computing@computingonline.net
www.computingonline.net

ISSN 1727-6209
International Journal of Computing

# AUTOMATED NETWORK PROTOCOL EVALUATION – THE POTSDAM WIRELESS TESTBED

**Sebastian J.F. Fudickar, Bettina Schnor**

University of Potsdam, August-Bebel-Strasse 89, 14482 Potsdam, Germany
[fudickar | schnor]@cs.uni-potsdam.de, www.cs.uni-potsdam.de/bs

**Abstract:** *The Potsdam Wireless Testbed supports validation and evaluation of Wi-Fi radio stacks and wireless applications in environments with heterogeneous hardware. In contrast to simulators, wireless testbeds support the network stack validation with specific radio chipsets and radio signal propagations. Furthermore, wireless testbeds unburden programmers from manually updating software on nodes. Scheduled test-runs are executed automatically for a defined duration including compilation and deployment of the protocols and measurement scripts as well as collection of measurement results and log files. The testbed supports heterogeneous processor architectures and radio chipsets via internal cross compilation. The developer can overview the visualized results of its validation and therefore can focus on the code and the results. Next to the support of several device and processor architectures, the Potsdam Wireless Testbed is intended to support additional radio frequency ranges as well as mobile device.*

**Keywords:** *Test bed, validation, evaluation, wireless, Wi-Fi.*

## 1. INTRODUCTION

This article introduces the Potsdam wireless testbed utilized for the validation and evaluation of radio network stacks and is an extended version of an article published in the proceedings of the Wireless communications and information conference [1].

A necessity for automated protocol and application validation and evaluation was recognized in the ongoing development of the KopAL [2] support system for elderly. In KopAL users are equipped with mobile devices that assist them by reminding on upcoming appointments, recognizing critical situations (such as falls or losing-tracks) and offering emergency-call functionality.

Since the mobile devices are maintained by their caretakers, long device runtimes and reduced maintenance times are necessary, to be accepted. The development of the necessary energy efficient protocols (e.g. for communication and localization) requires regular validation and evaluations on multiple device types. These devices' characteristics differ regarding radio chips (with radio frequency utilization or functionality variations) and processor architectures.

In case of KopAL, the development covers the routing, logical link control (LLC) or media access control (MAC) layers as well as the radio device drivers and application logic. Therefore the KopAL indicates a tendency towards mobile applications that regularly require cross layer developments and wireless communication stack manipulation to achieve precise in-door localization, wireless sensor node access or higher energy efficiency.

The development of wireless network stacks requires extensive validation and evaluation, in general. Since changes may influence additional layers, an overall validation und evaluation is required especially in case of cross-layer network stack implementations (as found in Sub-1 GHz networks or RFID). As a result, the proper functioning of wireless network stacks can only be ensured by validating the complete communication stack, as recognized by Moss [3].

With heterogeneous hardware even the functional spectrum of device drivers and chips differs as recognized in [4] and therefore radio hardware and drivers' influences have to be taken into account during validation and evaluation, as well.

The proper functioning of wireless network stacks must sufficiently handle inter-node communication and physical phenomena such as noise or reflections to assure sufficient robustness.

As a result, network stacks must be validated with multiple nodes in a representative environment (with realistically placed obstacles and mobility) to assure its proper functioning within the dedicated environments.

Frequent node reprogramming is a time consuming part of the validation and evaluation process. The manual alteration of node software (including node collection, reprogramming and node redistribution) is time consuming and deployed nodes are hard to access. To increase the validation efficiency, alternative approaches have been developed that aim to automate the deployment of software on distributed wireless devices and reduce the manual validation and evaluation effort.

Existing automated validation or evaluation systems can be categorized in network simulators, wireless testbeds and hybrid solutions as discussed in Section Related Work. As shown, the advantage of testbed approaches is that they support the validation of the full network stack including the radio chipsets functionality.

However, since existing testbeds mostly focus on homogeneous equipment, they have limited value for the validation of networks that include devices with heterogeneous hardware capabilities (such as different radio chipsets, processor architectures or memory). Limited solutions support automated node mobility. Since the KopAL system requires the network stacks validation for heterogeneous devices (as summarized in Section 2) and node mobility, the existing solutions do not fit our requirements, as discussed in Section 3 of the KopAL project. A novel testbed system has been developed (as described in Section 5 to 7) and successfully deployed within the Potsdam Wireless test bed. Section 8 gives an overview of future extensions, which are currently under development.

## 2. HARDWARE

As shown in Figure 1, the Potsdam Wireless Testbed includes a PC with Ethernet and Wi-Fi (IEEE 802.11 b/g) network interfaces acting as a testbed management server and the following Wi-Fi devices:

- The LinkSys WRT54GL Router[1] supports the IEEE 802.11b/g Wi-Fi standard via a Broadcom Wi-Fi chipset and 4 +1 Ethernet interfaces. Next to 4 10/100 GBit Ethernet connectors 16 MB main memory, 4 MB flash drive and a Broadcom BCM5352 MIPS processor with 200 MHz.
- The Nokia N8x0[2] devices are supported as mobile devices. They contain IEEE 802.11b and g Wi-Fi transceivers, next to an USB serial connection, a 400 MHz ARM CPU, 128MB memory.

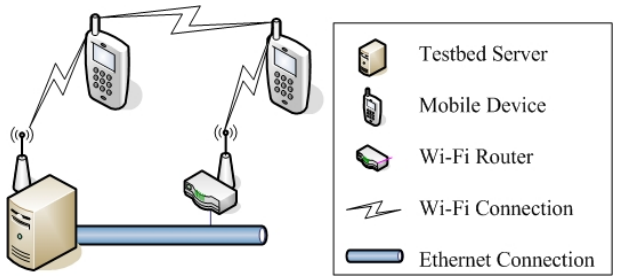


**Fig. 1 – Testbed hardware**

- The Openmoko Freerunner[3] is an alternate supported mobile device type. It contains as well IEEE 802.11b/g Wi-Fi transceivers, an USB serial connection, a 400 MHz ARM CPU, and 256MB memory.

Therefore the testbed has to support different processor architectures such as x86, ARM 9E and MIPS. As described in Section 3.3, this fact is concealed from the developer, as long as the compilation succeeded. In case of errors, the developer can analyze the logfiles and after alterations restart the compilation.

Depending on the devices network interfaces and mobility characteristics, the devices are connected with the testbed server. While the routers are accessible via an Ethernet connection, the mobile devices connect via Wi-Fi links – either with an in range Wi-Fi access point or directly with the server.

Since the inclusion of additional devices that include Sub-1 GHz radio network interfaces as the Chronoz Watch and the Mica2 sensor nodes is intended the testbed system should be able to handle multiple network standards and additional device classes.

## 3. REQUIREMENTS

The development of the KopAL System indicated general requirements that must be handled by the Potsdam Wireless testbed. The devices of the KopAL project have heterogeneous characteristics. These heterogeneous characteristics include the available radio interfaces (either the chipset and the frequency ranges and protocols), processor architectures and processing, memory or energy capabilities. In addition the devices mobility differs between the different device classes – while some remain stationary (as the Linksys Wi-Fi routers) others are mobile since carried around by the users. The validation system must handle this device diversity and the resulting necessities, which are further discussed in detail.

[1] http://www.linksysbycisco.com/LATAM/en/products/WRT54GL (14.09.2011)

[2] http://mea.nokia.com/support/product-support/nokia-n800-internet-tablet (14.09.2011)

[3] http://wiki.openmoko.org/wiki/Neo_FreeRunner (14.09.2011)

The intended radio standard diversity (even on a single node) requires the support of multiple network interfaces and the dynamic (de-)activation per node. In addition, the functional implementation provided by radio chips and device drivers differs from each other. As recognized in chips of the Sub-1 GHz frequency band, the implementation of CRC checks is supported by hardware in selected chipsets (such as the CC1100) while others (such as the CC1000) outsource it to the network drivers. In these frequencies even encoding of the Checksum varies (between high-bit and low-bit order). The Wi-Fi chipset variances as found in timestamp precision (of received messages) was recognized by Haustein [4] but complicates the network stack adaptation, next to differences between FullMAC and SoftMAC network cards. Since radio stacks may highly depend on functionalities (such as timestamp precision, RSSI measurements and CRC checks) network stacks must be evaluated on all relevant radio chipsets. The combination of different hardware devices with heterogeneous radio chips is problematic as well (e.g. by differing CRC checksum encoding or signal modulation). To validate robustness for these error types the validation must include all potential radio chipset combinations.

The heterogeneity of the devices (of the KopAL project) is not limited to the included radio chipsets. The devices as well include a various processor architectures (such as x86, ARM and MIPS). The compilation of the network stack for the different architectures should be handled autonomous by the testbed system, to relieve the programmer from preparing several solutions.

In addition, the available main memory of the devices varies. This has to be taken into account, since limiting the storage of measurement and log file on the devices during test runs. To exclude potential message losses, the testbed must handle this challenge as well.

The KopAL system includes localization functionality and supports node mobility. Since the node mobility may influence the link stability and the topology changes, it must be evaluated as well. To assure the comparability of the results, the node mobility must be reproducible.

While some devices exclude an internal energy supply (and are charged via power connectors) mobile devices include batteries and therefore are energy critical. Since the energy efficiency is a critical factor of a network stack – the testbed should support the collection of energy consumption as well.

Additional aspects of heterogeneity include supported programming languages for measurement scripts.

## 4. RELATED WORK

The manual reprogramming of nodes for the validation of wireless network stacks, can be overcome by several automated validation systems. These systems can be categorized as follows:

- Network simulators emulate virtual devices and simulate the message transmission between them with software. For simulating radio networks, the message transmissions are calculated via radio wave propagation models (such as [5] and [6]). Since the network stack is loaded into the simulator, developers must not reprogram multiple devices. In addition, for network simulation no specific hardware equipment is required. The limited investments and the reduced deployment times make network simulators a good choice for research groups with limited resources. In addition, simulators enable the evaluation of network stacks and protocols on tremendous amount of nodes. Simulators as well give easy debugging functionality and runtime speed-ups (by time accelerated).

However, simulators include several drawbacks that limit their validation usage to primer functional test of the upper layers.

Current radio propagation models approximate the radio propagation for algorithms efficiency, since "propagating each message all over the playground and hence delivering it to each node, slows down the simulation."[6] Therefore, if not excluded, these models [7] approximate phenomena such as interference, reflection or diffraction via ray tracing. In addition, the influence of obstacles (e.g. containing metal or water) on radio waves signal strength is excluded from any propagation algorithms, to the authors' best knowledge.

The recognized differences of radio chipset supported functionalities cannot be covered by simulators, making a validation on the utilized hardware indispensable.

- Testbed systems automate the deployment and validation process of network stacks on multiple devices. Therefore testbeds relieve developers from manual software-deployments, by automatically updating software on nodes. Next to the automatic update mechanism testbeds support the initiation and termination of test runs including the collection of measurement results and log files. Several existing testbeds include node mobility, either via robots [8], [9], or via humans carrying devices [10]. A major drawback of testbed systems is the limited debugging support, which in contrast the detailed simulation debug information is mainly limited to message statistics or log output (as

found in MoteLab [11]). While most existing wireless testbed systems support a single radio frequency, the EWANT [12], WHYNET [13] and DES [14] testbeds include wireless network transceivers for multiple frequencies (and combine 2,4GHz Wi-Fi with 868MHz transceivers). However even this system limits a test run to one frequency, while the alternate ones utilization is limited for configuration purposes instead of combined routing purposes. As a result these testbeds do not support the evaluation of hybrid networking solutions, what we aim for.

- Emulators, such as MiNT [15], combine node simulation with package transmissions over physical radio links by utilizing antennas (or attenuators) for message transmissions while the processing is executed on a single computer (similar to the simulator approach). Therefore emulators combine the advantages of both (efficient debugging of the simulation results with realistic radio signal distribution characteristics). However, the proper functioning cannot be validated on different hardware devices via hybrid solutions.

The above mentioned systems include the following restrictions, making their utilization for the KopAL system validation suboptimal. The simulation based testing and debugging seems appropriate for initial debugging, testing and validation – since message flows can be validated (even for large scale networks). However, simulation cannot validate the appropriate functioning on specific hardware and realistic circumstances (e.g. caused by approximate radio propagation models). Therefore an initial simulation based validation of developments must be supplemented by a "physical" test in realistic environments that should cover the intended hardware and physical conditions.

## 5. TESTBED MANAGER

The testbed manager is the essential component of the testbed. The testbed manager handles creation, deployment, execution and operation of test runs and initiates the collection of measurement results and log files.

Test-runs have a defined duration after which they are stopped.

Created via the Web-Frontend (as described in Section 6) they as well include a routing protocol, a measurement script and a (sub-) set of available nodes.

Once created, test-runs are scheduled in a Fi-Fo queue within the testbed manager for execution. The testbed manager executes uploaded testruns sequentially. When a test-run is executed, the nodes software is updated with the dedicated routing protocols and measurement scripts. In case the nodes have a separate network interface (such as Ethernet) it is used for configuration purposes. Nodes without a separate network interface, connected with the testbed server can continuously lose connection (caused by a faulty routing protocol). To prevent permanent disconnection of any node a management routing protocol is stored on the nodes. In the so called management mode the OLSR protocol [16] is used as "management routing protocol". In case a node permanently lost connectivity to the base station (recognized via heart-beat messages) the management mode is started.

For reprogramming purposes all nodes switch (between test runs) to the management mode as shown in Figure 2. Therefore disconnected nodes automatically switch to the management mode, waiting to rejoin the network after the current test run is finished.

When a test run is finished, the management server collects all measurements and log files and generates (visual and textual) summaries of them. Afterwards it requests all nodes to reset its routing protocol to the management mode. Receiving this reset request each node stays connected for 1 minute, to assure sufficient time to forward the reset message to the whole network, before changing into management mode.

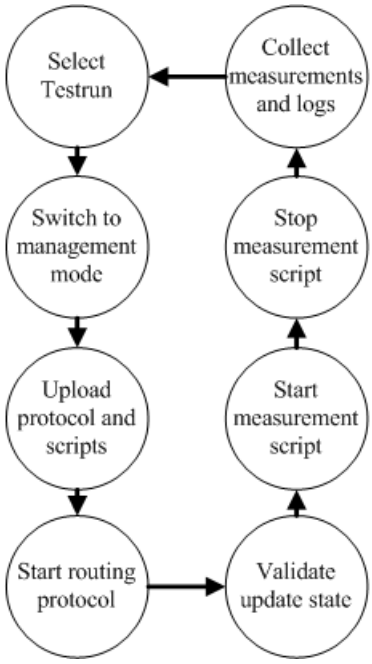


**Figure 2: Testbed state transitions**

After all required nodes have entered the backup mode, the routing protocol and measurement scripts of the new test run are deployed on the devices. The appropriate routing and measurement scripts are transmitted compressed to the nodes and afterwards

decompressed and installed on them.

Afterwards the testbed manager initiates a (delayed) start of the uploaded routing protocol on all specified nodes. As a result the nodes leave the managed mode. So does the testbed server.

The installation is validated via a status request message, which is initiated by the testbed manager after the restart. The status response indicates (next to the connection success) the version of the active routing protocol and measurement script.

Unavailable nodes are excluded from the testrun but are logged by the testbed manager for later debugging purposes. The user can interrupt the current testrun manually (at any time) and reschedule an altered version for later execution.

The testbed manager starts the test run by initiating the measurement script. As described in Section 7 measurement scripts collect connection data (such as quality or signal strength measurements) and generate the network traffic. As a result, the message flows during a testrun are regulated by measurement scripts and the routing protocols. Nodes as the Linksys WRT54GL router may regularly face full memory problems. In such case, measurements results can be transmitted (pushed) from the devices, to the testbed server during a test run. To reduce the network load all data is compressed on the devices before transmission. Being aware of potential influences on the measurement results, the measurements (and log files) are preferably stored on the nodes and pulled by the testbed server when the test run is finished. Since the Linksys routers use a wired connection with the testbed server for these transmissions these influences are excluded in our current setup.

On the testbed server the data is decompressed, processed and attached to the test run.

When the specified testrun duration has passed, a stop message is send by the testbed manager to all participating nodes, which finalize the measurement script execution (if not already finished).

All control messages for protocol and measurement script updates, test run management and collection of measurement results and log files are handled by the testbed manager. HTTP is utilized as communication protocol combined with a php processor on each node, to handle incoming requests, for interoperability and portability reasons. Some control tasks (such as the update of routing protocols or packaging of measurement results) utilize additional shell scripts, executed by the PHP scripts.

If available, testbed devices utilize the buildings' Ethernet infrastructure. To minimize the impact of resulting network traffic (which might increase in case of updates) on other computers and the overall network performance, the participating nodes are grouped within a virtual LAN.

Next to the execution and planning of upcoming testruns, the testbed manager handles additional functionality such as the preparation of uploaded routing protocols or measurement scripts.

Uploaded measurement scripts (that require compilation) or routing protocols are immediately prepared for deployment.

Therefore they are automatically compiled for all supported platforms via cross compiler tool chains.

## 6. USER INTERFACE

All user interactions of the test-bed system are executed via an AJAX Web page. As a result the testbed can be managed location independent.

Users can upload measurement scripts and routing protocols, manage nodes, create configuration and schedule test runs.

In addition the current state of a testrun and the results are summarized. Therefore test run results (including dynamic visualizations and raw data of measurement results as well as log files) are visualized. Automatic generated gnuplots of measurements can be downloaded as well and used for future documentation.

Since the testbed server is developed with Java, the user interface as well was developed in Java including the Spring[4] and Hibernate[5] Frameworks.

## 7. MEASUREMENT SCRIPTS

Measurement scripts should be executable on all device-types. Since separated from the routing protocols, measurement scripts are applicable in general (as long as the protocol supports the required interfaces) and enable a comparison of routing protocols.

In the Potsdam Wireless Testbed, measurement scripts can be programmed either as shell scripts, or as C programs. Therefore developers can use the full spectrum of the available Linux tools (such as ping or wget through scripting) or may develop specific solutions to precisely fit their requirements. The measurement scripts can be extended by additional precompiled programs (as long as portable to the selected nodes) which then should be packaged with the specific measurement script package.

[4] http://www.springsource.org/ (14.09.2011)
[5] http://www.hibernate.org/ (14.09.2011)

## 8. FUTURE WORK

The Potsdam wireless testbed currently supports various devices with Wi-Fi transceivers but is currently extended by additional radio spectrums, hardware platforms. The supported hardware platforms are extended by additional devices such as the Mica2[6] sensor nodes and the TI eZ430 Chronos watch[7]. Therefore additional radio spectrums such as Sub-1 GHz are supported as well. To ensure a high portability the inclusion of TinyOS[8] as additional Operating system for these devices is intended.

Furthermore, the testbed will be equipped with robots carrying mobile devices during test runs, including autonomous recharging functionality.

## 9. REFERENCES


[1] S. J. F. Fudickar, M. Strewe, O. Eggert, B. Schnor, The Potsdam wireless testbed, in *Proceedings of Wireless Communication and Information 2011*, Berlin, Germany, October 2011, Verlag Werner Hülsbusch.

[2] S. J. F. Fudickar, B. Schnor. J. Felber, F. J. Neyer, M. Lenz, M. Stede, KopAL – An orientation system for patients with dementia, *in Behaviour Monitoring and Interpretation – BMI Well-Being*, B. Gottfried. H. Aghajan, IOS Press, Amsterdam, Netherlands, April 2011.

[3] D. Moss, CCxx00 Radio Stack, Technical documentation.

[4] M. Haustein, *Untersuchung von Methoden zur Laufzeitmessung in Wireless LAN Netzwerken zum Zwecke der Positionsbestimmung*, Diploma Thesis, Chemnitz, 2011.

[5] A. Köpke, M. Swigulski, K. Wessel, D. Willkomm, P. T. Klein Haneveld, T. E. V. Parker, O. W. Visser, H. S. Lichte, S. Valentin, Simulating wireless and mobile networks in OMNeT++ the MiXiM vision, *in Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops (Simutools'08).*

[6] A. Kuntz, F. Schmidt-Eisenlohr, O. Graute, M. Zitterbart, Introducing probabilistic radio propagation models in OMNeT++ mobility framework and cross validation check with NS-2, *in Proceedings of the 1st International Workshop on OMNeT++ (hosted by SIMUTools 2008).*

[7] A. Lewandowski, V. Köster, C. Wietfeld, A new dynamic co-channel interference model for simulation of heterogenous wireless networks, *In Proceedings of the 2nd International Workshop on OMNeT++ (hosted by SIMUTools 2009).*

[8] P. De, A. Raniwala, R. Krishnan, K. Tatavarthi, N. S. Ahmed, J. Modi, T. Chiueh, MiNT-m: An autonomous mobile wireless experimentation platform, *In Proceedings of Mobisys, 2006.*

[9] Oliver Hahm, Mesut Günes, Felix Juraschek, Bastian Blywis, Nicolai Schmittberger, An experimental facility for wireless multi-hop networks in future internet scenarios, *In Proceedings of The 2011 IEEE International Conference on Internet of Things, IEEE, Dalian, China (2011)*

[10] Geoffrey Werner-Allen, Patrick Swieskowski, Matt Welsh, MoteLab: a wireless sensor network testbed, *In Proceedings of the 4th international symposium on Information processing in sensor networks. IPSN 2005*

[11] H. Lundgren, D. Lundberg, J. Nielsen, E. Nordstrom, C. Tschudin, A large-scale testbed for reproducible ad hoc protocol evaluations, *In Proceedings of IEEE Wireless Communications and Networking Conference, WCNC2002, 2002*.

[12] S. Sanghani, T. X. Brown, S. Bhandare, S. Doshi, EWANT: the emulated wireless ad hoc network testbed, *In Proceedings of Wireless Communications and Networking, WCNC 2003.*

[13] Maneesh Varshney, Zhiguo Xu, Shrinivas Mohan, Yi Yang, Defeng Xu, Rajive Bagrodia, WHYNET: a framework for in-situ evaluation of heterogeneous mobile wireless systems, *In Proceedings of the second ACM international workshop on Wireless network testbeds, experimental evaluation and characterization. WinTECH 2007*

[14] Oliver Hahm, Mesut Güneş, Kaspar Schleiser, DES-Testbed a wireless multi-hop network testbed for future mobile networks, *In Proceedings of 5th GI/ITG KuVS Workshop on Future Internet*, Stuttgart, Germany (2010)

[15] Pradipta De, Ashish Raniwala, Srikant Sharma, Tzi-cker Chiueh, MiNT: a miniaturized network testbed for mobile wireless research, *In Proceedings of IEEE Infocom, 2005.*

[16] T. Clausen, P. Jacquet, Optimized Link State Routing Protocol (OLSR), *IETF Request for Comments 3626,* 2003.


[6] http://bullseye.xbow.com:81/Products/productdetails.aspx?sid=174 (14.09.2011)

[7] http://processors.wiki.ti.com/index.php/EZ430-Chronos (14.09.2011)

[8] http://www.tinyos.net/ (14.09.2011)

***Sebastian J.F. Fudickar**, is a Phd. student at the department of Operating Systems and Distributed Systems at Potsdam University, where he received his Master of Science degree, following his B.Sc. in Software Systems Engineering of Hasso Plattner Institute at Potsdam University.*

*Next to his research in Wireless Sensor Networks and Mobile Ad-hoc Networks, he holds an expertise in Assisted Living Solutions for Elderlies.*

***Bettina Schnor**, was from 1990 to 1996 research scientist at the Institute of Operating Systems and Computer Networks at the Technical University Braunschweig. There, she established the research field "Distributed Systems" and the working group "Failure Tolerance and Load Balancing in Distributed Systems". From 1996 to 2000 she was researcher and lecturer at the Institute of Telematics at the University of Luebeck, Germany.*

*Since April 2000 she is head of the Department of Operating Systems and Distributed Systems at the Potsdam University.*

*Her research interests are distributed systems, cluster and grid computing, network security, and assisted living applications.*