RWTH Aachen

Department of Computer Science
Technical Report

From Simulations to Testbeds — Architecture of the Hybrid MCG-Mesh Testbed

Mesut Günes, Alexander Zimmermann, Martin Wenig,
Jan Ritzerfeld, and Ulrich Meis

ISSN 0935–3232 · Aachener Informatik Berichte · AIB-2006-10

RWTH Aachen · Department of Computer Science · August 2006

From Simulations to Testbeds — Architecture of the Hybrid MCG-Mesh Testbed

Mesut Günes, Alexander Zimmermann, Martin Wenig,
Jan Ritzerfeld, Ulrich Meis

Lehrstuhl für Informatik IV
RWTH Aachen, Germany
Email: {guenes,zimmermann,wenig,ritzerfeld,meis}@cs.rwth-aachen.de

Abstract. The study of wireless and mobile networks is mainly based on simulations. According to recent publications 76% of the studies in this area are based on simulations. Although simulation environments offer a convenient combination of flexibility and controllability, their largest disadvantage is that the results gained by using them are difficult to transfer into reality. This is due to the complex environment of mobile and wireless networks.

In this paper we introduce a hybrid testbed approach, which consists of real mesh nodes and a virtualization environment. This combination provides on the one hand a flexible development environment for distributed network protocols and applications, and on the other hand a high degree of realism. Therefore, it allows the design and evaluation of large scale networks where the results are easily transferred to the real world.

1 Motivation

In the last decade wireless networks became a popular alternative to wired networks. Standardization and a decrease in production costs have facilitated their success in the mass-market and also in private households.

Nowadays, two different kinds of wireless networks are widely spread. On the one hand there are the *Global System for Mobile Communication (GSM)* and the *Universal Mobile Telecommunications System (UMTS)*, the cellular networks of the second and third generation respectively. They are primarily used for mobile telephony, but also increasingly for data transfer. On the other hand, there are several established standards for the wireless connectivity of client devices. The client devices can be either static or mobile, and are in general denoted as nodes. These standards include, amongst others, the popular *IEEE 802.11 (WLAN)*, as well as *IEEE 802.15 (WPAN)* and *IEEE 802.16 (WMAN)*.

All these networks share one property: They are designed for communication where only few wireless links exist between the source and destination node. Either the communication peer can communicate directly or it cannot. In the first case there is only one wireless link and in the other case there is a more or less complex infrastructure, which is based on classical wired network technology. The wireless devices are connected to distinguished devices of the infrastructure that are denoted as base stations. In GSM, the infrastructure consists of the *Home Location Register (HLR)*, the *Visitor Location Register (VLR)*, various service nodes, and gateways to other wireless and wired networks. In WLANs the base stations, so-called access points, are typically connected via Ethernet.

In recent years many researchers have invested huge efforts on: *mobile ad hoc networks (MANETs)* [22,40,7,62,44]. A MANET is a collection of wireless mobile nodes, e. g., laptops, personal digital assistants (PDAs), or smartphones, dynamically forming a network without the need for a pre-established infrastructure such as access points or base stations. Due to the limited transmission range of wireless network interfaces, the nodes have to cooperate by forwarding packets so that nodes outside transmission range can communicate with each other. In other words, each mobile node operates not only as a host but also as a router. Typical MANET application scenarios are battlefields or emergency search-and-rescue operations, where the use of any pre-established infrastructure is not possible, either because it does not exist or because it is out of order.

In spite of massive efforts in research and development of mobile ad hoc networks, this type of network has not yet seen mass-market deployment. The low commercial penetration of products based on MANET technology is understandable, as the ongoing research is mainly focused on large-scale military applications with thousands of ad hoc nodes and in some exceptional cases applicable on scenarios like emergency search-and-rescue operations.[1] The military and specialized civilian applications are characterized by a lack of infrastructure and instant deployment. In contrast, users are interested in general-purpose applications where *high bandwidth* and *open access to the Internet* are consolidated. To make MANETs useful for the mass-market some changes to the common definition of MANETs are mandatory. By relaxing one of the main constraints of MANETs, "the network is made of user devices only and no infrastructure exists," a new class of networks emerges: *wireless mesh networks (WMN)* [10,2].

WMNs do not have the objective to constitute a multi-hop ad hoc network that is fully self-sustaining, self-configured and isolated, but rather a network which emerges as a flexible and low-cost extension of wired infrastructure networks coexisting with them. Neither mobility nor flexibility is fundamentally lost, since wireless mesh networks generalize mobile ad hoc networks.

1.1 Contribution of this Paper

The contribution of this paper is threefold, which is also reflected in its structure. The first aspect of this paper is to outline the requirements of a real testbed for wireless mesh networks. The focus hereby is on the transition from a pure simulation based study of wireless mesh networks to the study based on a real and manageable testbed. This transition is important, since pure simulation based study results do not fit with reality. Although there are many wireless mesh networking projects, the reports do not give an insight how to build and run a wireless mesh networking testbed. This is valid for the hardware components as well as for the software components. However, this is important, since the setup and running of a testbed is labor intensive.

The second aspect of the paper is to give a survey of existing wireless mesh networking testbed projects. The focus here is to extract the information which identifies them and to collect the information which the projects share. Based on the study of these existing projects we infer some advice on setting up a wireless mesh networking testbed.

The third and last aspect of this paper is to present and discuss our wireless mesh networking testbed. We have used the experience of other testbed projects

[1] See for example Victor Bahl's opening talk at the Mesh Networking Summit 2004 [39].

to improve some properties. For research purposes it is not enough to set up a wireless mesh network testbed only consisting of hardware and the essential software, i. e. operating system and router functionality. The study process during research studies has to be considered during the design of a testbed. Our approach is a *hybrid testbed*, which consists of real hardware, standard Linux software and a virtualization technique. The virtualization allows the development and testing of the software as if it was executed on real mesh routers. This improves on the one hand the development of distributed protocols and applications and simplifies on the other hand the software-development cycle.

1.2 Structure of the Paper

The remainder of the paper is organized as follows. Section 2 gives an overview of wireless mesh networks. In Section 3 we discuss environments used to study mobile and wireless networks, in particular wireless mesh networks. Our solution to investigate WMNs is presented in Section 4. Previous work, that is existing testbeds for WMNs, is surveyed in Section 5. Finally, in Section 6 we draw some conclusions and outline future work.

2 Wireless Mesh Networks

When reading about *wireless mesh networks* one quickly realizes that the term really denotes a category rather than a specific network type. Given two papers about WMNs one is likely to find two entirely different sets of assumptions. Therefore, this section aims to categorize WMNs into distinct types. In order to do so, we first introduce the general network architecture before we describe the characteristics of WMNs. In particular, a functional classification of node types encountered in WMNs is presented that provides us with a set of specific terms, which are used in the remainder of this paper.

2.1 System and Network Architecture

Since the term "wireless mesh network" is ambiguous we will present a definition that seems to us the most promising. Figure 1 depicts a hierarchical and layered architecture that integrates the common approaches and thus helps to identify the main parts of a WMN. Our view is more general than that usually presented, in that other approaches often leave out some layers, e. g., they consider only clients without routing functionality [10].

On the top level of Figure 1 are the *backbone mesh gateways* connected to the Internet by wire, indicated by solid lines. They provide wireless Internet access (dashed lines) to the second level entities, the so-called *backbone mesh routers*. These wireless routers form the core by building the wireless, meshed backbone of the WMN. On the lowest level, there are the mobile user devices, the *mesh clients*. As Figure 1 shows, these clients are subdivided into two groups. On the left hand side, there are *routing mesh clients* that also communicate among each other in a multi-hop fashion. They form a MANET with gateways that are not directly connected to the Internet, but to the backbone mesh gateways. On the right hand side, there are *non-routing mesh clients*, which connect to

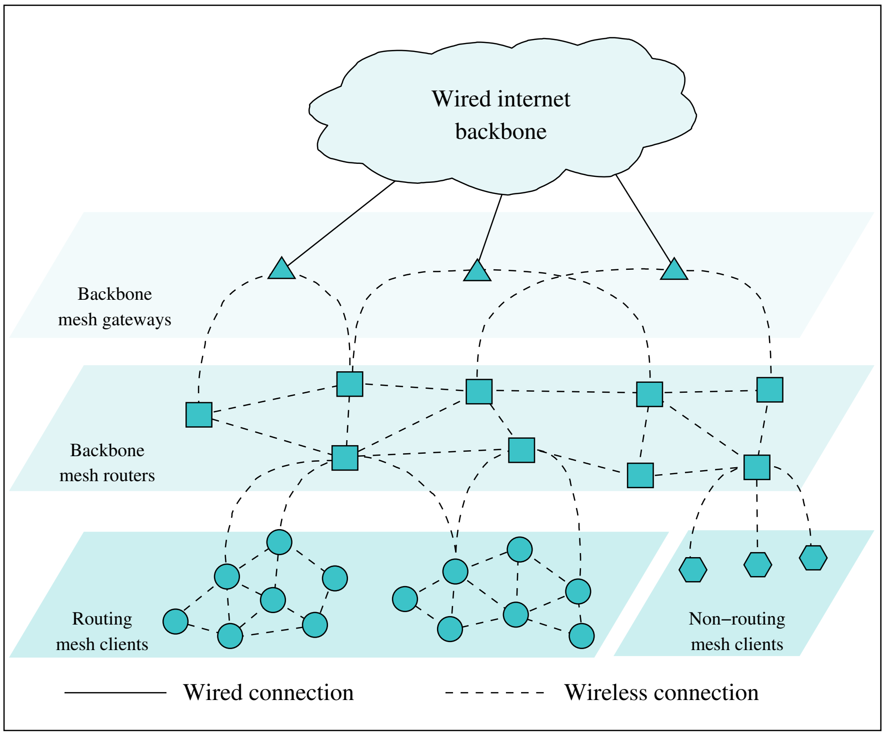

Fig. 1: Architecture of wireless mesh networks

mesh routers in the same way as conventional clients associate to wireless access points.

The architecture outlined above needs further discussion. First of all, the mesh gateways are specific mesh routers that have a wired, high-speed connection to the Internet. These wired connections are considered not to be part of the WMN. Thus, the WMN itself is fully wireless.

The mesh routers and gateways are installed at certain fixed positions. They establish a permanent infrastructure. However, new routers and gateways can easily be added, since the communication is wireless. Thus, the infrastructure and therefore the network topology is not completely static but has low dynamic character. Mesh routers and mesh gateways together establish a wireless multi-hop network that serves as a backbone. It routes traffic hop-by-hop from a mesh client to a backbone mesh that can forward it to the external network, and vice versa. This way of communicating is a major difference to conventional wireless access points. These provide only gateway or bridge functionality. In addition, a mesh router has multi-hop routing capabilities. Furthermore, the hierarchy achieved by the distinction between clients and routers promotes the utilization of multiple radios, separating the traffic in the backbone from that of the clients. Routing and configuration tasks are assigned to mesh routers in order to unburden mesh clients that are probably power-constrained because of their inherent mobility.

Due to the mobility of the mesh clients the wireless mesh network has a spontaneous and dynamic character. Mesh clients can leave the WMN at any time, and new clients arrive that want to join the WMN. Figure 1 introduces two groups of clients, in which the non-routing mesh clients are confined to direct communication with mesh routers only. They do not participate in the routing process of the WMN and use the mesh routers similar to conventional wireless clients communicating with their access point. The routing mesh clients are able to connect not only to mesh routers but also to other routing mesh clients. Since they participate in routing, these clients build up a sub-network on their own, which can also be considered as a MANET.

2.2 Network Characteristics

Our definition of the architecture of a WMN leads to several characteristics. These are quite general and many of them also hold for other perceptions of WMNs. To a certain extent they are also valid for (hybrid) MANETs [49], but there are more or less subtle differences.

Wireless: The most obvious property is the wireless nature of WMNs. Thus, WMNs must cope with the challenges that arise from wireless communication. On the one hand, they need to take into account the limited transmission range and the potentially high loss rates due to packet collision and fading of the wireless channel during the transmission. On the other hand, they have to deal with the mobility of nodes.

Multi-hop: WMNs use multi-hop routing to overcome the challenges mentioned above. Conventional wireless networks extend their network coverage by higher transmission power or additional access points that have to be interconnected by wire. In contrast, nodes of a WMN forward traffic wirelessly on behalf of others, which are not within their direct transmission range.

Redundancy: The wireless backbone of a WMN forms a meshed network. It provides redundant links between mesh routers, mesh gateways, and mesh clients. Thus, failure of one link or node will not necessarily lead to failure of large parts of the network. Trying to adopt this approach in conventional wired networks might be expensive, time-consuming, or even impossible. This is because of the large amount of cabling required for such a meshed network. Depending on the environment, in-wall installations could be impossible.

Mobility: Since both mesh routers and mesh gateways have low mobility, the backbone can support client mobility in a predictable and reliable fashion. Mobile clients that leave the communication range of one mesh router can easily connect to the next one coming into their communication range. The dynamic multi-hop routing will ensure that the traffic is still correctly forwarded to its destination.

Dynamics: All nodes have to establish the network spontaneous way *(self-organizing)* and to maintain their connectivity continuously *(self-healing)*. Leaving or newly joining nodes cause topology changes that the network has to adapt to. Nodes must reorganize their routes, invalidate paths that are not available anymore and include new paths that have become available. Additionally, the WMN should pass configuration information to new nodes in order to reduce or remove the need for user intervention *(self-configuring)*.

The characteristics mentioned above are essential, but MANETs share similar qualities. The following characteristics are specific to WMNs, they clearly distinguish WMNs from MANETs, even from hybrid ones.

Infrastructure: Unlike MANETs, WMNs have got a pre-existing, hierarchical architecture. Mesh gateways and routers forming the backbone infrastructure are nearly static and therefore less limited with regard to power consumption and computing power. They can be equipped with multiple radios, and they can take over routing and configuration tasks. The hierarchical static infrastructure simplifies such functionalities since the backbone is more reliable than mobile nodes are.

Integration: The non-routing mesh clients can join a WMN without the need for sophisticated routing support. Thus, light-weight and power-constrained clients can be attached to a WMN. They need not be an active part, in contrast to MANETs that require all nodes to be cooperative. Supporting such passive clients enables the integration of devices, or whole networks, into the wireless backbone.

The latter characteristics influence the ones mentioned before. The introduced hierarchy adds several improvement opportunities in comparison to MANETs.

2.3 Network Classification

Based on the definition of WMNs and with their characteristics in mind, we can present our classification approach. We will have to broaden our purely technical point of view to be able to grasp the vast differences implied by the variety of management styles that one finds in todays wireless mesh networks.

Fully managed WMN The most distinctive feature of fully managed WMNs is the absence of routing mesh clients (see Figure 1). Indeed, mesh clients do not perform any duties within the network. Therefore, all services are solely provided by the backbone and consequently there is a clear distinction between the WMN infrastructure and its users. The entire network is administrated by one entity.

Probably the most popular perception of a WMN has been promoted by Cisco's projects covering entire U. S. cities [26]. The clients, or customers, use the WMN as an Internet Service Provider. It is transparent to them that a WMN is providing this service to them. In particular, they do not take part in any routing other than exchanging packets with their selected peer, i. e., they are non-routing mesh clients. All other nodes belong to the backbone.

Semi-managed WMN In networks of this class, core parts are administered by one institution. However, a significant amount of nodes is not under the administration of that institution. Those nodes might join or leave the network at any time or even move within the network. Unlike clients in a fully managed WMN they are an integral part of the network and hence perform duties such as routing, auto-configuration support, and service discovery.

A popular representative of this category can be found in the city of Berlin. The "Freifunker" community [23] covers parts of the city with a WMN of more than 200 nodes. Some static nodes have been installed to cover long distances, thereby minimizing the maximum amount of hops. However, every node participates in routing. In contrast to a fully managed WMN the network topology changes frequently.

Unmanaged WMN Since there does not exist an institution to manage permanent resources, an unmanaged WMN merely uses available infrastructure but does not have one itself. Consequently, a high level of self-organisation is neccessary for all employed protocols. Some nodes might provide external connectivity by accessing adjacent—but separate—networks, e. g. via GSM, UMTS, or WLAN. These networks are, by their nature, a temporary phenomenon and might emerge in a seminar or conference room, a camping site, train, etc.

This case might not be central to Microsoft's research in this area but is at least covered by its "Mesh Connectivity Layer (MCL)" solution (see Section 5), which mainly encompasses a routing protocol. It is targeted at end-consumers and designed for unmanaged WMNs. Most notably, the implementation runs on Microsoft's OS for mobile devices. This contradicts Cisco's approach, which merely characterizes WMNs as a new backbone technology.

One notices that the definition at hand is similar to a hybrid MANET [49]. Indeed, researchers disagree upon whether there really exists such a thing as a WMN without pre-existing infrastructure.[2] However, it is common in MANETs to assume high levels of mobility. Conversely, in an unmanaged WMN one would expect the nodes to be rather fixed—at least while communicating. Most adaptations in routing are due to nodes joining or leaving the network. This assumption allows significant optimizations compared to pure MANETs.

2.4 Application Scenarios

As mentioned in Section 1, a novel technology needs promising applications. Indeed, they already exist for WMNs. To become commercially successful and widely accepted WMNs must offer added value to a large audience. Supporting only special applications, like MANETs do, rarely leads to mass-market deployment. In contrast, WMNs have a wide range of applications, and various scenarios are conceivable. In this section, four use cases of WMN deployment will be presented. The first three are exemplary for WMNs and they take ample advantage of their characteristics. We have chosen these scenarios because they do not only differ in size, but in the traffic pattern they generate within the network. Therefore, they illustrate common but distinct applications. The last scenario represents widely used examples for MANETs and clarifies that in these cases WMNs are applicable as well. However, this list is not exhaustive and further examples can be found in [2] and [10].

Broadband Home Networks In recent years, WLANs became popular in providing easy-to-deploy broadband Internet access at home, e. g., via DSL or cable modem. Their drawback is that a single wireless access point cannot cover a wide area like a whole house. Especially indoor placement can reduce the communication range due to walls, ceilings, and other interfering objects such as electronic devices. Thus, cabling is still needed for the interconnection of access points. Here, a WMN can be built up by mesh routers instead of conventional access points. They form the backbone that provides Internet access without any additional wiring. Conventional wireless clients are still able to connect to the WMN. The use of routing mesh clients is not mandatory but beneficial.

[2] e. g. see [10] pro versus [2] contra the infrastructure requirement.

Nonetheless, broadband home networks are not limited to Internet access. High capacity hard discs and efficient video codecs of today make fully digital video recorders feasible. A digital network is most appropriate to distribute TV or recorded programs. Remote control of distant devices is easily enabled by the bidirectional nature of the network. No additional cabling is required. The backbone of a WMN helps to establish a high-speed network connecting all video devices. Multimedia applications are time-critical and bandwidth-consuming, thus, a reliable and fast network is mandatory. WMNs will shortly be capable of fulfilling such requirements.

Another emerging application, IP telephony, has similar requirements. While it does not consume as much bandwidth as video streams do, it is also time-critical but certainly demands for higher mobility. Again, WMNs provide extensive network coverage with low installation effort and costs.

Enterprise Networks Enterprise buildings often consist of several floors and many offices that have to be interconnected. Modern buildings may be prepared for wired computer networks but as enterprises grow and network technology evolves existing cabling sooner or later becomes insufficient. Thus, networks are not only a one-time investment but they have to be renewed periodically. However, WMNs can be easily extended by adding mesh routers, and be upgraded by simply replacing them. In contrast, the renewal of wired networks often results in an all new, costly cabling.

When it comes to interconnect several buildings cabling gets really expensive. For legal and/or cost reasons leased lines are often inevitable. In an enterprise WMN, the costly, inflexible, and time-consuming cabling is replaced by wireless connections between mesh gateways, routers, and clients. Multiple mesh gateways and routers can provide redundant links to improve the robustness of the network.

Enterprises regularly maintain their own servers, e. g., for an intranet. Thus, as in broadband home networks, there is a huge amount of internal traffic. While Internet access speed is expected to be comparably low, one is used to high-speed internal network access. Low reliability or high latency to access vital business applications is unacceptable. Thus, providing fast and reliable intranet service wirelessly is a challenging task. The wireless backbone of WMNs can greatly improve reliability utilizing multiple radios and multiple links.

Community and Metropolitan Area Networks Community networks and wireless metropolitan area networks aim at offering an alternative to wired broadband network access. On the one hand, communities like the "Freifunker" [23] envision the spread of free networks, democratization of communication systems, and promotion of local social structures. By networking whole districts, villages, and regions they want to fight the digital divide; providing low cost Internet access is not their primary goal. On the other hand, commercial metropolitan area networks like [26] have the more pragmatic goal to offer wireless Internet access instead of bridging the last mile by wire.

Community networks fall into the category "Semi-managed WMN" (Section 2.3) and metropolitan area networks into "Fully managed WMN" (Section 2.3). Nonetheless, community networks may become metropolitan area networks in their final expansion stage. Such networks share the traffic pattern of

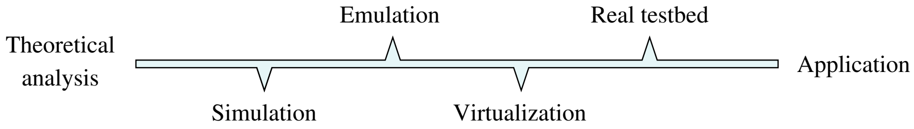

Fig. 2: Different environments for wireless networks

the first and the size of the second application scenario, if not more than that. This combination of factors clearly demands the hierarchical approach that is provided by WMNs.

Emergency and P2P Networks In addition to the application scenarios mentioned above WMNs also support typical MANET applications. For example, control centers and accommodations are established even in emergency or disaster situations. As they provide the infrastructure for the relief units, they can also provide a backbone and elevate the spontaneous network to a WMN. However, peer-to-peer communication is supported and enhanced by WMNs. Their backbone offers reliable, long-term connections by reducing the amount of mobile intermediate hops between two peers.

3 How to Study Wireless Mesh Networks

After the design of a new network protocol a researcher has typically several possibilities to evaluate and validate it. As Figure 2 shows, approaches of evaluation and performance analysis of network protocols can be classified into five categories. These are theoretical analysis, simulation, evaluation through emulation or virtualization, and the direct measurement in a real world testbed. All these evaluation methods are very different in their degree of abstraction, relative to the real application. Mathematical analysis has the highest abstraction followed by, in descending order, the simulation, emulation, virtualization, and finally reproduction in a real world testbed. The use of simplifying quantitative models thereby leads to a deviating behavior of the experimental setup. The more parameters remain unconsidered in such a model, the larger the inaccuracies that can occur in the evaluation.

There are some aspects that characterize reliable research studies. Typically, research studies are done by conducting series of experiments. According to Merriam–Webster [36] an experiment is an operation carried out under controlled conditions in order to discover an unknown effect or law. The following criteria are particularly important in the area of wireless network studies and should be considered [31].

Repeatable: The reported results should be reproducible by other researchers. This requires a detailed description of the experiment setup, study environment, and results.

Unbiased: The results should reflect a general idea of the subject of the study and should not be specific to an experiment.

Rigorous: The experiment setup must reflect the true character of the subject to study.

Statistically sound: The analysis of the experiment results must be based on mathematical methods.

3.1 Theoretical Analysis

Theoretical analysis uses mathematical constructs and models to evaluate network performance. Queueing theory is one of the most common mathematical tools in network performance studies. Unfortunately, theoretical analysis of mobile networks and wireless mesh networks is very difficult, since the mathematical constructs get very complex for realistic considerations; useful mathematical tools do not exist.

3.2 Simulations

Despite the fact that mobile and wireless networks have been subject of research for many years, the most experiences were gained by simulations. According to [31] 75.5% of the full papers in the prime conference MobiHoc used simulation to conduct their research studies.

A simulation environment offers a high degree of control and repeatable results to the researcher. This is especially useful when studying highly distributed networks like mobile ad hoc networks, sensor networks, and wireless mesh networks. During the study of such a network, typically few parameters are changed and most of them remain fixed. This allows the study of the network regarding these varied parameters. It allows the creation of complex network topologies, including mobile networks. Furthermore, it guarantees repeatable experiments, which in turn allows fellow researchers to conduct exactly the same experiments and confirm their results. Simulation studies are therefore very flexible and the related costs are low, since it is possible to conduct complex experiments even with only one computer.

However, a simulation study has also its disadvantages. The simulation environment is typically an abstraction of the reality and therefore contains many simplifications. In the case of mobile and wireless networks, which have a very complicated and dynamic environment, the simulation environments are far from being *realistic*. This leads to results that do not fit with real-world measurements.

The most used simulation tools in the area of wireless and mobile networks are the network simulators *ns2* [68], *QualNet* [50], *Opnet* [41], and *OmNet++* [67].

3.3 Emulation

Emulation is a hybrid study environment that consists of two parts: existing hardware and real network layers or parts, and a simulated environment [48,20] and [66,70,24]. Which elements are real and which are simulated depends on the study goals and may differ considerably. However, with emulation it is possible to increase the quality of the study environment by making it more realistic.

There are two approaches of emulation: *network-* and *environment emulation* [20]. In network emulation simulated components can communicate with network protocols realized in the real world. In environment emulation real network protocols are embedded into the simulation environment. In this case the simulation has to provide the same environment as a particular operating system for the network protocol implementation.

For example, the integration of a simulation environment for mobile ad hoc networks with a couple of laptops equipped with wireless network interfaces could enable the realistic computation of radio signals. In the simulation environment, each time a sender transmits a packet to a receiver, the signal strength at the receiver has to be determined. This is subsequently used to decide whether or not the packet has been received correctly. The transmission of the packet would occur between two of the laptops in reality and the result would be made available to the simulation environment.

An important advantage of emulation environments over simulation environments is the possibility of validation against real traffic. The advantage of emulation environments over real world experiments is the possibility of scaling to larger topologies by multiplexing simulated elements on physical resources, e. g., network interfaces [12].

3.4 Virtualization

In recent years virtualization has attracted enormous attention from academia as well as from industry. As a result of this trend there are many commercially as well as publicly available virtualization systems. Typically, dividing the resources of a computer into multiple execution environments is understood as "virtualization". These virtual execution environments or *virtual machines* are isolated from each other: it is not possible for the execution of one virtual machine to adversely affect the performance of another.

The term virtualization is not well-defined since it refers to the abstraction of resources across many aspects of computing. As Figure 3 shows, virtualization environments can be classified into three different classes. The first class of virtualization environments is the *system virtualization* with the virtual machine monitor *inside* the host system, as shown in Figure 3 (a). With this type the virtual machine simulates the complete hardware, allowing an unmodified OS for a completely different CPU to be run. Common examples for this virtualization class are VMware Player, -Workstation, and -Server [69], as well as Virtual PC and -Server [37].

Figure 3 (b) shows the second class of virtualization environments. It is also a type of system virtualization, but in contrast to the previous one, the virtual machine monitor is *underneath* the host operating system. Thus, the virtual machine monitor, in this context also called *hypervisor*, runs directly on the hardware. In general the hypervisor allows multiple operating systems to run, unmodified, at the same time. An example of this virtualization technique is VMware ESX Server [69], and Parallels Workstation and -Desktop for Mac [42].

However, since the x86 architecture used in most PC systems is particularly hard to virtualize and therefore the hypervisor has in general a high complexity, another virtualization approach called *paravirtualization* has recently emerged. Paravirtualization is a virtualization technique that presents a software interface to virtual machines that is similar but not identical to that of the underlying hardware. This requires operating systems to be explicitly ported to run on top of the virtual machine monitor but may enable the virtual machine monitor itself to be simpler. Thus, the virtual machines that run on the virtual machine

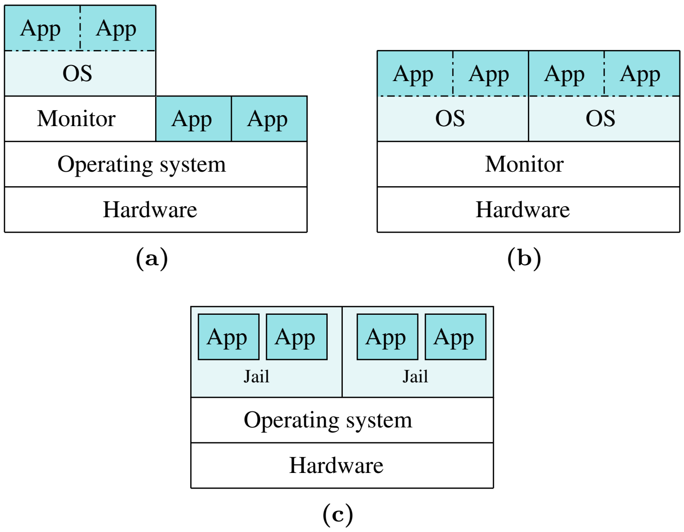

Fig. 3: Virtualization environments: **(a)** *System virtualization* with the virtual machine monitor *inside* of the host system; **(b)** *System virtualization* with the virtual machine monitor *underneath* the host operating system; **(c)** *Operating system-level virtualization*

monitor can achieve higher performance. The most popular example of this approach is Xen [64] with its art of virtualization [6]. Xen is also able to run both paravirtualized and, with the help of hardware virtualization extensions like Intel's Vanderpool [27] or AMD's Pacifica [3], fully virtualized, i. e. an unmodified operating system, as guest operating systems.

The third class of virtualization is the *operating system-level virtualization*. As Figure 3 (c) shows, it virtualizes a physical server at the operating system level, enabling multiple isolated and secure virtualized servers on a single physical server. The guest operating system is the same operating system as the host system, since the same operating system kernel is used to implement the guest environments. Prominent examples are Linux-VServer [55], OpenVZ [60], Solaris Containers [52], Virtuozzo [53], and FreeBSD Jails [54].

Beside the technical aspects the virtualization technique offers the researcher an adequate tool to evaluate communication protocols. With the aid of virtualization, it is possible to create several virtual machines on a single host system. Each virtual machine can run a separate operating system and hence represents an entire computer system. By coupling several virtual machines over the network, it is possible to create a whole virtual network of virtual machines. Furthermore, by extending the virtual network via emulation techniques it is possible to emulate a whole wireless network. The most important advantage of virtual environments is, that the development of the software can be done on the real machine, tested on the virtual network of virtual machines, and later be installed without any modifications on the real testbed.

3.5 Real Testbeds

The best environment to study a system is to conduct experiments on existing implementations. Typically, this is done by prototype implementations. The results and conclusions can be easily transferred to real world environments, since the prototype represents a high degree of the real environment.

However, in the case of distributed and mobile networks, it is very difficult to conduct experiments. The researcher has only limited control over the environment, since there are many influences from the study environment, e. g. interference with production networks. Experiments are typically difficult to repeat, and the experiment setups are restricted in size as well as in complexity. It is also very expensive to conduct experiments in the real world from the hardware point of view as well as from labor intensity. Last but not least, these kind of experiments are limited to existing technologies.

3.6 Summary

The characteristics of the discussed environments are depicted in Table 1. We focused on the following categories of each of the discussed study environments:

Applicability: Evaluates the degree of transferability of the results, conclusions, and the study environment into the real world. A low evaluation in this category asks for special attention when conclusions are drawn from experiments and transferred into the real world.

Repeatability: Rates how straightforward the repetition of a given experiment in that study environment is. The higher the grade in this category the easier is the reliable repetition of an experiment.

Controllability: Assesses the degree of control the researcher has over the study environment as well as the studied subject. In the case of wireless and mobile networks the environment has a huge influence on the overall performance, e. g. the radio propagation is highly influenced by the objects around and between the communicating peers.

Maintainability: Describes the ability to maintain the evaluation environment. In other words how much effort is necessary to keep the system runnable.

Scenario creation: Describes the freedom in creating different experiment scenarios. The network topology, the number of nodes, and the number of parallel connections are elements of an experiment scenario. Furthermore, the environment in which the network is created has to be considered, too.

Scalability: Assesses the feasibility of large scale experiments with respect to the number of nodes in the network, the experiment duration, and the number of network connections during the experiment.

Duration: Describes the experiment time. Variable means, that experiments can be conducted over long periods of time. This allows the study of the system over a long time, e. g. steady state behavior of the system. In contrast, realtime means that experiments are conducted in real-world time. In the latter case the experiment duration is more restricted than in the first case.

Cost: Evaluates the cost of experiments. The cost is related to hardware and software costs.

In Table 1 we have only three categories evaluated in the case of *theoretical analysis*. The other categories do not restrict the environment, since it depends heavily on the modeling capabilities of the researcher, e. g. scalability is not an issue here. Similar arguments could be given for the other categories, too. In

Table 1: Overview of the characteristic of environments for wireless networks.

Characteristic	Environments				
	Theoretical Analysis	Simulation	Emulation	Virtualization	Real Testbeds
Applicability	poor	low	middle	high	high
Repeatability	–	high	low	low	poor
Controllability	high	high	middle	middle	poor
Maintainability	–	high	middle	middle	poor
Scenario creation	–	simple	middle	middle	complex
Scalability	–	high	middle	middle	low
Duration	–	variable	realtime	realtime	realtime
Cost	–	low	middle	middle	high

summary, we can state that a theoretical analysis of a complete wireless mesh network is not possible, but can only be done for particular components of the whole network. This environment provides a high degree of control and abstraction and at the same time a poor applicability of the results and conclusions. The *simulation* combines low cost with high flexibility for different types of network studies. The most important disadvantage is the limited applicability of results to the real world. The *virtualization* provides a healthy tradeoff between maintainability, scalability, and applicability. From our point of view, virtualization has some inherent advantages. The virtual machine allows the development of code that is portable to the real mesh nodes of the testbed. The highest degree of applicability and therefore transferability of results, conclusion, and system environment is given in the case of *real testbeds*. The main disadvantage of this environment is its low scalability and the complexity in experiment scenario generation.

When designing experiments to study a particular performance parameter of wireless mesh networks it is important to have an idea which degree of realism can be expected from the study environments. In Table 2 we have summarized them with respect to networking layers. This helps to determine which of the environments provides the researcher with the required degree of realism.

Application Layer: In the application layer data is produced which has to be transmitted over the network. The data generation is based on the users' behavior, the used application, and the network protocols which are used to transmit the data. In essence, the users are only interested in the performance results of this layer, since they do typically interact with this layer.

Transport Layer: On the transport layer we typically find TCP and UDP which are responsible that data from the application layer is successfully transmitted. UDP offers an unreliable and simple transmission service. TCP in contrast offers a reliable and connection oriented service. Most applications use TCP on the transport layer.

Network Layer: On the network layer we find the IP protocol. It is responsible for the transmission of packets from the transport layer over heterogenous networks. One of the main tasks is also to provide nodes with IP addresses.

Data Link Layer: The data link layer is responsible for the transmission of IP datagrams hop-by-hop from one node to a neighbor node. This includes the

Table 2: Degree of realism of networking layers in various study environments.

Layer	Environments				
	Theoretical Analysis	Simulation	Emulation	Virtualization	Real Testbeds
Application	–	low	high	high	high
Transport	–	low	middle	high	high
Network	–	low	middle	high	high
Data Link	–	high	middle	middle	high
Physical	–	high	middle	low	high

control of the transport medium by appropriate protocols. In case of wireless networks there are prominent problems on this layer like the *hidden station problem* and the *exposed terminal problem* which have to be addressed.

Physical Layer: The physical layer is responsible for the physical transmission of all data which comes from the upper layer. The binary data has to be transformed so that it can be transmitted over the communication medium, i. e. in the case of wireless networks a bit has to be transformed into radio signals.

The *theoretical analysis* approach does not provide any realistic instances of the network layers. In contrast a *real testbed* provides realism on all layers. *Simulation* typically provides a high degree on the data link layer and physical layer. The upper layers are typically simplified, e. g. the input from the application layers are generated according to statistical distributions. The degree of realism in *emulation* depends heavily on the parts which are represented by real hardware and software. In the case of *virtualization* the upper layers are real, since the virtualized machine and the operating system provide all necessary functionalities. However, if virtual machines are coupled via a network the physical layer may have low realism, if both virtual machines are run on the same physical computer.

4 MCG-Mesh – A Hybrid Testbed for WMNs

In this Section we present our project *MCG-Mesh* at the Department of Computer Science, Informatik 4, RWTH Aachen University. The goal of this project is twofold. From the scientific point of view the goal is to build a large and scalable mesh network to conduct various networking studies. From the application point of view the goal is to provide the members of the Computer Science Department and the students with a simple and comfortable way to get high bandwidth network access anywhere in the computer science center.[3]

4.1 Motivation for a Hybrid Testbed

In Section 3 we discussed several environments for the study of wireless mesh networks. Besides the real testbed environment, two other approaches have a high degree of realism: *emulation* and *virtualization*.

[3] See http://www-i4.cs.rwth-aachen.de/mcg/projects/mcg-mesh for online information.

Within the emulation concept, the environment emulation provides the higher degree in realism compared to the network emulation, since the simulated components behave as they would in a real world environment. However, this requires the imitation of a particular existing operating system, which is difficult to achieve. The competing approach is virtualization, which replicates a whole machine or a particular operating system. Therefore, there is no need to simulate a machine, operating system, or network.

We will review the development and maintenance process for a testbed to show the advantages of our approach. The process can be subdivided into two phases. In the first phase, the testbed is designed and realized. In this step hardware and software decisions are made. The most important software decision is the selection of the operating system, which runs on the mesh routers. For this purpose Linux and Unix derivatives are mostly used. Subsequently, the design and implementation of networking protocols and tools starts. This process is iterative, i. e. implementation, debugging, and testing. The development and test environment should provide a high degree of realism, which is achieved best by real hardware and standard software components. Subsequently, the second phase starts, which is mainly focused on performance evaluation and research experiments. New versions of the software have to be distributed frequently onto all testbed nodes. This can be a labor intensive task. After distributing the software validation tests have to be run to ensure correct installation on all nodes. In case a failure occurs debugging information has to be collected and analyzed.

The described process of software development, distribution, debugging, testing, and redistribution of code is very complex and labor intensive. Virtualization is an approach which can be used to minimize the labor and to reduce the failure sources. In the virtualization approach, the hardware as well as the software is the same as on the wireless mesh routers. The overhead for implementation, debugging, and testing is therefore minimized. Especially in the case of performance and compatibility testing the virtualization approach provides a simple and easy way to build a scalable network of virtual mesh routers. Furthermore, the virtual mesh routers can be easily coupled with the real mesh routers to enhance the number of mesh network nodes. Therefore, it supports the researcher during the development phase as well as the testing phase.

In essence, a hybrid testbed environment consisting of the combination of virtualized and real hardware provides the best support for studying wireless mesh networks. In the following sections, we describe in detail the software, hardware, and virtualization component of our testbed.

4.2 System and Network Architecture

Figure 4 depicts the general system and network architecture of our testbed. In accordance with our previous considerations our testbed is realized using two different components: a virtualization environment and a real testbed.

Since we wanted to measure and evaluate the performance of different mesh architectures, our testbed is built in a way that allows us to easily change important parameters. For example, we are able to define which mesh router should act as a gateway, and we can enable or disable the routing functionality of the clients (routing clients vs. non-routing clients, see Section 2.1). All our mesh

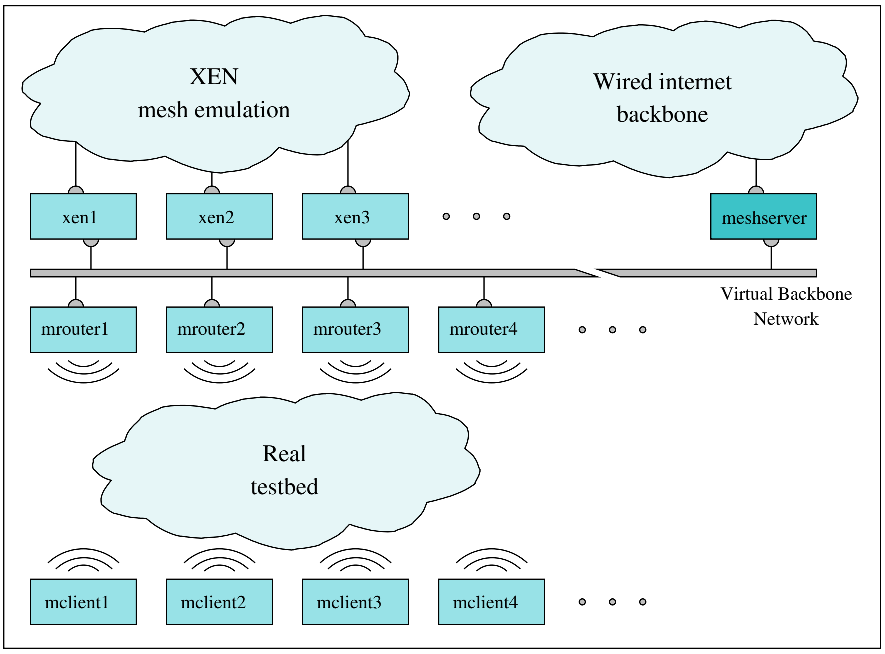

Fig. 4: Architecture of the MCG-Mesh

routers and Xen instances are connected using a common network, the *virtual backbone network*. This network is used to boot and configure the attached nodes as well as audit trail processing. The client data is not transmitted using this backbone network, but it is forwarded in a multi-hop fashion using the (virtual) wireless interfaces.

As we have seen in Section 3.6 a main disadvantage of real testbeds as well as virtualization environments is the high maintenance cost. To minimize this maintenance cost we use a *central configuration approach*. As Figure 4 shows, a central server called *meshserver* is integrated into the testbed. It has two responsibilities, the *"source functionality"* and the *"drain functionality"*. Firstly, "source functionality" describes the services that the server provides to the attached nodes. The most important service is to provide a single operating system image for all nodes via the network. Therefore, the basic setup for each node is the same for each of the two testbed parts, e. g. the Linux kernel and the modules and drivers. Another important service provided by the meshserver is internet access, which is required for the mesh gateways.

Secondly, the meshserver provides a "drain functionality" meaning that all important information about the real and emulated WMN is gathered and stored at the central server. Typical information are system log and SNMP messages or measurement results, which are stored in a database. This approach of a central log and information server enables us to detect any problem in our testbed.

Another advantage that a central configuration and log server offers is the simplified scenario creation as well as improved controllability. Since we store configuration files on one central server we can easily realize any WMN architecture (see Section 2.1) or WMN application scenario (see Section 2.4). For example, a routing mesh client can easily be reconfigured to a non-routing mesh client by disabling the routing functionality.

4.3 Testbed Realization

The *MCG-Mesh* testbed is deployed in the Computer Science Center, RWTH Aachen University. The center consists of one four- and two three-story buildings, which are interconnected. The mesh routers are located in different offices at different floors. Currently, we have installed two mesh routers per floor. Therefore, the four mesh routers of two adjacent floors generate a fully connected graph. None of the mesh routers have a line-of-sight to their respective neighbors.

As mentioned in Section 4.2 MCG-Mesh consists of two parts: a virtualization environment and a real testbed. In the remainder of this section we describe the used hard- and software for each of these parts.

Hardware Currently, the real testbed part of MCG-Mesh consists of 10 identical mesh routers. Each mesh router consists of a Single Board Computer (SBC), two IEEE 802.11a/b/g wireless interface cards, two omni-directional antennas and one 256 MB compact flash card. The SBC is a *WRAP.2C* board from *PC Engines* [43], which is a x86 computer on the basis of the 233 MHz AMD Geode SC1100 CPU. Besides the slots for the two mini-PCI cards and the CF card the board offers 128 MB RAM, one 100 Mb/s ethernet port, and one RS 232 serial port. The board does not have any VGA, keyboard or mouse connector. All communication with the mesh router goes over a serial line, or via network connections. The WRAP.2C is particularly suitable for the usage as a wireless mesh router since the board represents a small, quiet, and cost-effective SBC platform.

Each mesh router is equipped with two identical IEEE 802.11a/b/g wireless interface cards based on the *Atheros AR5213 XR* chip-set [5]. We use the first wireless card for router-to-router and the second wireless card for router-to-client communication. Other variations, such as interface channel assignment are conceivable, but currently not implemented. In order to separate the router-to-router communication from the router-to-client communication the first radio interface operates in 802.11a and the second radio interface in 802.11g mode. The former transmits at 50 mW, the latter at 100 mW. Both wireless cards operate in the standard IBSS (or "ad hoc") mode, RTS/CTS is disabled. The cards are connected to tri-band 5 dBi omni-directional antennas [29]. All mesh routers share the same pair of 802.11a/g channels and the same ESSID pair, that is one ESSID for the 802.11a channel (router-to-router communication) and one ESSID for the 802.11g channel (router-to-client communication).

The virtualization environment of our testbed consists of 7 standard Pentium 4 PC with 512 MB RAM. Even with this small amount of RAM it is possible to run about 5 virtual machines per host.

Software As mentioned in Section 4.2 we want to implement an isolated, homogenous and controlled virtual backbone network. To realize this, we have chosen the open source VPN package *OpenVPN* [59]. OpenVPN implements IP Layer 2 and 3 secure network extension using the industry standard SSL/TLS protocol, supports flexible client authentication methods based, e. g. on certificates, and allows user or group-specific access control policies.

The second subgoal is a central configuration. As we described in Section 4.2 one operating system image is provided to all nodes via the network. We use the

Network File System (NFS) protocol for this purpose. As a consequence, we are not bound to using a stripped-down Linux distribution that fits onto the routers' compact flash cards. Therefore, we have chosen to run a standard Gentoo Linux distribution. Gentoo differs from other Linux distributions in that it is source-based, all packages are usually compiled locally before installation. Due to this system new versions are quickly available, the source code of all installed packages is promptly accessible and above all our own patches can easily be integrated. Except for some alterations in the boot process, which are described below, nodes are running standard Linux software.

The central audit trail processing is realized with a combination of logging and monitoring. The logging task is performed by *syslog* [33], the de facto standard for forwarding log messages in an IP network. It provides a centralized, securely stored log of all network devices and incorporates many powerful features, including filtering based on message content, as well as customisable data mining and analysis capabilities. The other task—monitoring—is performed by *snmp* [1]. With SNMP we do not only have the possibility to retrieve information from the mesh routers but also a convenient way to make changes to the managed testbed, e. g. to change the wireless channel or the transmitting power.

On the one hand we apply the *madwifi-ng* driver [34] to implement the wireless mesh architecture in the real testbed. One of the most interesting features of madwifi is the "Virtual Access Point (VAP) mode", which allows the operation of multiple virtual wireless devices, concurrently running in different modes. In particular, it is possible to run a first VAP in the access point mode and a second VAP in the ad hoc mode. Using this feature we are able to connect both routing and non-routing mesh clients.

On the other hand the nodes in our virtual testbed are driven by *Xen* [64], which was initially developed by the *University of Cambridge*. Considering the options described in Section 3.4 operating system-level virtualization can be eliminated since our nodes need to have separate network stacks. One of the reasons for choosing Xen is the fact that it employs the most efficient approach to system virtualization, paravirtualization. Furthermore, it is an open-source project. It is likely to be included into the standard Linux Kernel in the near future and CPU manufacterers actively support its development. Thus, the upcoming integration of hardware virtualization support as well as Xen development in general is likely to make quick progress.

Host and guest systems, called domains, are equipped with a modified version of the Linux Kernel. Domain administration is carried out in the host system, *dom0* in Xen terminology. On a given *dom0* the maximum number of guests, or *domU*s, is bounded by the amount of physical memory and the available processing power.

To emulate a wireless medium we use a combination of the advanced networking features of the Linux Kernel. At the core we have a virtual network that exists on top of our backbone as illustrated in Figure 4. For that, we utilize the tunneling protocol *Generic Routing Encapsulation (GRE)* [21]. It emulates a broadcast medium on top of an existing IP network by using a multicast address for its broadcast traffic. To control who can communicate with whom, standard packet filtering provided by *iptables* [57] is employed. We use Linux's versatile traffic control utility *tc* [32] to emulate wireless medium characteristics. Packet loss and packet delay can be generated using user-supplied distributions and can be combined with hierarchical traffic shaping, e. g. token bucket filters. Values and distributions for these parameters can be derived from observations in reality, e. g. from our mesh routers and mesh gateways.

Currently, a modified version of the *Dynamic MANET On-demand (DYMO)* routing protocol implementation [13] from the *University of Murcia* [65] and the *Optimized Link State Routing (OLSR)* [14] protocol implementation from the *OLSR.org Project* [58] are employed. We made this choice since the two routing protocols are typical representatives of the two routing philosophies in MANETs: reactive and proactive routing.

As previously mentioned our testbed nodes are booted via the network. The vital parts of this process include getting an IP configuration[4] and a kernel to boot. We utilize a combination of EtherBoot [19] and PXELinux [4] for this purpose. EtherBoot obtains the IP configuration from a DHCP server and provides a standardized environment, so-called *Preboot Execution Environment (PXE)*, to PXELinux. PXELinux in turn uses this environment to access the network and to fetch a kernel image from the supplied server address. The following overview of the boot process provides the context for the software described in this section.

1. **Power on:** This step depends on the node type:
   - **Mesh Router:** The BIOS loads EtherBoot from the router's flash memory, which fetches PXELinux via the network from the central server.
   - **Xen host (dom0):** Either PXELinux is loaded directly by a PXE enabled network card, if present, or indirectly via a CD equipped with Etherboot.
   - **Xen guest (domU):** Since a Linux kernel image is provided directly by dom0, these nodes skip the next step and jump directly to kernel boot.
2. **PXELinux:** A configuration file that contains image locations and boot parameters is fetched via the network. Its name is derived from the local IP or MAC address. Subsequently, a Linux kernel image and an initial ramdisk are loaded.
3. **Kernel boot:** The Linux kernel loads our initial ramdisk and invokes an initalization script. The script starts `openvpn` which establishes a connection to the backbone network. After retrieval of an IP configuration via DHCP, it mounts the central Linux image via NFS and executes its `init` system.
4. **Linux boot:** The Gentoo Linux `init` system will start the core services. Last but not least a hand-made script will load and configure the madwifi driver running the WLAN cards and the monitoring daemon, and will configure the Linux IP stack according to the current measurement setup.

5 Related Work – Existing Testbeds

In this Section we will present wireless mesh networking projects, which have the aim to create a testbed. We will discuss their goals, advantages, disadvantages, the used software, and the hardware on which the mesh nodes are based on. The discussion of the projects is organized according to the requirements presented in Section 3.5. The discussed list of projects is not exhaustive. The goal is to

[4] Besides IP address and subnet mask this might also encompass a gateway and DNS server for instance.

show the various interpretations of wireless mesh networks and the different implementations.

5.1 MIT Roofnet

The *MIT Roofnet project* [8,35] consists of 37 nodes based on PCs running Linux and the Click [30] modular router. Each node has an IEEE 802.11b network interface and an omni-directional antenna. The goal of the project is to provide Internet access to the students on the MIT Campus. The Roofnet nodes are run by volunteering students. All Roofnet nodes are running on the same channel. There are a total of 4 gateways which provide Internet access to Roofnet. The Roofnet mesh routers can allocate IP addresses via DHCP for the mesh clients and provide them with access to the Roofnet as well as the Internet. When mesh clients access the Internet the Roofnet mesh routers act as a network address translator (NAT). Roofnet's routing protocol is denoted as Srcr which is a source routing protocol similar to DSR. In its metric it is similar to the routing protocol in MCL, see Section 5.2.

The advantage of this project is its easy and simple architecture. It is enough to install the Roofnet nodes. They will build a static wireless mesh network and provide Internet access to the mesh clients. Yet, there are also some disadvantages of Roofnet. First of all, a mesh client gets its IP address from a Roofnet mesh router. This address is only valid for the connection to this router. When the mesh client moves around and leaves the service range of that router the mesh client will loose its current connection. So Roofnet does not provide roaming of mesh clients. Furthermore, it supports only non-routing mesh clients, i. e., there is always one wireless link from a mesh client to a Roofnet mesh router. This restricts the range of Roofnet only to places where Roofnet mesh routers can offer service.

The *Berlin RoofNet Project* [25] is somehow the German counterpart of the MIT Roofnet project. The goal of the project is similar to that of the MIT Roofnet project, namely to provide Internet access over a set of static wireless nodes which construct a wireless mesh network. The used hardware and software is similar to that of the MIT project; the advantages and disadvantages are also similar.

5.2 Microsoft Mesh Networking

Microsoft Research is working on a community mesh network [38]. The goal of the project is to enable the building of a community mesh network, which allows the residents of a neighborhood to share existing Internet gateways. The core of the project is the *Mesh Connectivity Layer (MCL)*, which is a virtual network driver for Windows from the technical point of view. After installing MCL the user has an additional network interface which represents the wireless network as a normal network link. Each node applying MCL can route data for other nodes in the mesh network. For this, MCL uses a modified version of DSR [28] which is called Link Quality Source Routing (LQSR) [18]. From the network architecture point of view, MCL is located between the Network Layer and MAC Layer, hence to be placed on layer 2.5. MCL uses MAC addresses for routing. With an appropriate configuration of Windows it is also possible to enable Internet connectivity through a gateway.

MCL is available for academic institutions in source code. After registration Microsoft sends an academic resource toolkit with all necessary products to compile and run MCL. There are no projects known to us which use MCL as the basis for their wireless mesh network.

5.3 UCSB MeshNet

Belding-Royer et al. [46] run a mesh network project called *MeshNet* [63] at the University of California at Santa Barbara, USA. In MeshNet each mesh router consists of two Linksys WRT54G wireless devices. One of the devices is used for routing within the mesh network and the other device is used for the management of the router. The WRT54G wireless devices run OpenWRT [61], a special Linux distribution, and a modified version of AODV routing protocol. The important difference is the metric used. Instead of the shortest-hop as the routing metric they use a reliability-based routing metric.

The research group has developed some management tools for wireless mesh network testbeds. The *Testbed Configuration Tool* configures a device and works in two steps. In the first step, the hardware and software settings of a device are collected. In the second step, the configuration according to the network environment is done. The *Interference Meter* collects data, which allows the estimation of the interference among the mesh routers. For this a simple approximation is used. Each mesh router scans all possible communication channels and counts the number of nodes which use that channel. Thus, the number of nodes using a particular communication channel is used to estimate the interference. The collected data is sent to a server that provides the testbed operator with this information. The third tool is the *Network Monitoring Tool* that collects information like the number of packets sent and received, topology data, routing table information, and quality of various links. The collected data is sent to a server, which in turn provides them to the testbed. The *Topology Control Tool* allows the construction of virtual topologies in the testbed. This is very interesting, since typically physically moving nodes is very labor intensive. The idea is based on selectively dropping packets from nodes which are not in the testbed. To this end, the nodes which are actually not in the virtual testbed are marked. The other nodes can filter their packets and drop them.

5.4 WMN Testbed at Purdue University

Hu et al. run a wireless mesh project at Purdue University, USA, which is called *Mesh@Purdue* [45]. It consists of 30 nodes. A so-called MAP mesh router is a small form-factor desktop equipped with two wireless interface cards and a wired ethernet network interface. The latter is used for management purposes. Besides the MAP mesh routers, there are also some Laptop and iPAQ PDAs which are used as hosts. These hosts can access the Internet over the mesh network. The research group deploys AODV and OLSR as routing algorithms within the mesh network. The AODV version used is modified to support the ETX [16,17] routing metric.

5.5 WMNs Research at Georgia Tech

Akyildiz et al. [2] run a wireless mesh network project at the Broadband and Wireless Network (BWN) Lab, Georgia Institute of Technology, USA, called *BWN-Mesh*. It consists of 15 nodes. The goal of the project is to study various performance metrics of wireless mesh networks, e. g. the effects of inter-router distance, backbone placement and clustering. Furthermore, existing protocols are re-investigated to review their performance in the testbed, e. g. end-to-end delay and throughput.

5.6 WMN at Carleton University

Kunz et al.[11] run a wireless mesh network project at Carleton University, Canada. Each mesh router is equipped with two wireless interface cards. One of these interfaces is IEEE 802.11a/g compliant and the other one is IEEE 802.11b compliant. The first is used for the communication within the wireless mesh network among the wireless mesh routers and the latter is used for the communication with the clients. The wireless mesh network provides Internet access to the clients.

The wireless mesh routers are based on Intel IXP425 series XScale computers with two Mini-PCI slots, two 10/100 Base-TX Ethernet channels, and two RS232 serial ports for management and debug purposes. A mesh router includes 64 MB RAM, and 16 MB of Flash memory. Thus, the mesh router does not need a hard drive. The mesh routers are running μClinux [56], a popular embedded Linux distribution, and QoS OLSR from Communications Research Centre (CRC) [15] as the routing algorithm. To provide clients with addresses DHCPv6 is used. Thus, the wireless mesh network deploys only IPv6. If clients need to access an IPv4 network, e g. the Internet, the packets are tunneled by applying the Dual Stack Transition Method (DSTM) [9].

5.7 Hyacinth

Hyacinth [51,47] is the wireless mesh network project at the State University of New York, USA. Each Hyacinth node is a small form-factor PC running Windows XP and is equipped with three IEEE 802.11a wireless interfaces. The mesh nodes obtain IP addresses by using a two-step method which is based on DHCP. In the first step, a mesh node allocates a temporary IP address from the reserved range 192.168/16. In the second step, a unique IP address is obtained from a global DHCP server, which is placed in the wired network. Each mesh node also acts as a local DHCP server and can assign IP addresses to mobile stations. Each mesh node receives a range of IP addresses from the global DHCP server. This ensures that each mobile station has an unique IP address in the whole mesh network. Furthermore, roaming of mobile stations is supported, since all mesh nodes act also as a home/foreign agent similar to Mobile IP. The goal of the research group is to conduct research in the area of channel assignment among the wireless mesh nodes and routing.

Table 3: Overview of wireless mesh network testbed projects.

Project	Nodes	802.11	Software	Routing		Roaming	Config	MANET
				Layer	Protocol			
MIT Roofnet	37	b/g	Linux, CMR	RL	Srcr	–	x	–
Microsoft	23	a/b/g	Windows(CE)	MAC	MCL	–	x	x
UCSB Meshnet	25	a/b/g	OpenWrt	IP	AODV	–	x	–
Purdue	30	a/b/g	—	IP	AODV, OLSR	–	x	–
Georgia Tech	15	b/g	—	—	—	–	–	–
Carleton Univ.	??	a/g	μCLinux	IP	OLSR	–	x	–
Hyacinth	10	a	Windows XP	—	—	x	x	–
MCG-Mesh	20	a/b/g	Linux	IP	DYMO, OLSR	x	x	x

5.8 Summary

The properties of the discussed projects are depicted in Table 3. We give a brief summary by collecting all the important information about projects. We collected the following information for each of the projects:

Project: The name of the project. This is either the name of the project taken from the project description or the name of the university.

Nodes: The number of nodes in the wireless mesh network testbed. The values here are obtained either from the project website or from the latest publication of the research group.

802.11: The technology which is used for the communication. This is a set of the available technologies including IEEE 802.11a, IEEE 802.11b, IEEE 802.11g.

Software: The software which is used to run the wireless mesh network testbed, i. e., operating system and if important additional software.

Routing: This property denotes the protocol stack layer on which the routing runs. This is either MAC or IP. In the case of MAC, the MAC-addresses are used to perform routing. In the case of IP, IP addresses are used to perform routing.

Roaming: If checked, a mesh client can be configured by one mesh router and can move around without losing the connection to the mesh network. For this a handover of mesh clients among mesh routers must be implemented.

Config: If checked, the testbed supports automatic configuration of the nodes, i. e., the routers as well as clients.

MANET: If checked, the testbed supports the connection of autonomous mobile ad hoc networks, i. e. the testbed provides multi-hop relaying of packets also within the mesh clients.

Table 3 summarizes the information. All wireless mesh networking testbeds deploy IEEE 802.11x compliant technology. This holds for the communication among the wireless mesh network routers as well as for the communication between these routers and mobile stations. Currently, there is no project which uses the combination of IEEE 802.11x and IEEE 802.16. The latter could be deployed for the communication among the static wireless mesh routers.

The largest testbed regarding the number of nodes in the wireless mesh network is the MIT Roofnet with 37 nodes.

Most testbeds use either Linux or a derivative as the operating system on the mesh nodes to perform the complex tasks of a router. This is due to the open architecture of Linux and its availability in source code. Two projects deploy the Click Modular Router (CMR) for the routing purposes. Only the Microsoft Research project and the Hyacinth testbed are deploying Windows as operating system. In case of Microsoft Research the reason is obvious. In the case of the Hyacinth testbed we did not find any reason why Windows XP was selected.

Four projects perform routing on the IP level, three projects on the MAC level, and for two projects we could not figure out on which level the routing is done. AODV, OLSR, DSR, and DYMO or variants are used as routing protocols. Most projects do not consider roaming of mesh clients; only Hyacinth and MCG-Mesh do.

The integration of independent MANETs with the wireless mesh network testbed, or the establishing of a MANET with the same architecture/software of the testbed without the existence of static wireless mesh routers is only considered in two projects. In the case of Microsoft's MCL this lies in the nature of the architecture, since the goal is to support neighborhood networks without a maintenance layer. In the case of MCG-Mesh the architecture considers also routing mesh clients, which can independently establish a MANET.

6 Conclusions and Future Work

This paper presented an overview of existing methodologies and technologies to study wireless mesh networks. As WMNs become more and more popular the need for reliable and applicable results grows. The numerous possibilities to get to results were presented here and we evaluated them according to the main question: How reliable are the results when applied to a real world product?

We first gave an overview of the architecture of wireless mesh networks. We described the seven key properties of all WMNs and identified three classes of wireless mesh networks. This architectural overview contributed to the establishment of a common terminology throughout this work and the authors hope that it will lead to a widely accepted terminology in the near future.

According to recent publications 76% of studies in the area of wireless networks are based on simulation. Although simulation environments provide the researcher with many advantages like low cost, flexibility, and controllability they also possess some disadvantages which degrade their usefulness. The prime disadvantage comes from the high dynamic and complexity of mobile and wireless networks, which is caused by the high influence of the environment. It has been shown that the performance of wireless networks in simulation and the real world differ very much. The counterpart to simulation studies are real world testbeds. They provide the same environment for the researcher as it exists in the production world. All results and inferences can be easily transferred to real world systems. However, real world testbeds have other limitations, e. g. typically they do not scale, since it is very hard to set up a large wireless network testbed. A possible solution for this dilemma is to apply a hybrid testbed.

We were able to categorize the existing approaches into eight categories. This allows us to evaluate the value of the produced results. Using this evaluation, a researcher or product developer can now make an educated decision, which method to use for his specific problem.

Based on our evaluation, we introduced the MCG-Mesh. It is a hybrid wireless mesh networking testbed, which consists of real hardware, standard Linux software and a virtualization environment. The former ensures a high degree of realism and enables us to transfer the results and conclusions into the real world. The second part provides us with a flexible environment to develop various networking protocols. We described in detail our hardware and software setup to give other researchers the possibility to build similar testbeds.

In the future we will focus on the integration of various wireless networks, i. e. wireless sensor networks, GSM, UMTS, into our wireless mesh testbed. For this several protocols on different layers have to be considered. Particularly, we are interested in the development and performance evaluation of *Autoconfiguration*, *TCP*, *Load Balancing*, and *Channel assignment*.

References

1. J. Case adn R. Mundy, D. Partain, and B. Stewart. Introduction and applicability statements for internet standard management framework. RFC 3410, December 2002. Available from: http://www.ietf.org/rfc/rfc3410.txt.
2. Ian F. Akyildiz, Xudong Wang, and Weilin Wang. Wireless mesh networks: a survey. *Computer Networks*, 47(4):445–487, March 2005.
3. AMD, Inc. AMD's virtualization solutions – Pacifica. Available from: http://enterprise.amd.com/us-en/Solutions/Consolidation/virtualization.aspx.
4. H. Peter Anvin. PXELinux. Available from: http://syslinux.zytor.com/pxe.php.
5. Atheros Communications, Inc. AR5004 WLAN chipset product bulletins. Available from: http://www.atheros.com/.
6. Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, and Andrew Warfield. Xen and the art of virtualization. In *Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP'03)*, pages 164–177. ACM Press, October 2003.
7. Stefano Basagni, Marco Conti, Silvia Giordano, and Ivan Stojmenovic, editors. *Mobile Ad Hoc Networking*. John Wiley & Sons, Inc., New York, NY, USA, August 2004.
8. John Bicket, Daniel Aguayo, Sanjit Biswas, and Robert Morris. Architecture and evaluation of an unplanned 802.11b mesh network. In *Proceedings of the 11th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'05)*, pages 31–42. ACM Press, August 2005.
9. J. Bound, L. Toutain, and JL. Richier. Dual stack ipv6 dominant transition mechanism (DSTM). Internet Draft, October 2005. Available from: http://www.watersprings.org/pub/id/draft-bound-dstm-exp-04.txt.
10. Raffaele Bruno, Marco Conti, and Enrico Gregori. Mesh networks: Commodity multihop ad hoc networks. *IEEE Communications Magazine*, 43(3):123–131, March 2005.
11. Carleton University. Wireless mesh networking. Available from: http://kunz-pc.sce.carleton.ca/MESH/index.htm.
12. Mark Carson and Darrin Santay. NIST Net: a Linux-based network emulation tool. *ACM SIGCOMM Computer Communication Review*, 33(3):111–126, July 2003.
13. I. Chakeres and C. Perkins. Dynamic manet on-demand (DYMO) routing. Internet Draft, June 2006. Available from: http://www.watersprings.org/pub/id/draft-ietf-manet-dymo-04.txt.
14. T. Clausen and P. Jacquet. The optimized link state routing protocol (OLSR). RFC 3626, October 2003. Available from: http://www.ietf.org/rfc/rfc3626.txt.
15. Communications Research Centre. CRC OLSR. Available from: http://www.crc.ca/en/html/manetsensor/home/software/software.

16. Douglas S. J. De Couto, Daniel Aguayo, John Bicket, and Robert Morris. A high-throughput path metric for multi-hop wireless routing. In *Proceedings of the 9th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'03)*, pages 134–146. ACM Press, September 2003.
17. Richard Draves, Jitendra Padhye, and Brian Zill. Comparison of routing metrics for static multi-hop wireless networks. In *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM'04)*, pages 133–144. ACM Press, September 2004.
18. Richard Draves, Jitendra Padhye, and Brian Zill. Routing in multi-radio, multi-hop wireless mesh networks. In *Proceedings of the 10th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'04)*, pages 114–128. ACM Press, September 2004.
19. EtherBoot Project. EtherBoot. Available from: http://www.etherboot.org/.
20. Kevin Fall. Network emulation in the Vint/NS simulator. In *Proceedings of the 4th IEEE Symposium on Computers and Communications (ISCC'99)*, pages 244–250. IEEE Computer Society Press, July 1999.
21. D. Farinacci, T. Li, S. Hanks, D. Meyer, and P. Traina. Generic routing encapsulation (GRE). RFC 2784, March 2000. Available from: http://www.ietf.org/rfc/rfc2784.txt.
22. Gianluigi Ferrari and Ozan Tonguz. *Ad Hoc Wireless Networks: A Communication-Theoretic Perspective*. John Wiley & Sons, Inc., New York, NY, USA, May 2006.
23. Förderverein freie Netzwerke e.V. Freifunk.net. Available from: http://freifunk.net/.
24. Shashi Guruprasad, Robert Ricci, and Jay Lepreau. Integrated network experimentation using simulation and emulation. In *Proceedings of the 1st International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TRIDENTCOM'05)*, pages 204–212. IEEE Computer Society Press, February 2005.
25. Humboldt University Berlin. Berlin Roof Net Project. Available from: http://sarwiki.informatik.hu-berlin.de/BerlinRoofNet.
26. Information Week. Cisco leaps into mesh. Available from: http://www.informationweek.com/news/showArticle.jhtml?articleID=174400255.
27. Intel Corporation. Intel virtualization technology – Vanderpool. Available from: http://developer.intel.com/technology/computing/vptech/.
28. David B. Johnson, David A. Maltz, and Yih-Chun Hu. The dynamic source routing protocol for mobile ad hoc networks (DSR). Internet Draft, July 2004. Available from: http://www.watersprings.org/pub/id/draft-ietf-manet-dsr-10.txt.
29. Joymax Electronics Co., Ltd. Replacement antenna – TW-614. Available from: http://www.joymax.com.tw/.
30. Eddie Kohler, Robert Morris, Benjie Chen, John Jannotti, and M. Frans Kaashoek. The click modular router. *ACM Transactions on Computer Systems*, 18(3):263–297, August 2000.
31. Stuart Kurkowski, Tracy Camp, and Michael Colagrosso. MANET simulation studies: the incredibles. *ACM SIGMOBILE Mobile Computing and Communications Review*, 9(4):50–61, October 2005.
32. LARTC Project. Linux advanced routing & traffic control. Available from: http://www.lartc.org/.
33. C. Lonvick. The BSD syslog protocol. RFC 3164, August 2001. Available from: http://www.ietf.org/rfc/rfc3164.txt.
34. Madwifi Project. Madwifi – multiband atheros driver for wireless fidelity. Available from: http://madwifi.org/.
35. Massachusetts Institute of Technology. MIT RoofNet. Available from: http://pdos.csail.mit.edu/roofnet/doku.php?id=centralsq.
36. Merriam-Webster OnLine. "experiment". Available from: http://www.webster.com.
37. Microsoft. Virtual PC and -server. Available from: http://www.microsoft.com.
38. Microsoft Research. Mesh connectivity layer (MCL). Available from: http://research.microsoft.com/mesh/.
39. Microsoft Research. Mesh networking summit 2004: Making meshes real. Available from: http://research.microsoft.com/meshsummit/techprogram.aspx.
40. C. Siva Ram Murthy and B.S. Manoj. *Ad Hoc Wireless Networks: Architectures and Protocols*. PTR Prentice-Hall, Upper Saddle River, NJ, USA, May 2004.
41. OPNET Technologies, Inc. OPNET – optimum network engineering tools. Available from: http://www.opnet.com/.

42. Parallels, Inc. Parallels workstation and -desktop for mac. Available from: http://www.parallels.com/.
43. PC Engines. WRAP router platform (Version WRAP.2C). Available from: http://www.pcengines.ch.
44. Charles E. Perkins. *Ad Hoc Networking*. Addison-Wesley, Reading, MA, USA, January 2001.
45. Purdue University. Purdue University wireless mesh network testbed. Available from: https://engineering.purdue.edu/MESH.
46. Krishna N. Ramachandran, Kevin C. Almeroth, and Elizabeth M. Belding-Royer. A framework for the management of large-scale wireless network testbeds. In *Proceedings of the 1st Workshop on Wireless Network Measurements (WiNMee'05)*, April 2005.
47. Ashish Raniwala and Tzi-cker Chiueh. Architecture and algorithms for an IEEE 802.11-based multi-channel wireless mesh network. In *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (Infocom'05)*, volume 3, pages 2223– 2234. IEEE Communications Society Press, March 2005.
48. Luigi Rizzo. Dummynet and forward error correction. In *Proceedings of the USENIX Annual Technical Conference, FREENIX Track*, pages 244–250. USENIX Association, June 1998.
49. S. Ruffino, P. Stupar, T. Clausen, and S. Singh. Connectivity scenarios for MANET. Internet Draft, July 2005. Available from: http://www.watersprings.org/pub/id/draft-ruffino-conn-scenarios-01.txt.
50. Scalable Network Technologies. QualNet. Available from: http://www.qualnet.com/products/qualnet.php.
51. State University of New York at Stony Brook. Hyacinth: An IEEE 802.11-based Multi-channel Wireless Mesh Network. Available from: http://www.ecsl.cs.sunysb.edu/multichannel/.
52. Sun. Solaris enterprise system. Available from: http://www.sun.com/software/solaris/.
53. SWsoft. Virtuozzo. Available from: http://www.virtuozzo.com/.
54. The FreeBSD Project. FreeBSD operating system. Available from: http://www.freebsd.org/.
55. The Linux-VServer Project. Linux-VServer. Available from: http://linux-vserver.org/.
56. The μClinux Project. μClinux – Embedded Linux/Microcontroller Project. Available from: http://www.uclinux.org/.
57. The netfilter.org Project. iptables. Available from: http://www.netfilter.org/.
58. The Olsr.org Project. OLSRd – the olsr.org OLSR daemon. Available from: http://www.olsr.org/.
59. The OpenVPN Project. OpenVPN. Available from: http://openvpn.net/.
60. The OpenVZ Project. OpenVZ. Available from: http://openvz.org/.
61. The OpenWrt Project. OpenWrt – a Linux distribution for wireless routers. Available from: http://openwrt.org/.
62. C.-K. Toh. *Ad Hoc Mobile Wireless Networks: Protocols and Systems*. PTR Prentice-Hall, Upper Saddle River, NJ, USA, November 2002.
63. University of California at Santa Barbara. MeshNet Project. Available from: http://moment.cs.ucsb.edu/meshnet.
64. University of Cambridge. XEN – the xen virtual machine monitor. Available from: http://www.cl.cam.ac.uk/Research/SRG/netos/xen/.
65. University of Murcia. DYMOUM – an implementation of the DYMO (Dynamic MANET On-demand) routing protocol. Available from: http://sourceforge.net/projects/dymoum/.
66. Amin Vahdat, Ken Yocum, Kevin Walsh, Priya Mahadevan, Dejan Kostić;, Jeff Chase, and David Becker. Scalability and accuracy in a large-scale network emulator. In *Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI'02)*, pages 271–284. USENIX Association, December 2002.
67. András Varga. OMNeT++ – a discrete event simulation system. Available from: http://www.opnet.com/.
68. VINT Project. NS-2 – The Network Simulator - ns. Available from: http://nsnam.isi.edu/nsnam/index.php/Main_Page.
69. VMware, Inc. VMware player, -workstation, -server, and -infrastructure. Available from: http://www.vmware.com/.

70. Brian White, Jay Lepreau, Leigh Stoller, Robert Ricci, Shashi Guruprasad, Mac Newbold, Mike Hibler, Chad Barb, and Abhijeet Joglekar. An integrated experimental environment for distributed systems and networks. In *Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI'02)*, pages 255–270. USENIX Association, December 2002.

Aachener Informatik-Berichte

This is a list of recent technical reports. To obtain copies of technical reports please consult http://aib.informatik.rwth-aachen.de/ or send your request to: Informatik-Bibliothek, RWTH Aachen, Ahornstr. 55, 52056 Aachen, Email: biblio@informatik.rwth-aachen.de

1987-01 *	Fachgruppe Informatik: Jahresbericht 1986
1987-02 *	David de Frutos Escrig, Klaus Indermark: Equivalence Relations of Non-Deterministic Ianov-Schemes
1987-03 *	Manfred Nagl: A Software Development Environment based on Graph Technology
1987-04 *	Claus Lewerentz, Manfred Nagl, Bernhard Westfechtel: On Integration Mechanisms within a Graph-Based Software Development Environment
1987-05 *	Reinhard Rinn: ber Eingabeanomalien bei verschiedenen Inferenzmodellen
1987-06 *	Werner Damm, Gert Dhmen: Specifying Distributed Computer Architectures in AADL*
1987-07 *	Gregor Engels, Claus Lewerentz, Wilhelm Schfer: Graph Grammar Engineering: A Software Specification Method
1987-08 *	Manfred Nagl: Set Theoretic Approaches to Graph Grammars
1987-09 *	Claus Lewerentz, Andreas Schrr: Experiences with a Database System for Software Documents
1987-10 *	Herbert Klaeren, Klaus Indermark: A New Implementation Technique for Recursive Function Definitions
1987-11 *	Rita Loogen: Design of a Parallel Programmable Graph Reduction Machine with Distributed Memory
1987-12	J. Brstler, U. Mncke, R. Wilhelm: Table compression for tree automata
1988-01 *	Gabriele Esser, Johannes Rckert, Frank Wagner Gesellschaftliche Aspekte der Informatik
1988-02 *	Peter Martini, Otto Spaniol: Token-Passing in High-Speed Backbone Networks for Campus-Wide Environments
1988-03 *	Thomas Welzel: Simulation of a Multiple Token Ring Backbone
1988-04 *	Peter Martini: Performance Comparison for HSLAN Media Access Protocols
1988-05 *	Peter Martini: Performance Analysis of Multiple Token Rings
1988-06 *	Andreas Mann, Johannes Rckert, Otto Spaniol: Datenfunknetze
1988-07 *	Andreas Mann, Johannes Rckert: Packet Radio Networks for Data Exchange
1988-08 *	Andreas Mann, Johannes Rckert: Concurrent Slot Assignment Protocol for Packet Radio Networks
1988-09 *	W. Kremer, F. Reichert, J. Rckert, A. Mann: Entwurf einer Netzwerktopologie fr ein Mobilfunknetz zur Untersttzung des ffentlichen Straenverkehrs
1988-10 *	Kai Jakobs: Towards User-Friendly Networking
1988-11 *	Kai Jakobs: The Directory - Evolution of a Standard
1988-12 *	Kai Jakobs: Directory Services in Distributed Systems - A Survey
1988-13 *	Martine Schmmer: RS-511, a Protocol for the Plant Floor

1988-14 * U. Quernheim: Satellite Communication Protocols - A Performance Comparison Considering On-Board Processing

1988-15 * Peter Martini, Otto Spaniol, Thomas Welzel: File Transfer in High Speed Token Ring Networks: Performance Evaluation by Approximate Analysis and Simulation

1988-16 * Fachgruppe Informatik: Jahresbericht 1987

1988-17 * Wolfgang Thomas: Automata on Infinite Objects

1988-18 * Michael Sonnenschein: On Petri Nets and Data Flow Graphs

1988-19 * Heiko Vogler: Functional Distribution of the Contextual Analysis in Block-Structured Programming Languages: A Case Study of Tree Transducers

1988-20 * Thomas Welzel: Einsatz des Simulationswerkzeuges QNAP2 zur Leistungsbewertung von Kommunikationsprotokollen

1988-21 * Th. Janning, C. Lewerentz: Integrated Project Team Management in a Software Development Environment

1988-22 * Joost Engelfriet, Heiko Vogler: Modular Tree Transducers

1988-23 * Wolfgang Thomas: Automata and Quantifier Hierarchies

1988-24 * Uschi Heuter: Generalized Definite Tree Languages

1989-01 * Fachgruppe Informatik: Jahresbericht 1988

1989-02 * G. Esser, J. Rckert, F. Wagner (Hrsg.): Gesellschaftliche Aspekte der Informatik

1989-03 * Heiko Vogler: Bottom-Up Computation of Primitive Recursive Tree Functions

1989-04 * Andy Schrr: Introduction to PROGRESS, an Attribute Graph Grammar Based Specification Language

1989-05 J. Brstler: Reuse and Software Development - Problems, Solutions, and Bibliography (in German)

1989-06 * Kai Jakobs: OSI - An Appropriate Basis for Group Communication?

1989-07 * Kai Jakobs: ISO's Directory Proposal - Evolution, Current Status and Future Problems

1989-08 * Bernhard Westfechtel: Extension of a Graph Storage for Software Documents with Primitives for Undo/Redo and Revision Control

1989-09 * Peter Martini: High Speed Local Area Networks - A Tutorial

1989-10 * P. Davids, Th. Welzel: Performance Analysis of DQDB Based on Simulation

1989-11 * Manfred Nagl (Ed.): Abstracts of Talks presented at the WG '89 15th International Workshop on Graphtheoretic Concepts in Computer Science

1989-12 * Peter Martini: The DQDB Protocol - Is it Playing the Game?

1989-13 * Martine Schmmer: CNC/DNC Communication with MAP

1989-14 * Martine Schmmer: Local Area Networks for Manufactoring Environments with hard Real-Time Requirements

1989-15 * M. Schmmer, Th. Welzel, P. Martini: Integration of Field Bus and MAP Networks - Hierarchical Communication Systems in Production Environments

1989-16 * G. Vossen, K.-U. Witt: SUXESS: Towards a Sound Unification of Extensions of the Relational Data Model

1989-17 * J. Derissen, P. Hruschka, M.v.d. Beeck, Th. Janning, M. Nagl: Integrating Structured Analysis and Information Modelling

1989-18 A. Maassen: Programming with Higher Order Functions

1989-19 * Mario Rodriguez-Artalejo, Heiko Vogler: A Narrowing Machine for Syntax Directed BABEL

1989-20 H. Kuchen, R. Loogen, J.J. Moreno Navarro, M. Rodriguez Artalejo: Graph-based Implementation of a Functional Logic Language

1990-01 * Fachgruppe Informatik: Jahresbericht 1989

1990-02 * Vera Jansen, Andreas Potthoff, Wolfgang Thomas, Udo Wermuth: A Short Guide to the AMORE System (Computing Automata, MOnoids and Regular Expressions)

1990-03 * Jerzy Skurczynski: On Three Hierarchies of Weak SkS Formulas

1990-04 R. Loogen: Stack-based Implementation of Narrowing

1990-05 H. Kuchen, A. Wagener: Comparison of Dynamic Load Balancing Strategies

1990-06 * Kai Jakobs, Frank Reichert: Directory Services for Mobile Communication

1990-07 * Kai Jakobs: What's Beyond the Interface - OSI Networks to Support Cooperative Work

1990-08 * Kai Jakobs: Directory Names and Schema - An Evaluation

1990-09 * Ulrich Quernheim, Dieter Kreuer: Das CCITT - Signalisierungssystem Nr. 7 auf Satellitenstrecken; Simulation der Zeichengabestrecke

1990-11 H. Kuchen, R. Loogen, J.J. Moreno Navarro, M. Rodriguez Artalejo: Lazy Narrowing in a Graph Machine

1990-12 * Kai Jakobs, Josef Kaltwasser, Frank Reichert, Otto Spaniol: Der Computer fhrt mit

1990-13 * Rudolf Mathar, Andreas Mann: Analyzing a Distributed Slot Assignment Protocol by Markov Chains

1990-14 A. Maassen: Compilerentwicklung in Miranda - ein Praktikum in funktionaler Programmierung (written in german)

1990-15 * Manfred Nagl, Andreas Schrr: A Specification Environment for Graph Grammars

1990-16 A. Schrr: PROGRESS: A VHL-Language Based on Graph Grammars

1990-17 * Marita Mller: Ein Ebenenmodell wissensbasierter Konsultationen - Untersttzung fr Wissensakquisition und Erklrungsfhigkeit

1990-18 * Eric Kowalewski: Entwurf und Interpretation einer Sprache zur Beschreibung von Konsultationsphasen in Expertensystemen

1990-20 Y. Ortega Mallen, D. de Frutos Escrig: A Complete Proof System for Timed Observations

1990-21 * Manfred Nagl: Modelling of Software Architectures: Importance, Notions, Experiences

1990-22 H. Fassbender, H. Vogler: A Call-by-need Implementation of Syntax Directed Functional Programming

1991-01 Guenther Geiler (ed.), Fachgruppe Informatik: Jahresbericht 1990

1991-03 B. Steffen, A. Ingolfsdottir: Characteristic Formulae for Processes with Divergence

1991-04 M. Portz: A new class of cryptosystems based on interconnection networks

1991-05 H. Kuchen, G. Geiler: Distributed Applicative Arrays

1991-06 * Ludwig Staiger: Kolmogorov Complexity and Hausdorff Dimension

1991-07 * Ludwig Staiger: Syntactic Congruences for w-languages

1991-09 * Eila Kuikka: A Proposal for a Syntax-Directed Text Processing System

1991-10 K. Gladitz, H. Fassbender, H. Vogler: Compiler-based Implementation of Syntax-Directed Functional Programming

1991-11 R. Loogen, St. Winkler: Dynamic Detection of Determinism in Functional Logic Languages

1991-12 * K. Indermark, M. Rodriguez Artalejo (Eds.): Granada Workshop on the Integration of Functional and Logic Programming

1991-13 * Rolf Hager, Wolfgang Kremer: The Adaptive Priority Scheduler: A More Fair Priority Service Discipline

1991-14 * Andreas Fasbender, Wolfgang Kremer: A New Approximation Algorithm for Tandem Networks with Priority Nodes

1991-15 J. Brstler, A. Zndorf: Revisiting extensions to Modula-2 to support reusability

1991-16 J. Brstler, Th. Janning: Bridging the gap between Requirements Analysis and Design

1991-17 A. Zndorf, A. Schrr: Nondeterministic Control Structures for Graph Rewriting Systems

1991-18 * Matthias Jarke, John Mylopoulos, Joachim W. Schmidt, Yannis Vassiliou: DAIDA: An Environment for Evolving Information Systems

1991-19 M. Jeusfeld, M. Jarke: From Relational to Object-Oriented Integrity Simplification

1991-20 G. Hogen, A. Kindler, R. Loogen: Automatic Parallelization of Lazy Functional Programs

1991-21 * Prof. Dr. rer. nat. Otto Spaniol: ODP (Open Distributed Processing): Yet another Viewpoint

1991-22 H. Kuchen, F. Lcking, H. Stoltze: The Topology Description Language TDL

1991-23 S. Graf, B. Steffen: Compositional Minimization of Finite State Systems

1991-24 R. Cleaveland, J. Parrow, B. Steffen: The Concurrency Workbench: A Semantics Based Tool for the Verification of Concurrent Systems

1991-25 * Rudolf Mathar, Jrgen Mattfeldt: Optimal Transmission Ranges for Mobile Communication in Linear Multihop Packet Radio Networks

1991-26 M. Jeusfeld, M. Staudt: Query Optimization in Deductive Object Bases

1991-27 J. Knoop, B. Steffen: The Interprocedural Coincidence Theorem

1991-28 J. Knoop, B. Steffen: Unifying Strength Reduction and Semantic Code Motion

1991-30 T. Margaria: First-Order theories for the verification of complex FSMs

1991-31 B. Steffen: Generating Data Flow Analysis Algorithms from Modal Specifications

1992-01 Stefan Eherer (ed.), Fachgruppe Informatik: Jahresbericht 1991

1992-02 * Bernhard Westfechtel: Basismechanismen zur Datenverwaltung in strukturbezogenen Hypertextsystemen

1992-04 S. A. Smolka, B. Steffen: Priority as Extremal Probability

1992-05 * Matthias Jarke, Carlos Maltzahn, Thomas Rose: Sharing Processes: Team Coordination in Design Repositories

1992-06 O. Burkart, B. Steffen: Model Checking for Context-Free Processes

1992-07 * Matthias Jarke, Klaus Pohl: Information Systems Quality and Quality Information Systems

1992-08 * Rudolf Mathar, Jrgen Mattfeldt: Analyzing Routing Strategy NFP in Multihop Packet Radio Networks on a Line

1992-09 * Alfons Kemper, Guido Moerkotte: Grundlagen objektorientierter Datenbanksysteme

1992-10 Matthias Jarke, Manfred Jeusfeld, Andreas Miethsam, Michael Gocek: Towards a logic-based reconstruction of software configuration management

1992-11 Werner Hans: A Complete Indexing Scheme for WAM-based Abstract Machines

1992-12 W. Hans, R. Loogen, St. Winkler: On the Interaction of Lazy Evaluation and Backtracking

1992-13 * Matthias Jarke, Thomas Rose: Specification Management with CAD

1992-14 Th. Noll, H. Vogler: Top-down Parsing with Simultaneous Evaluation on Noncircular Attribute Grammars

1992-15 A. Schuerr, B. Westfechtel: Graphgrammatiken und Graphersetzungssysteme(written in german)

1992-16 * Graduiertenkolleg Informatik und Technik (Hrsg.): Forschungsprojekte des Graduiertenkollegs Informatik und Technik

1992-17 M. Jarke (ed.): ConceptBase V3.1 User Manual

1992-18 * Clarence A. Ellis, Matthias Jarke (Eds.): Distributed Cooperation in Integrated Information Systems - Proceedings of the Third International Workshop on Intelligent and Cooperative Information Systems

1992-19-00 H. Kuchen, R. Loogen (eds.): Proceedings of the 4th Int. Workshop on the Parallel Implementation of Functional Languages

1992-19-01 G. Hogen, R. Loogen: PASTEL - A Parallel Stack-Based Implementation of Eager Functional Programs with Lazy Data Structures (Extended Abstract)

1992-19-02 H. Kuchen, K. Gladitz: Implementing Bags on a Shared Memory MIMD-Machine

1992-19-03 C. Rathsack, S.B. Scholz: LISA - A Lazy Interpreter for a Full-Fledged Lambda-Calculus

1992-19-04 T.A. Bratvold: Determining Useful Parallelism in Higher Order Functions

1992-19-05 S. Kahrs: Polymorphic Type Checking by Interpretation of Code

1992-19-06 M. Chakravarty, M. Khler: Equational Constraints, Residuation, and the Parallel JUMP-Machine

1992-19-07 J. Seward: Polymorphic Strictness Analysis using Frontiers (Draft Version)

1992-19-08 D. Grtner, A. Kimms, W. Kluge: pi-Red^+ - A Compiling Graph-Reduction System for a Full Fledged Lambda-Calculus

1992-19-09 D. Howe, G. Burn: Experiments with strict STG code

1992-19-10 J. Glauert: Parallel Implementation of Functional Languages Using Small Processes

1992-19-11 M. Joy, T. Axford: A Parallel Graph Reduction Machine

1992-19-12 A. Bennett, P. Kelly: Simulation of Multicache Parallel Reduction

1992-19-13 K. Langendoen, D.J. Agterkamp: Cache Behaviour of Lazy Functional Programs (Working Paper)

1992-19-14 K. Hammond, S. Peyton Jones: Profiling scheduling strategies on the GRIP parallel reducer

1992-19-15 S. Mintchev: Using Strictness Information in the STG-machine

1992-19-16 D. Rushall: An Attribute Grammar Evaluator in Haskell

1992-19-17 J. Wild, H. Glaser, P. Hartel: Statistics on storage management in a lazy functional language implementation

1992-19-18 W.S. Martins: Parallel Implementations of Functional Languages

1992-19-19 D. Lester: Distributed Garbage Collection of Cyclic Structures (Draft version)

1992-19-20 J.C. Glas, R.F.H. Hofman, W.G. Vree: Parallelization of Branch-and-Bound Algorithms in a Functional Programming Environment

1992-19-21 S. Hwang, D. Rushall: The nu-STG machine: a parallelized Spineless Tagless Graph Reduction Machine in a distributed memory architecture (Draft version)

1992-19-22 G. Burn, D. Le Metayer: Cps-Translation and the Correctness of Optimising Compilers

1992-19-23 S.L. Peyton Jones, P. Wadler: Imperative functional programming (Brief summary)

1992-19-24 W. Damm, F. Liu, Th. Peikenkamp: Evaluation and Parallelization of Functions in Functional + Logic Languages (abstract)

1992-19-25 M. Kesseler: Communication Issues Regarding Parallel Functional Graph Rewriting

1992-19-26 Th. Peikenkamp: Charakterizing and representing neededness in functional loginc languages (abstract)

1992-19-27 H. Doerr: Monitoring with Graph-Grammars as formal operational Models

1992-19-28 J. van Groningen: Some implementation aspects of Concurrent Clean on distributed memory architectures

1992-19-29 G. Ostheimer: Load Bounding for Implicit Parallelism (abstract)

1992-20 H. Kuchen, F.J. Lopez Fraguas, J.J. Moreno Navarro, M. Rodriguez Artalejo: Implementing Disequality in a Lazy Functional Logic Language

1992-21 H. Kuchen, F.J. Lopez Fraguas: Result Directed Computing in a Functional Logic Language

1992-22 H. Kuchen, J.J. Moreno Navarro, M.V. Hermenegildo: Independent AND-Parallel Narrowing

1992-23 T. Margaria, B. Steffen: Distinguishing Formulas for Free

1992-24 K. Pohl: The Three Dimensions of Requirements Engineering

1992-25 * R. Stainov: A Dynamic Configuration Facility for Multimedia Communications

1992-26 * Michael von der Beeck: Integration of Structured Analysis and Timed Statecharts for Real-Time and Concurrency Specification

1992-27 W. Hans, St. Winkler: Aliasing and Groundness Analysis of Logic Programs through Abstract Interpretation and its Safety

1992-28 * Gerhard Steinke, Matthias Jarke: Support for Security Modeling in Information Systems Design

1992-29 B. Schinzel: Warum Frauenforschung in Naturwissenschaft und Technik

1992-30 A. Kemper, G. Moerkotte, K. Peithner: Object-Orientation Axiomatised by Dynamic Logic

1992-32 * Bernd Heinrichs, Kai Jakobs: Timer Handling in High-Performance Transport Systems

1992-33 * B. Heinrichs, K. Jakobs, K. Lenen, W. Reinhardt, A. Spinner: Euro-Bridge: Communication Services for Multimedia Applications

1992-34 C. Gerlhof, A. Kemper, Ch. Kilger, G. Moerkotte: Partition-Based Clustering in Object Bases: From Theory to Practice

1992-35 J. Brstler: Feature-Oriented Classification and Reuse in IPSEN

1992-36 M. Jarke, J. Bubenko, C. Rolland, A. Sutcliffe, Y. Vassiliou: Theories Underlying Requirements Engineering: An Overview of NATURE at Genesis

1992-37 * K. Pohl, M. Jarke: Quality Information Systems: Repository Support for Evolving Process Models

1992-38 A. Zuendorf: Implementation of the imperative / rule based language PROGRES

1992-39 P. Koch: Intelligentes Backtracking bei der Auswertung funktional-logischer Programme

1992-40 * Rudolf Mathar, Jrgen Mattfeldt: Channel Assignment in Cellular Radio Networks

1992-41 * Gerhard Friedrich, Wolfgang Neidl: Constructive Utility in Model-Based Diagnosis Repair Systems

1992-42 * P. S. Chen, R. Hennicker, M. Jarke: On the Retrieval of Reusable Software Components

1992-43 W. Hans, St.Winkler: Abstract Interpretation of Functional Logic Languages

1992-44 N. Kiesel, A. Schuerr, B. Westfechtel: Design and Evaluation of GRAS, a Graph-Oriented Database System for Engineering Applications

1993-01 * Fachgruppe Informatik: Jahresbericht 1992

1993-02 * Patrick Shicheng Chen: On Inference Rules of Logic-Based Information Retrieval Systems

1993-03 G. Hogen, R. Loogen: A New Stack Technique for the Management of Runtime Structures in Distributed Environments

1993-05 A. Zndorf: A Heuristic for the Subgraph Isomorphism Problem in Executing PROGRES

1993-06 A. Kemper, D. Kossmann: Adaptable Pointer Swizzling Strategies in Object Bases: Design, Realization, and Quantitative Analysis

1993-07 * Graduiertenkolleg Informatik und Technik (Hrsg.): Graduiertenkolleg Informatik und Technik

1993-08 * Matthias Berger: k-Coloring Vertices using a Neural Network with Convergence to Valid Solutions

1993-09 M. Buchheit, M. Jeusfeld, W. Nutt, M. Staudt: Subsumption between Queries to Object-Oriented Databases

1993-10 O. Burkart, B. Steffen: Pushdown Processes: Parallel Composition and Model Checking

1993-11 * R. Groe-Wienker, O. Hermanns, D. Menzenbach, A. Pollacks, S. Repetzki, J. Schwartz, K. Sonnenschein, B. Westfechtel: Das SUKITS-Projekt: A-posteriori-Integration heterogener CIM-Anwendungssysteme

1993-12 * Rudolf Mathar, Jrgen Mattfeldt: On the Distribution of Cumulated Interference Power in Rayleigh Fading Channels
1993-13 O. Maler, L. Staiger: On Syntactic Congruences for omega-languages
1993-14 M. Jarke, St. Eherer, R. Gallersdoerfer, M. Jeusfeld, M. Staudt: ConceptBase - A Deductive Object Base Manager
1993-15 M. Staudt, H.W. Nissen, M.A. Jeusfeld: Query by Class, Rule and Concept
1993-16 * M. Jarke, K. Pohl, St. Jacobs et al.: Requirements Engineering: An Integrated View of Representation Process and Domain
1993-17 * M. Jarke, K. Pohl: Establishing Vision in Context: Towards a Model of Requirements Processes
1993-18 W. Hans, H. Kuchen, St. Winkler: Full Indexing for Lazy Narrowing
1993-19 W. Hans, J.J. Ruz, F. Saenz, St. Winkler: A VHDL Specification of a Shared Memory Parallel Machine for Babel
1993-20 * K. Finke, M. Jarke, P. Szczurko, R. Soltysiak: Quality Management for Expert Systems in Process Control
1993-21 M. Jarke, M.A. Jeusfeld, P. Szczurko: Three Aspects of Intelligent Cooperation in the Quality Cycle
1994-01 Margit Generet, Sven Martin (eds.), Fachgruppe Informatik: Jahresbericht 1993
1994-02 M. Lefering: Development of Incremental Integration Tools Using Formal Specifications
1994-03 * P. Constantopoulos, M. Jarke, J. Mylopoulos, Y. Vassiliou: The Software Information Base: A Server for Reuse
1994-04 * Rolf Hager, Rudolf Mathar, Jrgen Mattfeldt: Intelligent Cruise Control and Reliable Communication of Mobile Stations
1994-05 * Rolf Hager, Peter Hermesmann, Michael Portz: Feasibility of Authentication Procedures within Advanced Transport Telematics
1994-06 * Claudia Popien, Bernd Meyer, Axel Kuepper: A Formal Approach to Service Import in ODP Trader Federations
1994-07 P. Peters, P. Szczurko: Integrating Models of Quality Management Methods by an Object-Oriented Repository
1994-08 * Manfred Nagl, Bernhard Westfechtel: A Universal Component for the Administration in Distributed and Integrated Development Environments
1994-09 * Patrick Horster, Holger Petersen: Signatur- und Authentifikationsverfahren auf der Basis des diskreten Logarithmusproblems
1994-11 A. Schrr: PROGRES, A Visual Language and Environment for PROgramming with Graph REwrite Systems
1994-12 A. Schrr: Specification of Graph Translators with Triple Graph Grammars
1994-13 A. Schrr: Logic Based Programmed Structure Rewriting Systems
1994-14 L. Staiger: Codes, Simplifying Words, and Open Set Condition
1994-15 * Bernhard Westfechtel: A Graph-Based System for Managing Configurations of Engineering Design Documents
1994-16 P. Klein: Designing Software with Modula-3
1994-17 I. Litovsky, L. Staiger: Finite acceptance of infinite words

1994-18	G. Hogen, R. Loogen: Parallel Functional Implementations: Graphbased vs. Stackbased Reduction
1994-19	M. Jeusfeld, U. Johnen: An Executable Meta Model for Re-Engineering of Database Schemas
1994-20 *	R. Gallersdrfer, M. Jarke, K. Klabunde: Intelligent Networks as a Data Intensive Application (INDIA)
1994-21	M. Mohnen: Proving the Correctness of the Static Link Technique Using Evolving Algebras
1994-22	H. Fernau, L. Staiger: Valuations and Unambiguity of Languages, with Applications to Fractal Geometry
1994-24 *	M. Jarke, K. Pohl, R. Dmges, St. Jacobs, H. W. Nissen: Requirements Information Management: The NATURE Approach
1994-25 *	M. Jarke, K. Pohl, C. Rolland, J.-R. Schmitt: Experience-Based Method Evaluation and Improvement: A Process Modeling Approach
1994-26 *	St. Jacobs, St. Kethers: Improving Communication and Decision Making within Quality Function Deployment
1994-27 *	M. Jarke, H. W. Nissen, K. Pohl: Tool Integration in Evolving Information Systems Environments
1994-28	O. Burkart, D. Caucal, B. Steffen: An Elementary Bisimulation Decision Procedure for Arbitrary Context-Free Processes
1995-01 *	Fachgruppe Informatik: Jahresbericht 1994
1995-02	Andy Schrr, Andreas J. Winter, Albert Zndorf: Graph Grammar Engineering with PROGRES
1995-03	Ludwig Staiger: A Tight Upper Bound on Kolmogorov Complexity by Hausdorff Dimension and Uniformly Optimal Prediction
1995-04	Birgitta Knig-Ries, Sven Helmer, Guido Moerkotte: An experimental study on the complexity of left-deep join ordering problems for cyclic queries
1995-05	Sophie Cluet, Guido Moerkotte: Efficient Evaluation of Aggregates on Bulk Types
1995-06	Sophie Cluet, Guido Moerkotte: Nested Queries in Object Bases
1995-07	Sophie Cluet, Guido Moerkotte: Query Optimization Techniques Exploiting Class Hierarchies
1995-08	Markus Mohnen: Efficient Compile-Time Garbage Collection for Arbitrary Data Structures
1995-09	Markus Mohnen: Functional Specification of Imperative Programs: An Alternative Point of View of Functional Languages
1995-10	Rainer Gallersdrfer, Matthias Nicola: Improving Performance in Replicated Databases through Relaxed Coherency
1995-11 *	M.Staudt, K.von Thadden: Subsumption Checking in Knowledge Bases
1995-12 *	G.V.Zemanek, H.W.Nissen, H.Hubert, M.Jarke: Requirements Analysis from Multiple Perspectives: Experiences with Conceptual Modeling Technology
1995-13 *	M.Staudt, M.Jarke: Incremental Maintenance of Externally Materialized Views
1995-14 *	P.Peters, P.Szczurko, M.Jeusfeld: Oriented Information Management: Conceptual Models at Work

1995-15 * Matthias Jarke, Sudha Ram (Hrsg.): WITS 95 Proceedings of the 5th Annual Workshop on Information Technologies and Systems

1995-16 * W.Hans, St.Winkler, F.Saenz: Distributed Execution in Functional Logic Programming

1996-01 * Jahresbericht 1995

1996-02 Michael Hanus, Christian Prehofer: Higher-Order Narrowing with Definitional Trees

1996-03 * W.Scheufele, G.Moerkotte: Optimal Ordering of Selections and Joins in Acyclic Queries with Expensive Predicates

1996-04 Klaus Pohl: PRO-ART: Enabling Requirements Pre-Traceability

1996-05 Klaus Pohl: Requirements Engineering: An Overview

1996-06 * M.Jarke, W.Marquardt: Design and Evaluation of Computer–Aided Process Modelling Tools

1996-07 Olaf Chitil: The Sigma-Semantics: A Comprehensive Semantics for Functional Programs

1996-08 * S.Sripada: On Entropy and the Limitations of the Second Law of Thermodynamics

1996-09 Michael Hanus (Ed.): Proceedings of the Poster Session of ALP96 - Fifth International Conference on Algebraic and Logic Programming

1996-09-0 Michael Hanus (Ed.): Proceedings of the Poster Session of ALP 96 - Fifth International Conference on Algebraic and Logic Programming: Introduction and table of contents

1996-09-1 Ilies Alouini: An Implementation of Conditional Concurrent Rewriting on Distributed Memory Machines

1996-09-2 Olivier Danvy, Karoline Malmkjr: On the Idempotence of the CPS Transformation

1996-09-3 Vctor M. Gulas, Jos L. Freire: Concurrent Programming in Haskell

1996-09-4 Sbastien Limet, Pierre Rty: On Decidability of Unifiability Modulo Rewrite Systems

1996-09-5 Alexandre Tessier: Declarative Debugging in Constraint Logic Programming

1996-10 Reidar Conradi, Bernhard Westfechtel: Version Models for Software Configuration Management

1996-11 * C.Weise, D.Lenzkes: A Fast Decision Algorithm for Timed Refinement

1996-12 * R.Dmges, K.Pohl, M.Jarke, B.Lohmann, W.Marquardt: PRO-ART/CE* — An Environment for Managing the Evolution of Chemical Process Simulation Models

1996-13 * K.Pohl, R.Klamma, K.Weidenhaupt, R.Dmges, P.Haumer, M.Jarke: A Framework for Process-Integrated Tools

1996-14 * R.Gallersdrfer, K.Klabunde, A.Stolz, M.Emajor: INDIA — Intelligent Networks as a Data Intensive Application, Final Project Report, June 1996

1996-15 * H.Schimpe, M.Staudt: VAREX: An Environment for Validating and Refining Rule Bases

1996-16 * M.Jarke, M.Gebhardt, S.Jacobs, H.Nissen: Conflict Analysis Across Heterogeneous Viewpoints: Formalization and Visualization

1996-17 Manfred A. Jeusfeld, Tung X. Bui: Decision Support Components on the Internet

1996-18 Manfred A. Jeusfeld, Mike Papazoglou: Information Brokering: Design, Search and Transformation

1996-19 * P.Peters, M.Jarke: Simulating the impact of information flows in networked organizations

1996-20 Matthias Jarke, Peter Peters, Manfred A. Jeusfeld: Model-driven planning and design of cooperative information systems

1996-21 * G.de Michelis, E.Dubois, M.Jarke, F.Matthes, J.Mylopoulos, K.Pohl, J.Schmidt, C.Woo, E.Yu: Cooperative information systems: a manifesto

1996-22 * S.Jacobs, M.Gebhardt, S.Kethers, W.Rzasa: Filling HTML forms simultaneously: CoWeb architecture and functionality

1996-23 * M.Gebhardt, S.Jacobs: Conflict Management in Design

1997-01 Michael Hanus, Frank Zartmann (eds.): Jahresbericht 1996

1997-02 Johannes Faassen: Using full parallel Boltzmann Machines for Optimization

1997-03 Andreas Winter, Andy Schrr: Modules and Updatable Graph Views for PROgrammed Graph REwriting Systems

1997-04 Markus Mohnen, Stefan Tobies: Implementing Context Patterns in the Glasgow Haskell Compiler

1997-05 * S.Gruner: Schemakorrespondenzaxiome untersttzen die paargrammatische Spezifikation inkrementeller Integrationswerkzeuge

1997-06 Matthias Nicola, Matthias Jarke: Design and Evaluation of Wireless Health Care Information Systems in Developing Countries

1997-07 Petra Hofstedt: Taskparallele Skelette fr irregulr strukturierte Probleme in deklarativen Sprachen

1997-08 Dorothea Blostein, Andy Schrr: Computing with Graphs and Graph Rewriting

1997-09 Carl-Arndt Krapp, Bernhard Westfechtel: Feedback Handling in Dynamic Task Nets

1997-10 Matthias Nicola, Matthias Jarke: Integrating Replication and Communication in Performance Models of Distributed Databases

1997-11 * R. Klamma, P. Peters, M. Jarke: Workflow Support for Failure Management in Federated Organizations

1997-13 Markus Mohnen: Optimising the Memory Management of Higher-Order Functional Programs

1997-14 Roland Baumann: Client/Server Distribution in a Structure-Oriented Database Management System

1997-15 George Botorog: High-Level Parallel Programming and the Efficient Implementation of Numerical Algorithms

1998-01 * Fachgruppe Informatik: Jahresbericht 1997

1998-02 Stefan Gruner, Manfred Nagel, Andy Schrr: Fine-grained and Structure-Oriented Document Integration Tools are Needed for Development Processes

1998-03 Stefan Gruner: Einige Anmerkungen zur graphgrammatischen Spezifikation von Integrationswerkzeugen nach Westfechtel, Janning, Lefering und Schrr

1998-04 * O. Kubitz: Mobile Robots in Dynamic Environments

1998-05 Martin Leucker, Stephan Tobies: Truth - A Verification Platform for Distributed Systems

1998-06 *	Matthias Oliver Berger: DECT in the Factory of the Future
1998-07	M. Arnold, M. Erdmann, M. Glinz, P. Haumer, R. Knoll, B. Paech, K. Pohl, J. Ryser, R. Studer, K. Weidenhaupt: Survey on the Scenario Use in Twelve Selected Industrial Projects
1998-09 *	Th. Lehmann: Geometrische Ausrichtung medizinischer Bilder am Beispiel intraoraler Radiographien
1998-10 *	M. Nicola, M. Jarke: Performance Modeling of Distributed and Replicated Databases
1998-11 *	Ansgar Schleicher, Bernhard Westfechtel, Dirk Jger: Modeling Dynamic Software Processes in UML
1998-12 *	W. Appelt, M. Jarke: Interoperable Tools for Cooperation Support using the World Wide Web
1998-13	Klaus Indermark: Semantik rekursiver Funktionsdefinitionen mit Striktheitsinformation
1999-01 *	Jahresbericht 1998
1999-02 *	F. Huch: Verifcation of Erlang Programs using Abstract Interpretation and Model Checking — Extended Version
1999-03 *	R. Gallersdrfer, M. Jarke, M. Nicola: The ADR Replication Manager
1999-04	Mara Alpuente, Michael Hanus, Salvador Lucas, Germn Vidal: Specialization of Functional Logic Programs Based on Needed Narrowing
1999-05 *	W. Thomas (Ed.): DLT 99 - Developments in Language Theory Fourth International Conference
1999-06 *	Kai Jakobs, Klaus-Dieter Kleefeld: Informationssysteme fr die angewandte historische Geographie
1999-07	Thomas Wilke: CTL+ is exponentially more succinct than CTL
1999-08	Oliver Matz: Dot-Depth and Monadic Quantifier Alternation over Pictures
2000-01 *	Jahresbericht 1999
2000-02	Jens Vge, Marcin Jurdzinski A Discrete Strategy Improvement Algorithm for Solving Parity Games
2000-03	D. Jger, A. Schleicher, B. Westfechtel: UPGRADE: A Framework for Building Graph-Based Software Engineering Tools
2000-04	Andreas Becks, Stefan Sklorz, Matthias Jarke: Exploring the Semantic Structure of Technical Document Collections: A Cooperative Systems Approach
2000-05	Mareike Schoop: Cooperative Document Management
2000-06	Mareike Schoop, Christoph Quix (eds.): Proceedings of the Fifth International Workshop on the Language-Action Perspective on Communication Modelling
2000-07 *	Markus Mohnen, Pieter Koopman (Eds.): Proceedings of the 12th International Workshop of Functional Languages
2000-08	Thomas Arts, Thomas Noll: Verifying Generic Erlang Client-Server Implementations
2001-01 *	Jahresbericht 2000
2001-02	Benedikt Bollig, Martin Leucker: Deciding LTL over Mazurkiewicz Traces
2001-03	Thierry Cachat: The power of one-letter rational languages

2001-04	Benedikt Bollig, Martin Leucker, Michael Weber: Local Parallel Model Checking for the Alternation Free mu-Calculus
2001-05	Benedikt Bollig, Martin Leucker, Thomas Noll: Regular MSC Languages
2001-06	Achim Blumensath: Prefix-Recognisable Graphs and Monadic Second-Order Logic
2001-07	Martin Grohe, Stefan Whrle: An Existential Locality Theorem
2001-08	Mareike Schoop, James Taylor (eds.): Proceedings of the Sixth International Workshop on the Language-Action Perspective on Communication Modelling
2001-09	Thomas Arts, Jrgen Giesl: A collection of examples for termination of term rewriting using dependency pairs
2001-10	Achim Blumensath: Axiomatising Tree-interpretable Structures
2001-11	Klaus Indermark, Thomas Noll (eds.): Kolloquium Programmiersprachen und Grundlagen der Programmierung
2002-01 *	Jahresbericht 2001
2002-02	Jrgen Giesl, Aart Middeldorp: Transformation Techniques for Context-Sensitive Rewrite Systems
2002-03	Benedikt Bollig, Martin Leucker, Thomas Noll: Generalised Regular MSC Languages
2002-04	Jrgen Giesl, Aart Middeldorp: Innermost Termination of Context-Sensitive Rewriting
2002-05	Horst Lichter, Thomas von der Maen, Thomas Weiler: Modelling Requirements and Architectures for Software Product Lines
2002-06	Henry N. Adorna: 3-Party Message Complexity is Better than 2-Party Ones for Proving Lower Bounds on the Size of Minimal Nondeterministic Finite Automata
2002-07	Jrg Dahmen: Invariant Image Object Recognition using Gaussian Mixture Densities
2002-08	Markus Mohnen: An Open Framework for Data-Flow Analysis in Java
2002-09	Markus Mohnen: Interfaces with Default Implementations in Java
2002-10	Martin Leucker: Logics for Mazurkiewicz traces
2002-11	Jrgen Giesl, Hans Zantema: Liveness in Rewriting
2003-01 *	Jahresbericht 2002
2003-02	Jrgen Giesl, Ren Thiemann: Size-Change Termination for Term Rewriting
2003-03	Jrgen Giesl, Deepak Kapur: Deciding Inductive Validity of Equations
2003-04	Jrgen Giesl, Ren Thiemann, Peter Schneider-Kamp, Stephan Falke: Improving Dependency Pairs
2003-05	Christof Lding, Philipp Rohde: Solving the Sabotage Game is PSPACE-hard
2003-06	Franz Josef Och: Statistical Machine Translation: From Single-Word Models to Alignment Templates
2003-07	Horst Lichter, Thomas von der Maen, Alexander Nyen, Thomas Weiler: Vergleich von Anstzen zur Feature Modellierung bei der Softwareproduktlinienentwicklung
2003-08	Jrgen Giesl, Ren Thiemann, Peter Schneider-Kamp, Stephan Falke: Mechanizing Dependency Pairs
2004-01 *	Fachgruppe Informatik: Jahresbericht 2003

2004-02	Benedikt Bollig, Martin Leucker: Message-Passing Automata are expressively equivalent to EMSO logic
2004-03	Delia Kesner, Femke van Raamsdonk, Joe Wells (eds.): HOR 2004 – 2nd International Workshop on Higher-Order Rewriting
2004-04	Slim Abdennadher, Christophe Ringeissen (eds.): RULE 04 – Fifth International Workshop on Rule-Based Programming
2004-05	Herbert Kuchen (ed.): WFLP 04 – 13th International Workshop on Functional and (Constraint) Logic Programming
2004-06	Sergio Antoy, Yoshihito Toyama (eds.): WRS 04 – 4th International Workshop on Reduction Strategies in Rewriting and Programming
2004-07	Michael Codish, Aart Middeldorp (eds.): WST 04 – 7th International Workshop on Termination
2004-08	Klaus Indermark, Thomas Noll: Algebraic Correctness Proofs for Compiling Recursive Function Definitions with Strictness Information
2004-09	Joachim Kneis, Daniel Mlle, Stefan Richter, Peter Rossmanith: Parameterized Power Domination Complexity
2004-10	Zinaida Benenson, Felix C. Grtner, Dogan Kesdogan: Secure Multi-Party Computation with Security Modules
2005-01 *	Fachgruppe Informatik: Jahresbericht 2004
2005-02	Maximillian Dornseif, Felix C. Grtner, Thorsten Holz, Martin Mink: An Offensive Approach to Teaching Information Security: "Aachen Summer School Applied IT Security"
2005-03	Jrgen Giesl, Ren Thiemann, Peter Schneider-Kamp: Proving and Disproving Termination of Higher-Order Functions
2005-04	Daniel Mlle, Stefan Richter, Peter Rossmanith: A Faster Algorithm for the Steiner Tree Problem
2005-05	Fabien Pouget, Thorsten Holz: A Pointillist Approach for Comparing Honeypots
2005-06	Simon Fischer, Berthold Vcking: Adaptive Routing with Stale Information
2005-07	Felix C. Freiling, Thorsten Holz, Georg Wicherski: Botnet Tracking: Exploring a Root-Cause Methodology to Prevent Distributed Denial-of-Service Attacks
2005-08	Joachim Kneis, Peter Rossmanith: A New Satisfiability Algorithm With Applications To Max-Cut
2005-09	Klaus Kursawe, Felix C. Freiling: Byzantine Fault Tolerance on General Hybrid Adversary Structures
2005-10	Benedikt Bollig: Automata and Logics for Message Sequence Charts
2005-11	Simon Fischer, Berthold Vcking: A Counterexample to the Fully Mixed Nash Equilibrium Conjecture
2005-12	Neeraj Mittal, Felix Freiling, S. Venkatesan, Lucia Draque Penso: Efficient Reductions for Wait-Free Termination Detection in Faulty Distributed Systems
2005-13	Carole Delporte-Gallet, Hugues Fauconnier, Felix C. Freiling: Revisiting Failure Detection and Consensus in Omission Failure Environments
2005-14	Felix C. Freiling, Sukumar Ghosh: Code Stabilization
2005-15	Uwe Naumann: The Complexity of Derivative Computation

2005-16 Uwe Naumann: Syntax-Directed Derivative Code (Part I: Tangent-Linear Code)

2005-17 Uwe Naumann: Syntax-directed Derivative Code (Part II: Intraprocedural Adjoint Code)

2005-18 Thomas von der Maen, Klaus Mller, John MacGregor, Eva Geisberger, Jrg Drr, Frank Houdek, Harbhajan Singh, Holger Wumann, Hans-Veit Bacher, Barbara Paech: Einsatz von Features im Software-Entwicklungsprozess - Abschlubericht des GI-Arbeitskreises "Features"

2005-19 Uwe Naumann, Andre Vehreschild: Tangent-Linear Code by Augmented LL-Parsers

2005-20 Felix C. Freiling, Martin Mink: Bericht ber den Workshop zur Ausbildung im Bereich IT-Sicherheit Hochschulausbildung, berufliche Weiterbildung, Zertifizierung von Ausbildungsangeboten am 11. und 12. August 2005 in Kln organisiert von RWTH Aachen in Kooperation mit BITKOM, BSI, DLR und Gesellschaft fuer Informatik (GI) e.V.

2005-21 Thomas Noll, Stefan Rieger: Optimization of Straight-Line Code Revisited

2005-22 Felix Freiling, Maurice Herlihy, Lucia Draque Penso: Optimal Randomized Fair Exchange with Secret Shared Coins

2005-23 Heiner Ackermann, Alantha Newman, Heiko Rglin, Berthold Vcking: Decision Making Based on Approximate and Smoothed Pareto Curves

2005-24 Alexander Becher, Zinaida Benenson, Maximillian Dornseif: Tampering with Motes: Real-World Physical Attacks on Wireless Sensor Networks

2006-03 Uwe Naumann: Intraprocedural Adjoint Code Generated by the Differentiation-Enabled NAGWare Fortran Compiler

2006-04 Ebadollah Varnik, Uwe Naumann, Andrew Lyons: Toward Low Static Memory Jacobian Accumulution

2006-06 Joachim Kneis, Daniel Mlle, Stefan Richter, Peter Rossmanith: Divide-and-Color

2006-07 Thomas Colcombet, Christof Lding:: Transforming structures by set interpretations

2006-08 Uwe Naumann, Yuxiao Hu: Optimal Vertex Elimination in Single-Expression-Use Graphs

* These reports are only available as a printed version.

Please contact `biblio@informatik.rwth-aachen.de` to obtain copies.