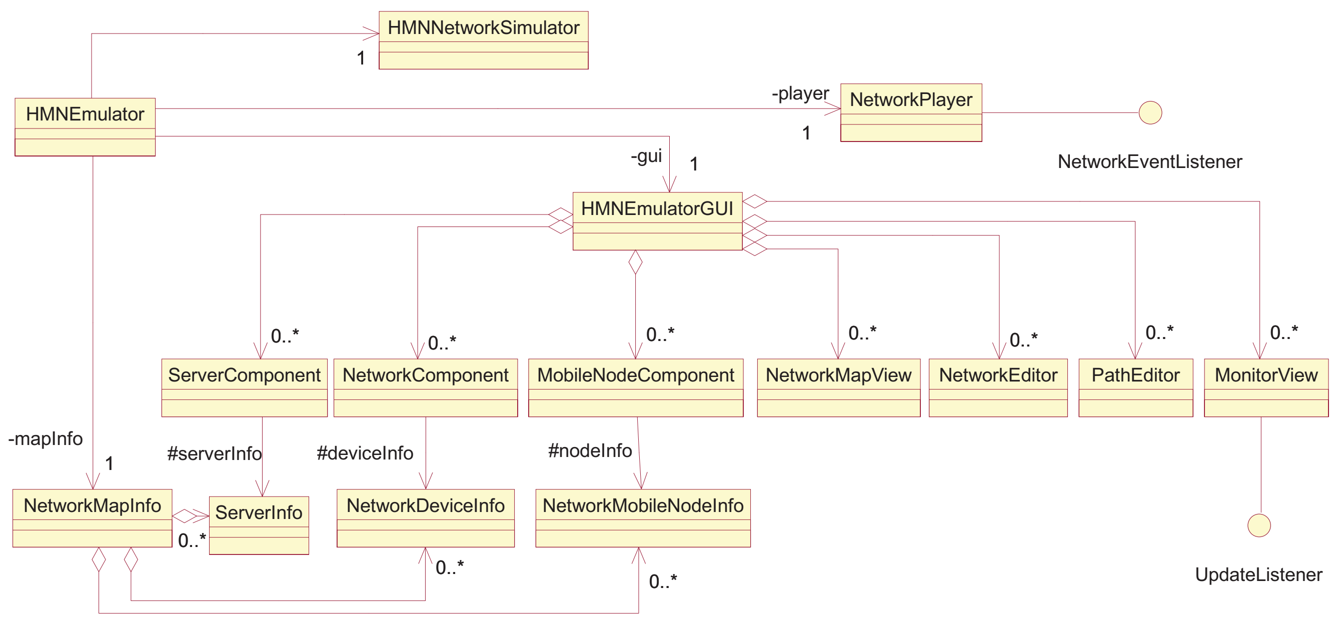


Figure 2: Overall class diagram of the HMNToolSuite

throughput, a burst error, a packet loss ratio, cost rate, transmitter power consumption, receiver power consumption, idle power consumption, minimum supported velocity, maximum supported velocity, and a color value. We also define each mobile node with following parameters: a name, an icon, a color, a maximum velocity, a minimum velocity, a supported application list, and a supported network interface list. Finally, we define each network server with following parameters: a name, an icon, and a server type.

HMNEmulator emulates how networks and mobile nodes function in the environment represented by a network map. We move each mobile node for testing mobility and see visualized results on a monitor view such as connected and unconnected mobile nodes, the current velocity and direction of a mobile node, the number and type of handovers for each mobile node, the available network(s) and their associated metrics (i.e, RSS and quality), and application status.

HMNSimulator simulates networks and mobile nodes based on a simulation scenario created by a network map editor. It shows a network status, a mobile node status, a handover status, a simulation performance using graphs, and the number of handovers, velocity, and connected time of each mobile node in the network map. HMNSimulatorCLI simulates a network map on the Command Line Interface (CLI) without a GUI.

Figure 2 is the main class diagram of our tool. We used the *Model-View-Controller (MVC)* pattern for this design, which is a well-known software pattern for separating the modeling of the domain from the presentation and the actions taken in the domain based on user input [14].

The HMNEmulator class is the main class responsible for executing the emulation tasks. It is built with no concern for how it will "look and feel" when presented to the user. It has a purely functional interface, meaning that it has a set of public functions that can be used to execute all of its functionality. It provides interfaces to access the NetworkMapInfo data classes. The NetworkMapInfo has three types of data: a) NetworkDeviceInfo, b) NetworkMobileNodeInfo, and c) ServerInfo. The NetworkDeviceInfo class provides information related to network devices, such as type of BS or AP, and their characteristics. It also provides network information, such as whether it is CDMA, WLAN, or Mobile WiMax. The NetworkMobileNodeInfo class provides information related to how the mobile node is moving, such as by walking, or using a car or bus. It also provides mobile device information, such as the set of applications and network interfaces that it supports.

The HMNEmulator supports several different views of the network map, including HMNEmulatorGUI, HMNNetworkSimulator, MonitorView, and CLI view. The HMNEmulatorGUI shows the current emulating environment. We also create emulating scenarios using this GUI. It has ServerComponents, NetworkComponents, MobileNodeComponents, a NetworkMapView, a NetworkEditor, a PathEditor, and a MonitorView. The ServerComponent, NetworkComponent, and MobileNodeComponent have their specific icons and information defined using the ServerInfo, NetworkDeviceInfo, and NetworkMobileNodeInfo information classes, respectively. The NetworkMapView shows the current network map and all components. The PathEditor enables us to create and modify paths for each mobile node. The MonitorView provides a current snapshot of all components by monitoring them periodically and displaying them on demand. Finally, the NetworkPlayer generates network events for transferring packets between network devices, mobile devices, and servers in the network map.

4.2. Design of Mobile Device

Mobile devices have many functions, but we focus only on their network interfaces and applications that are major modules for testing mobility in mobile devices.

We used *Feature Oriented Software Development (FOSD)* [15] for creating mobile devices in our tool. FOSD is a general paradigm for program synthesis in *Software Product Lines (SPL)*, which is is a set of software-intensive systems sharing a common, managed set of features that satisfy the specific need of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way [16]. The purpose of FOSD is to define a discrete and enumerable space of configurations for creating SPL using a feature model. Features are increments in functionality, and thus are the building blocks of programs. Each configuration, which is a composition of features, represents a program.

The domain of a mobile device is appropriate to applying an SPL because mobile devices have many commonalities and variabilities in their functionality. We present a feature model for creating a mobile device product line. In a mobile device product line, a network interface, an application, a user policy, and a monitor are examples of variation points. These variation points have variants called *features* in a feature model.

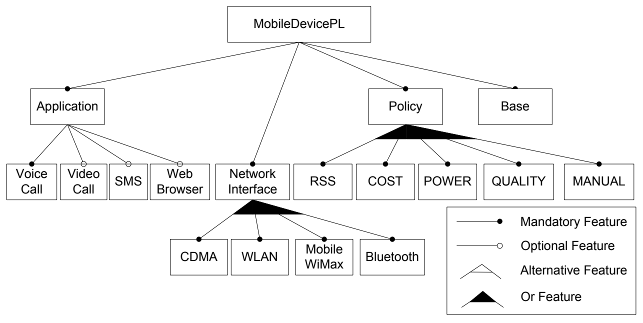


Figure 3: An example of a feature model for a mobile device product line

Figure 3 shows an example of a feature model for a mobile device product line. By using the given model, new mobile devices can quickly be developed by emphasizing their commonalities with and variabilities between existing product features.

Mobile devices can be represented using the notations described in [15] as follows:

$$\begin{aligned}
\mathbf{PL} &= \{p_1, p_2, \ldots, p_m\} \\
p_1 &= Application \bullet NetworkInterface \bullet Policy \\
&= VoiceCall \bullet SMS \bullet CDMA \bullet WLAN \bullet MobileWiMax \bullet \\
&\quad RSS \bullet COST \\
p_2 &= VoiceCall \bullet SMS \bullet VideoCall \bullet CDMA \bullet WLAN \bullet \\
&\quad POWER \\
&\vdots \\
p_m &= VoiceCall \bullet VideoCall \bullet SMS \bullet WebBrowser \bullet \\
&\quad CDMA \bullet WLAN \bullet MobileWiMax \bullet Bluetoth \bullet \\
&\quad RSS \bullet COST \bullet POWER \bullet QUALITY
\end{aligned} \tag{1}$$

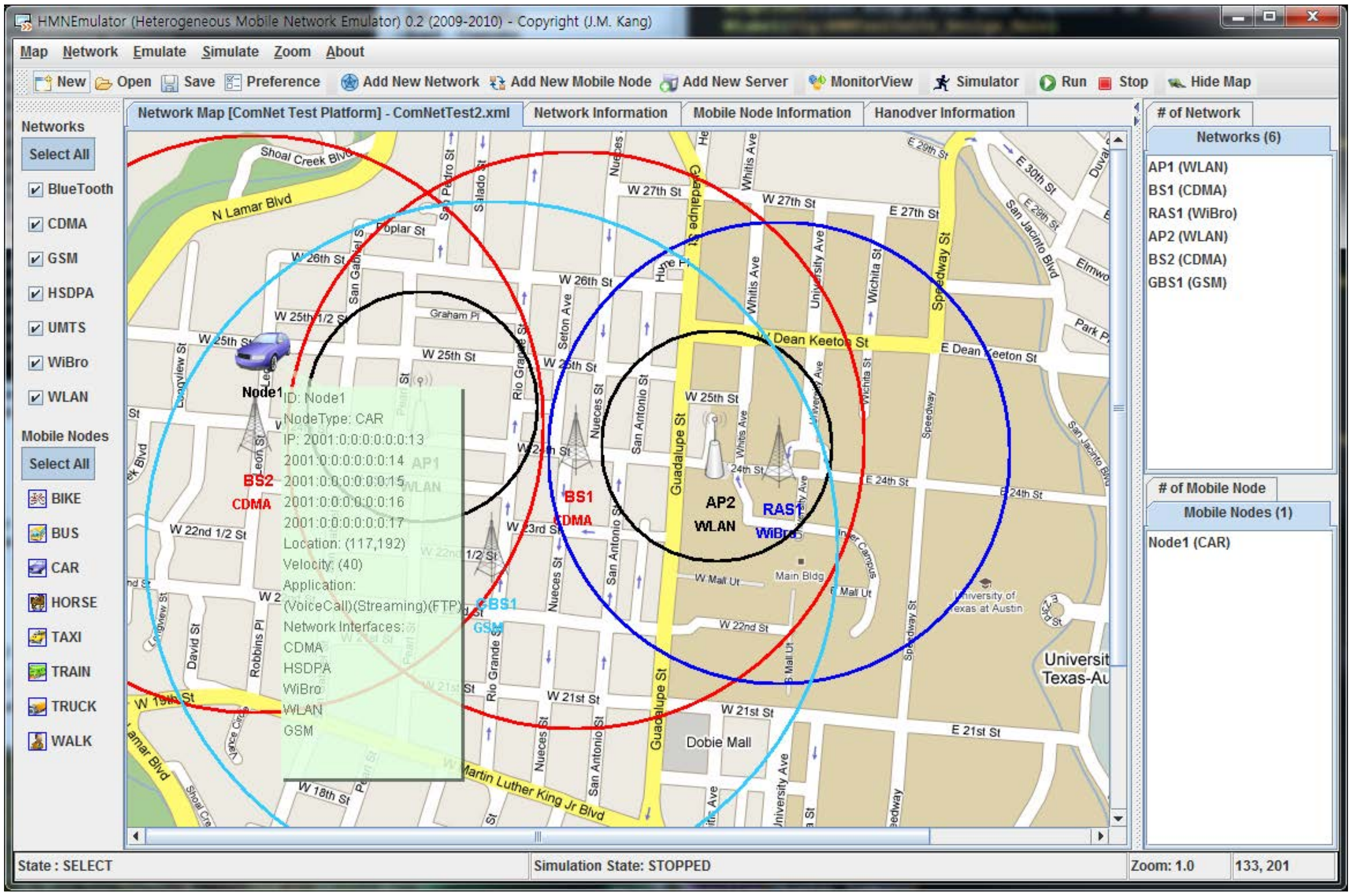
, where *PL* is a set of mobile device specified by the feature model and p_i is a mobile device configured statically ($1 \geq i \geq m$). For example, the mobile device p_1 has *Application*, *NetworkInterface*, and *Policy* features. As applications, it has *VoiceCall*, and *SMS*. It also has network interfaces which support *CDMA*, *WLAN*, and *MobileWiMax* communications and user policies with *RSS* and *COST*.

4.3. Implementation of the HMNToolSuite

In this section, we present implementation details of our HMNToolSuite. We implemented our HMNToolSuite using the Java programming language (Java platform standard edition, JDK 6). We used an XML parser in the javax.xml.parsers package for creating and editing configuration files of a network map, a network, a mobile node, and a network server. We also used jFreeChart [17] for displaying performance result graphs. Finally, we used Java Swing for constructing a GUI.

We describe our tool using some screenshots. Figure 4 shows a screenshot of our HMNToolSuite based on all requirements and designs mentioned in Section 3 and 4. First, we create a network map with a given map image and size. Then, we create new access networks by selecting the "Add New Network" command button from the menu palette or the menu bar. This creates a dialog box that prompts the developer to assign some important characteristics for the access network. Then, the developer clicks on the network map where it is to be positioned. Each access network has different predefined characteristics, but our tool enables the developer to configure each parameter for each supported network type. Thus, we can modify the existing access network configuration, or create a new one. In Figure 4, the circles with the different colors represent different access networks. For example, the red circle represents a CDMA access network.

Then, we create a mobile node by selecting the "Add New Mobile Node" command button from the menu palette. This creates a dialog box that prompts the developer to assign


HMNEmulator (Heterogeneous Mobile Network Emulator) 0.2 (2009-2010) - Copyright (J.M. Kang)
Map Network Emulate Simulate Zoom About
New Open Save Preference Add New Network Add New Mobile Node Add New Server MonitorView Simulator Run Stop Hide Map
Network Map [ComNet Test Platform] - ComNetTest2.xml
Network Information
Mobile Node Information
Hanodver Information
Networks
Select All
BlueTooth
CDMA
GSM
HSDPA
UMTS
WiBro
WLAN
Mobile Nodes
Select All
BIKE
BUS
CAR
HORSE
TAXI
TRAIN
TRUCK
WALK
BS2 CDMA
BS1 CDMA
AP2 WLAN
RAS1 WiBro
GBS1 GSM
Node1
# of Network
Networks (6)
# of Mobile Node
Mobile Nodes (1)

ID: Node1
NodeType: CAR
IP: 2001:0:0:0:0:0:0:13
2001:0:0:0:0:0:0:14
2001:0:0:0:0:0:0:15
2001:0:0:0:0:0:0:16
2001:0:0:0:0:0:0:17
Location: (117,192)
Velocity: (40)
Application:
(VoiceCall)(Streaming)(FTP)
Network Interfaces:
CDMA
HSDPA
WiBro
WLAN
GSM

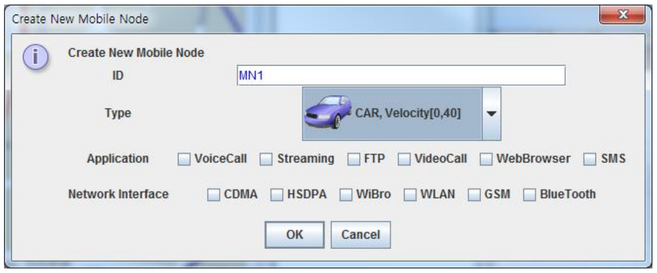
AP1 (WLAN)
BS1 (CDMA)
RAS1 (WiBro)
AP2 (WLAN)
BS2 (CDMA)
GBS1 (GSM)

Node1 (CAR)

State : SELECT
Simulation State: STOPPED
Zoom: 1.0
133, 201

Figure 4: A screenshot of the HMNToolSuite

some important characteristics for the mobile node, as shown in Figure 5. Then, the developer clicks on the network map where the node is to be positioned.


Create New Mobile Node

Create New Mobile Node

ID: MN1

Type: CAR, Velocity[0,40]

Application: ☐ VoiceCall ☐ Streaming ☐ FTP ☐ VideoCall ☐ WebBrowser ☐ SMS

Network Interface: ☐ CDMA ☐ HSDPA ☐ WiBro ☐ WLAN ☐ GSM ☐ BlueTooth

OK Cancel

Figure 5: A screenshot of a mobile node creator of the HMN-ToolSuite

As we mentioned earlier, we developed the mobile node creator using FOSD. We can create various mobile nodes, which have different types of applications and network interfaces. When we create a mobile node, we determine possible network interfaces to connect for mobile services. For example, if we select CDMA, HSDPA, and Mobile WiMax access networks for a mobile device, the mobile device can only access those three types of access networks. We can modify the configuration or create a new one to meet the needs of the scenario being simulated.

5. CASE STUDY: CONTEXT-AWARE HANDOVER DECISION MANAGEMENT

In this section, we present a case study for showing the feasibility of our HMNToolSuite. We implement six handover decision algorithms and compare their performance using our tool. We show how to create an experimental scenario and get results for our previous research [5, 18].

In the experiment for [18], we wanted to compare our handover decision algorithm with other handover decision algorithms. Our algorithm provides handover decisions for a mobile device based on context information, application requirements, and characteristics of mobile networks to satisfy end users' demands. First, we constructed a heterogeneous mobile network environment, as illustrated in Figure 6, using the HMNToolSuite. Then, we created a mobile device that sup-

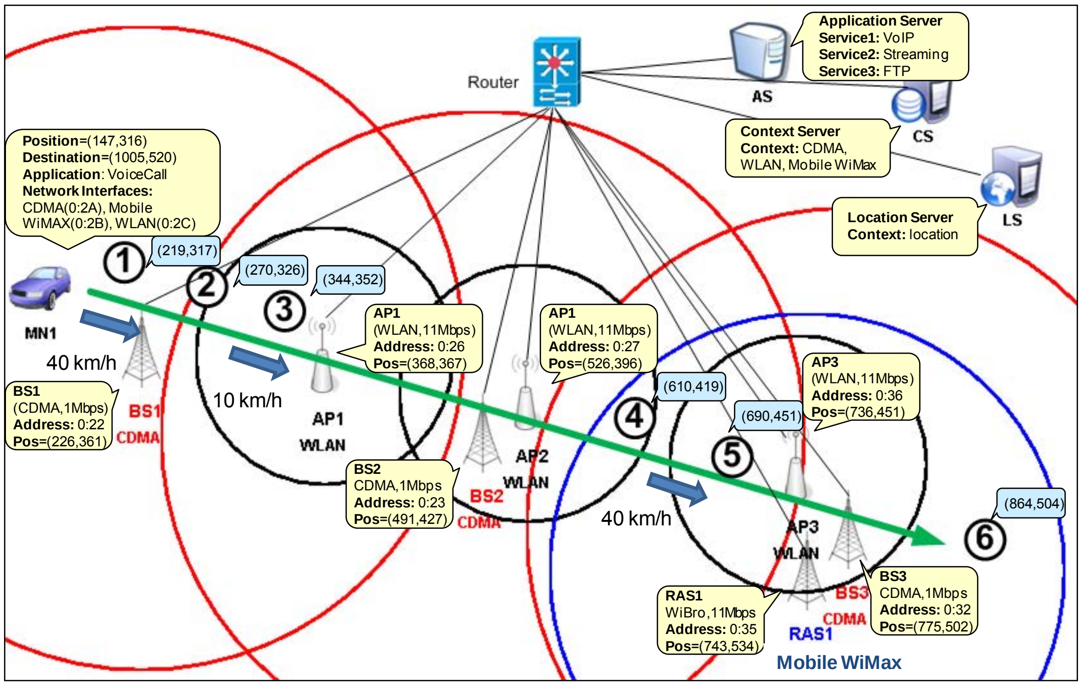
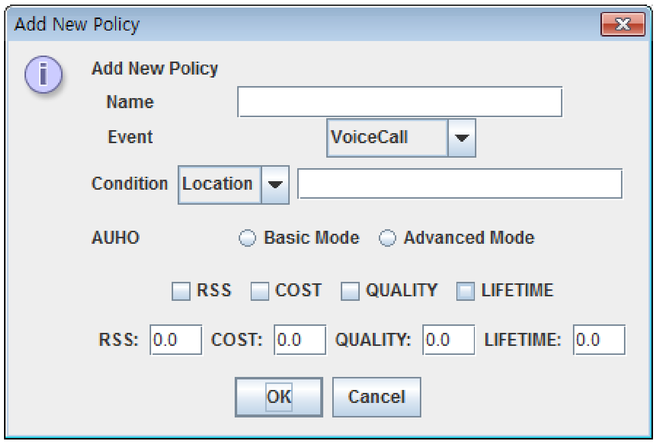


Figure 6: Simulation environment for handover decisions in CDMA, WLAN, and Mobile WiMax access networks

ports multiple network interfaces and applications. We then assigned a moving path for the mobile device; this is shown in Figure 6 as the large horizontal arrow. We then applied three different types of application traffic, which we generated from an NS-2 network simulator. Finally, we measured performance metrics of each handover decision algorithm and compared them.

In this experiment, we used CDMA, IEEE 802.16 Mobile WiMax (Mobile WiMax), and IEEE 802.11 based WLAN (WLAN) mobile networks. The area of the simulation network was 1,000 m by 1,000 m. Three CDMA BSs, one Mobile WiMax *Radio Access Station (RAS)*, and three WLAN APs covered the area. These access nodes were connected to the Router via 100 Mbps trunks with different traffic parameters. The coverage of each access point was represented by an associated ellipse. We chose the MIPv6 protocol as the IP mobility management protocol for the mobile nodes. One mobile node, MN1, was managed in our simulation environment. This mobile node moved from a starting coordinate $(147, 316)$ to an ending coordinate $(864, 504)$, with a speed of 40 km/h. The MN1 had three different types of network interfaces: CDMA, Mobile WiMax, and WLAN, which enabled it to communicate with each access network for the specific application. The context server gathered the context information from each access network. We controlled all the network parameters of each network device. The application server provided three different types of application traffic: VoIP, Streaming Multimedia, and FTP. We created traffic for each application using an NS-2 network simulator.

In the experiment, we predefined time-series parameters of each mobile network manually. Each of the six locations represented different control points to calculate handover decisions. We compared our handover decision algorithm with other handover decision algorithms at these locations using the HMNToolSuite.



Add New Policy

Name	
Event	VoiceCall
Condition	Location
AUHO	○ Basic Mode ○ Advanced Mode
	☐ RSS ☐ COST ☐ QUALITY ☐ LIFETIME
RSS:	0.0
COST:	0.0
QUALITY:	0.0
LIFETIME:	0.0

Figure 7: A screenshot of a policy setting dialog of the HMNToolSuite

5.1. Handover Decision Implementation

After creating all access networks and mobile nodes required for a given scenario, we implemented to configure handover decision policies as shown in Figure 7. We created a new policy with a name, an event, a set of conditions to be specified, and a set of actions to be taken [5]. In this case, an event or set of events, shows that something significant has happened (e.g., request to start an application, or the loss of communication). More importantly, events are used to trigger the evaluation of a condition. Conditions (one or a boolean combination of conditions) are used to determine which action or set of actions should be executed in response to the event (or set of events). Contextual information such as environmental data, is one important type of condition. We implemented three types of context: location, time, and contact. An action can be used to invoke one of the handover decision methods. After setting policies, we assigned a path for simulating each mobile node using a path editor. We created a path by adding points that define the path to be taken. We also changed the velocity and the time that the node stays at each point.

5.2. Discussion

In this case study, we showed that our tool suite provides testing handover decision algorithms for mobile devices in heterogeneous mobile networks. First, we implemented previous handover decision algorithms such as RSS-based, Quality-based, Cost-based, and Lifetime-based decision algorithms. We could obtain the same experimental results to those of the previous approaches using the HMNToolSuite. After that, we validated our handover decision algorithms using the tool suite. We can say that this tool suite is useful for testing mobility for mobile devices in heterogeneous mobile networks.

6. CONCLUDING REMARKS

In this paper, we have presented a novel tool suite for emulating and simulating mobile devices in heterogeneous mobile networks for testing management methods. Our HMNToolSuite allowed the user to create multiple types of wireless access networks, mobile nodes, and network servers for creating simulation scenarios. In the case study, we presented context-aware handover decision management which is a management issue for mobile devices. We also deployed this tool as an open source project to encourage other researchers to test and improve it.

For future work, we will integrate our tool into existing network simulators by importing network traces from them. Currently, time-series packet traffic data is manually configured for simulation. It is important to note that this tool is not an independent simulator designed to replace traditional simulators. We will also apply our tool suite to other useful case studies such as configuration, fault, and performance management.

ACKNOWLEDGMENTS

Joon-Myung Kang gratefully thanks Prof. Don Batory in the Dept. of Computer Science in the University of Texas at Austin for his comments on feature-oriented software development. This research was supported by the MKE (The Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program supervised by the NIPA (National IT Industry Promotion Agency) (NIPA-2010-C1090-1031-0009) and by the WCU(World Class University) program through National Research Foundation of Korea funded by the Ministry of Education, Science and Technology (R31-2010-000-10100-0).

REFERENCES

[1] Yan, X., Ahmet Sekercioğlu, Y., and Narayanan, S., 2010. A Survey of Vertical Handover Decision Algorithms in Fourth Generation Heterogeneous Wireless Networks. Computer Networks, 54, no. 11 (August):1848–1863.

[2] Kang, J.-M., Ju, H.-T., and Hong, J. W.-K., 2006. Towards Autonomic Handover Decision Management in 4G Networks. Lecture Notes in Computer Science, 4276 (October):145–157.

[3] Kassar, M., Kervella, B., and Pujolle, G., 2008. An Overview of Vertical Handover Decision Strategies in Heterogeneous Wireless Networks. Computer Communications, 31, no. 10 (June):2607–2620.

[4] Nguyen-Vuong, Q., Agoulmine, N., and Ghamri-Doudane, Y., 2008. A User-Centric and Context-aware Solution to Interface Management and Access Network Selection in Heterogeneous Wireless Environments. Computer Networks, 52, no. 18 (December):3358–3372.

[5] Kang, J.-M., 2011. Autonomic Management for Personalized Handover Decisions in Heterogeneous Wireless Networks. Ph.D. thesis, Pohang University of Science and Technology (POSTECH).

[6] Lampropoulos, G., Salkintzis, A., and Passas, N., 2008. Media-independent Handover for Seamless Service Provision in Heterogeneous Networks. IEEE Communications Magazine, 46, no. 1 (January):64–71.

[7] Heterogeneous Mobile Network Tool Suite (HMNToolSuite). `http://code.google.com/p/hmntoolsuite`, [Online; accessed 09-January-2011].

[8] NetSim, Tetcos. `http://tetcos.com/software.html`, [Online; accessed 10-January-2011].

[9] NS2 Network Simulator. `http://www.isi.edu/nsnam/ns/`, [Online; accessed 07-January-2011].

[10] NS3 Network Simulator. `http://www.nsnam.org/`, [Online; accessed 10-January-2011].

[11] OpNet Technologies Inc. `http://www.opnet.com/`, [Online; accessed 10-January-2011].

[12] Varga, A. and Hornig, R., 2008. An Overview of the OMNeT++ Simulation Environment. In Proc. of Simutools '08, (March 3-7), Marseille, France, 1–10.

[13] Sobeih, A., Hou, J., Kung, L., Li, N., Zhang, H., Chen, W., Tyan, H., and Lim, H., 2006. J-Sim: A Simulation and Emulation Environment for Wireless Sensor Networks. IEEE Wireless Communications, 13, no. 4 (August):104–119.

[14] Gamma, E., Helm, R., Johnson, R., and Vlissides, J., 1995. Design patterns: elements of reusable object-oriented software. Addison-wesley Reading, MA.

[15] Batory, D., 2005. Feature models, grammars, and propositional formulas. In Software Product Lines Conference, SPLC 05, LNCS 3714. (September):7–20.

[16] Clements, P. C. and Northrop, L., 2001. Software Product Lines: Practices and Patterns. SEI Series in Software Engineering, Addison-Wesley.

[17] jFreeChart: free Java chart library. `http://www.jfree.org/jfreechart/`, [Online; accessed 03-November-2010].

[18] Kang, J.-M., Strassner, J., Seo, S., , and Hong, J. W.-K., 2011. Autonomic Personalized Handover Decisions for Mobile Services in Heterogeneous Wireless Networks. Computer Networks, accepted to appear.

Biography

Joon-Myung Kang (eliot@postech.ac.kr) is Postdoctoral Fellow in the Dept. of Computer Science and Engineering at POSTECH, Pohang, Korea. He received his B.Sc. and Ph.D. in Computer Science and Engineering from POSTECH in 2005 and 2011, respectively. From 2000 to 2004, he worked at Alticast Corporation, as a software engineer. His research interests include autonomic network management, mobile device management, mobility management, personalized services, and software product lines.

Sin-seok Seo (sesise@postech.ac.kr) received his B.Sc. in Computer Science and Engineering from Inha University in 2008. Currently, he is a Ph.D. candidate in the Dept. of Computer Science and Engineering, POSTECH, Pohang, Republic of Korea. His research interests include autonomic network management and mobile device management.

John Strassner (johns@postech.ac.kr) is a Professor in the Division of IT Convergence Engineering in POSTECH, and leads its Autonomic Computing group. Previously, he was a Visiting Professor at Waterford Institute of Technology in Ireland, where he worked on various FP7 and Irish research programs. Before that, he was a Motorola Fellow and Vice President of Autonomic Research at Motorola Labs, where he was responsible for directing Motorola's efforts in autonomic computing and networking, policy management, and knowledge engineering. Previously, John was the Chief Strategy Officer for Intelliden and a former Cisco Fellow. John is the Chairman of the Autonomic Communications Forum, and the past chair of the TMF's NGOSS SID, metamodel and policy working groups, along with the past chair of several IETF and WWRF groups. He has authored two books (Directory Enabled Networks and Policy Based Network Management), written chapters for 5 other books, and has been co-editor of 5 journals dedicated to network and service management and autonomics. John is the recipient of the IEEE Daniel A. Stokesbury memorial award for excellence in network management, the Albert Einstein award for innovation in high technology, is a member of the Industry Advisory Board for both University of California Davis and DePaul University, a TMF Fellow, and has authored over 235 refereed journal papers and publications. He has 47 patents. He holds BSEE, BSCS, MSCS, and Ph.D. degrees.

James Won-Ki Hong (jwkhong@postech.ac.kr) is Professor and Head of Division of IT Convergence Engineering and Dean of Graduate School for Information Technology, POSTECH, Pohang, Korea. He received a Ph.D. degree from the University of Waterloo, Canada in 1991. His research interests include network management, network monitoring and analysis, convergence engineering, ubiquitous computing, and smartphonomics. James has served as Chair (2005-2009) for IEEE Comsoc Committee on Network Operations and Management (CNOM). He is serving as Director of Online Content for the IEEE Comsoc. He is a NOMS/IM Steering Committee Member and a Steering Committee Member of APNOMS. He was General Chair of APNOMS 2006 and APNOMS 2008. He was a General Co-Chair of 2010 IEEE/IFIP Network Operations and Management Symposium (NOMS 2010). He is an Associate Editor of IJNM and editorial board member of IEEE TNSM, JNSM, JCN, and JTM.