MOBILEMAN

IST-2001-38113

Mobile Metropolitan Ad hoc Networks

MOBILEMAN

Architecture, protocols and services

Deliverable D5

Contractual Report Preparation Date: September 2003
Actual Date of Delivery: 7 October 2004
Estimated Person Months: 46
Number of pages: 165

Contributing Partners: Consiglio Nazionale delle Ricerche (Italy), University of Cambridge (UK), Institut Eurecom (France), Helsinki University (Finland), NETikos (Italy), Scuola Universitaria Professionale della Svizzera Italiana (Switzerland)

Authors: Marco Conti, Giovanni Turi, Gaia Maselli (CNR), Jon Crowcroft, Sven Ostring (Cambridge), Pietro Michiardi, Refik Molva (Eurecom), Jose Costa Requena (HUT), Piergiorgio Cremonese, Veronica Vanni (Netikos), Ivan Defilippis, Silvia Giordano, Alessandro Puiatti (SUPSI)

Abstract: The aim of this deliverable is to provide the overview of the architecture, protocols and services designed for the MobileMAN paradigm as emerging at the end of the first year of the project. First, we present the complete architecture with the communication flows among different functions, and then we discuss protocols belonging to the MobileMAN protocols' stack. Protocols are presented by following a bottom up approach from wireless technologies up to the application and economic issues.

Project funded by the European Community under the "Information Society Technologies" Programme (1998-2002)

CONTENTS LIST

SUMMARY

The aim of this deliverable is to provide the overview of the architecture, protocols, and services designed for the MobileMAN paradigm at the end of the first year of the project. Specifically, the deliverable is organized as follows. First, we present the complete architecture with the communication flows among different functions, then the reminder of the deliverable presents an in depth analysis of protocols belonging to the MobileMAN protocols' stack. Protocols are presented by following a bottom up approach from wireless technologies up to the application and economic issues. In addition, protocols' presentations are grouped to reflect the working groups that are operating inside the MobileMAN project. As explained in Deliverable D2, to better coordinate the project activities, working groups have been defined to address the specific issues of each research area: Wireless Technologies (CNR and SUPSI), Networking and Security (CNR, Eurecom and HUT), Middleware and Applications (Cambridge, CNR, HUT, Netikos).

As far as the architecture, we decided to enhance the reference model of MobileMAN in order to integrate in a careful way, in our architecture, the novel view of cross layering, while maintaining layer independence when opportune. A cross-layering approach is emerging as the most suitable attempt to optimize the architecture and protocols for systems with high dependences among layers, as wireless ones. In MobileMAN, some specific information, gathered at different layers of the network stack, is shared in a common local memory structure, and used to adapt the behavior of the node depending on the particular circumstance (e.g., traffic type, channel perturbations, network status, node selfishness and/or maliciousness, among the others) the node operates in. This is likely to be the first attempt, in the research community, that such type of reference architecture is adopted. However, the MobileMAN reference architecture tries to achieve the advantages of an advanced cross layer design (i.e., joint optimization of protocols belonging to different layers) still satisfying the layer separation principle, i.e., protocols belonging to different layers can be added/removed from the protocol stack without modifying the protocols operating at the other layers. This implies that a high attention will be given to the choice of the shared information, and the balance between cross-layering and layer separation will be matter of careful research.

The reminder of the deliverable presents, for all the layers of the MobileMAN protocol stack, an exhaustive overview of relevant issues and the directions for solving them. In detail, for each section dedicated to a particular layer, a detailed state of the art is presented focusing on the particular requirements needed to satisfy both the ad hoc networking and the MobileMAN objectives. Where suitable, an analysis of existing approaches that solve the issues specific to each layer is presented. Investigating on whether the techniques available in the literature were suitable for the MobileMAN purposes was one of the fundamental directions to follow in the design of the system architecture in order to be able to focus on specific and unsolved issues. Depending on the progress status of each part of the network stack, the design and the implementation details of original components is provided, when available. Moreover, if a final solution or partial results only, were available, a detailed description of each research direction is presented.

This deliverable represents a precious reference point for a collaborative development of the MobileMAN testbed, a source of information for a cross layering design in which the issues, objectives, and solutions proposed for each specific layer are available and exploitable by protocols belonging to different levels.

1. MOBILEMAN ARCHITECTURE

One of the major challenges in the research on mobile ad hoc networks is to have them fully functional with good performance while, at the same time, make them able to communicate with the rest of the Internet.

The IETF MANET WG proposes a view of mobile ad hoc networks as an evolution of the Internet. This mainly implies an IP-centric view of the network, and the use of a layered architecture. This paradigm has greatly simplified network design and led to the robust scalable protocols in the Internet. The use of the IP protocol has two main advantages: it simplifies MANET interconnection to the Internet, and guarantees the independence from wireless technologies [MC03]. However, current results show that the layered approach is not equally valid in terms of performance [GW02]. The layered approach leads the research efforts mainly to target isolated components of the overall network design (e.g., routing, MAC, power control). Each layer in the protocol stack is designed and operated independently, with interfaces between layers that are static and independent of the individual network constraints and applications. However, as shown in Figure 1.1, in a MANET some functions cannot be assigned to a single layer. Energy management, security and cooperation, quality of service, among the others cannot be completely implemented in a single layer but they are implemented by combining and exploiting mechanisms implemented in all layers. An efficient implementation of these functions can thus be achieved by avoiding a strict layering approach in which the protocols at each layer are developed in isolation, but rather within an integrated and hierarchical framework to take advantage of the interdependencies between them. For example, from the energy management standpoint, power control and multiple antennas at the link layer are coupled with scheduling at MAC layer, and with energy-constrained and delay-constrained routing at network layer.

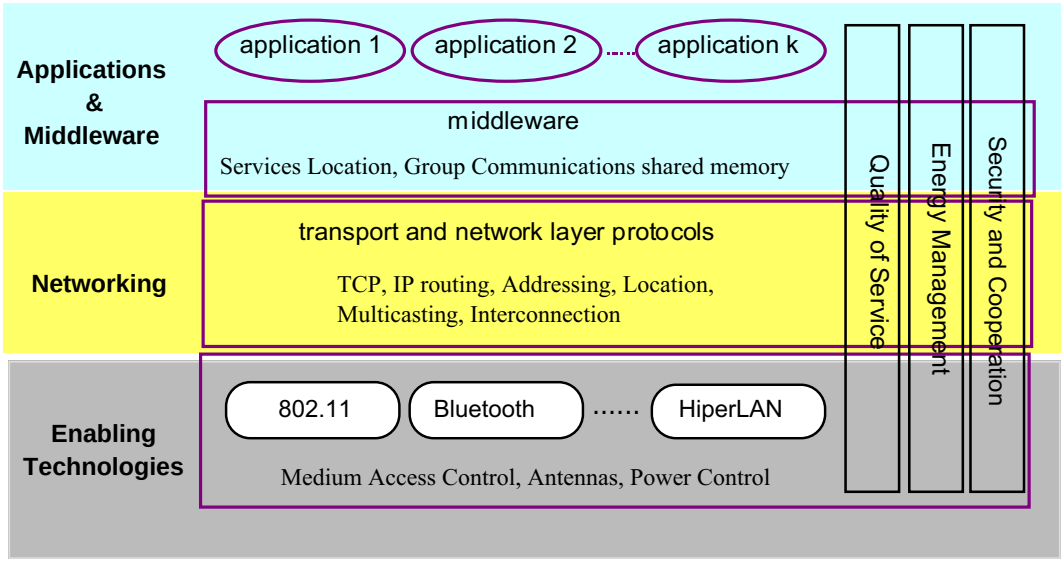


Figure 1.1: MANET Layered Architecture

Relaxing the Internet layered architecture, by removing strict layer boundaries, is therefore an open issue in the mobile ad hoc networks evolution. However, the layered approach was, and is, one of the key elements of the world-wide diffusion of the Internet. The question is to what extent the pure layered approach needs to be modified.

At one end, we have solutions based on *layer triggers* that are still compatible with the principle of separation among layers. A full cross-layering-design represents the other extreme.

Layer triggers are pre-defined signals to notify some events to the higher layers, e.g., failure in data delivery, thus increasing the cooperation among layers. Layer triggers have been extensively used both in the wired and wireless Internet. For example, in wired Internet, the Explicit Congestion Notification (ECN) mechanism is used by intermediate routers to notify congestion conditions to the TCP layer, while in [C00] it was proposed to add L2 triggers, between the link and IP layer, to efficiently detect (at IP layer) changes in the wireless links' status.

A full cross-layer design is a more extreme solution that tries to exploit, in the protocols design, layers' interdependencies to optimize the overall network performance. In this case, control information is continuously flowing top down and bottom up through the protocols' stack and a protocol behavior adapts both to higher and lower protocols' status. For example, the physical layer can adapt rate, power, and coding to meet the requirements of the application given current channel and network conditions; the MAC layer can adapt based on underlying link and interference conditions as well as delay constraints and bit priorities. Adaptive routing protocols can be developed based on current link, network, and traffic conditions. Finally, the application layer can utilize a notion of soft QoS that adapts to the underlying network conditions to deliver the highest possible application quality [GW02].

In MANET, the use of layer triggers has been extensively proposed for fixing the problems due to TCP - IP - MAC interactions [CRVP01] [HV02]. For example, to minimize the impact of mobility and link disconnection on TCP performance, it was proposed to introduce explicit signaling (Route Failure and Route Re-establishment notifications) from intermediate nodes to notify the sender TCP of the disruption of the current route, and construction of a new one [CRVP01]. However, recent works indicate that layer triggers are not enough for fixing MANET performance problems, and a cross layer design must be adopted, see for example [FZX03]. In this paper, to fix the problems due to TCP - IP - MAC interactions, in addition to the use of layer triggers, it has been proposed a "cross layer" design of protocols' mechanisms. Specifically, two link level mechanisms, Link RED[1] and adaptive spacing[2], are introduced to improve TCP efficiency, thus implying a joint design of the MAC and TCP protocols. Several other examples have been presented and discussed in the literature showing the advantages of a cross-layer design at different layers of the protocol stack: from the MAC and physical layers [YLA02], to middleware and routing layers [CZN02]. Existing works point out the advantages of the cross-layer design focusing only on a specific problem (e.g., data accessibility [CZN02]), and looking at the joint design of 2-3 layers only, e.g., physical, MAC and routing.

Currently, a debate is ongoing among ad-hoc-network researchers on cross layered vs. legacy layered architectures. While it is well recognized that cross layering can provide significant performance benefits, it is also pointed out that a layered design has been one of the key element of the success and proliferation of Internet [KK03]. Supporters of the layered architectures point out:

i. this design approach guarantees controlled interactions among layers, and hence designers of protocols at a particular layer do not need to worry about the rest of the stack. On the other hand, a cross layer design can produce unintended interactions among layers (e.g., adaptation loops) resulting in performance degradation;

[1] Similarly to ECN, it provides TCP with an early sign of overload at link level.

[2] It extends the backoff interval of the station that has successfully transmitted, thus reducing the risk of stations' starvation.

ii. an "unbridled" cross layer design can produce a spaghetti-like code that is impossible to maintain in an efficient way as every modification needs to be propagated to the all protocols.

We believe that layer triggers are not the only solution for overcoming all MANET performance problems, and that a "careful" cross layer design must be adopted. Our approach is to introduce inside the layered architecture the possibility of protocols belonging to different layers to cooperate by sharing network-status information still maintaining layers' separation for protocols design.

This is a very innovative approach for a working solution. At the best of our knowledge, no reference architecture has been defined to exploit a full[3] cross-layer design of a MANET protocol stack. For this reason, in the framework of the MOBILEMAN project we have defined reference architecture for MANET able to exploit the advantages of a balanced cross-layer design.

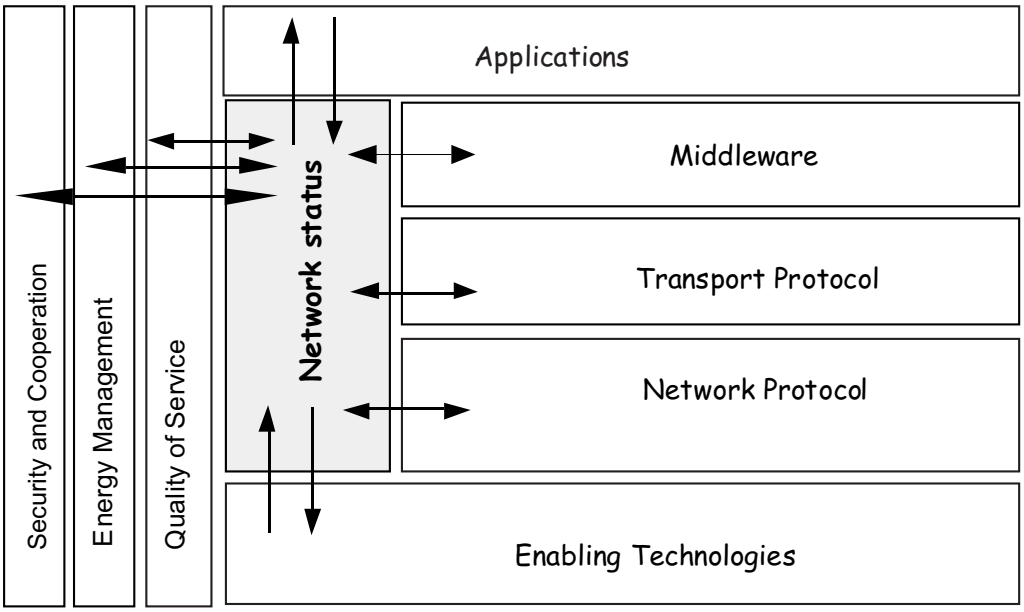


Figure 1.2: MobileMAN reference architecture

Figure 1.2 shows the MOBILEMAN cross-layer reference architecture. In this architecture, cross layering is limited to parameters and implemented through data sharing. As shown in the figure, the innovation of the architecture is a shared memory, "Network Status" in the figure that is a repository of all the network status information collected by the network protocols. All protocols can access this memory to write the information to share with the other protocols, and to read information produced/collected from the other protocols. This avoids duplicating the layers' efforts for collecting network-status information, thus leading to a more efficient system design. In addition, inter-layer co-operations can be easily implemented by variables sharing. However, protocols are still implemented inside each layer, as in the traditional layered reference architecture. This has several advantages:

- Allows for a full compatibility with standards, as it does not touch the core functions of each layer.

[3] With the term "full" we refer to the fact that the cross layered design applies to all layers.

- Is robust to upgrading, and protocols belonging to different layers can be added/removed from the protocol stack without modifying the operations at the other layers.
- It maintains all the advantages of a modular architecture.

To summarize the MOBILEMAN reference architecture tries to achieve the advantages of a full cross layer design (i.e., joint optimization of protocols belonging to different layers) still satisfying the layer separation principle.

Layer separation is achieved by standardizing the access to the Network Status. This mainly implies defining the way protocols can read and write the data from it. Interactions between protocols and the Network Status are placed beside (juxtaposed) to normal layers behavior, and provide optimization without compromising the expected normal functioning. Replacing a Network Status oriented protocol with its legacy counterpart will therefore allow the whole stack to keep working properly. For example, using the legacy TCP protocol as the transport protocol of the MOBILEMAN architecture implies that cross-layer optimizations will not occur at this layer. In addition, in this case the transport protocol will not provide any information to the Network Status but, even though in a degraded way, from the performance standpoint, the overall protocols stack will still correctly operate.

Several approaches can be used to implement the Network Status and to standardize the interactions between the protocols and the Network Status. A promising approach is based on its implementation through a shared memory, where data is organized as a tuple space [MCE02]. Operations defined on tuple spaces can be used to model the interactions between protocols and the Network Status. Using the tuple-space paradigm for implementing the Network Status opens also interesting possibilities such as Network Status sharing among nodes that are one-hop away (see the concept of transient tuple space in Lime [MPR01]).

We believe that the MobileMAN reference architecture guarantees significant performance advantages in the ad hoc network design:

Cross Layer optimization for all network functions. Cross layering is a must for functions such as energy management, but provides benefits for all network functions. As explained below, MobileMAN intends to study the advantages of cross layering in the design of all network functions.

Local and global adaptation can be performed to adapt the system to highly variable ad hoc network conditions, and to better control the system performance. For example, by exploiting a cross-layering design, both local and global adaptation to network congestion can be performed. Specifically, the MAC locally reacts to congestion by exponential backoff; when congestion is high this is not enough, and a global compensation may be performed: i) the forwarding mechanism may re-route the traffic to avoid the bottleneck, or ii) if an alternative routes do not exist, the transport protocol mechanisms can be used to freeze the traffic sending.

Full Context Awareness at all layers. At each layer, protocols can be designed to be aware of the network status, energy level, etc. Cross layering makes easy to achieve context awareness also at higher layers such as middleware and application layers.

Reduced overhead for collecting the network status information, avoiding data duplication at different layers.

These benefits can be achieved with two different approaches: i) Protocols re-design; and ii) Protocols adaptation. We are following both approaches at different layers. The MobileMAN testbed will be used to test and experience both directions. In order to fully exploit cross layering, and to measure its impact on ad hoc networks performance, novel solutions will be developed for

the enabling technologies, network and transport layer, and for the middleware layer. Solutions for power management, security and cooperation will cross all these layers, see Figure 1.3.

Enhanced Wi-Fi. 802.11 is the reference technology for the MobileMAN project. To fix 802.11 problems in ad hoc configurations and to enhance its performance, in MobileMAN we will implement an enhanced IEEE 802.11 card. As explained in Section 2, the enhanced Wi-Fi will exploit cross-layer coordination with higher layers (mainly the Network and Transport).

Network Layer. The main functions (Routing, forwarding, and nodes' location) will be analyzed and possibly re-designed with the aim to exploit cross layering.[4] The design of Network-layer functions will be performed jointly with cooperation and *performability* functions.

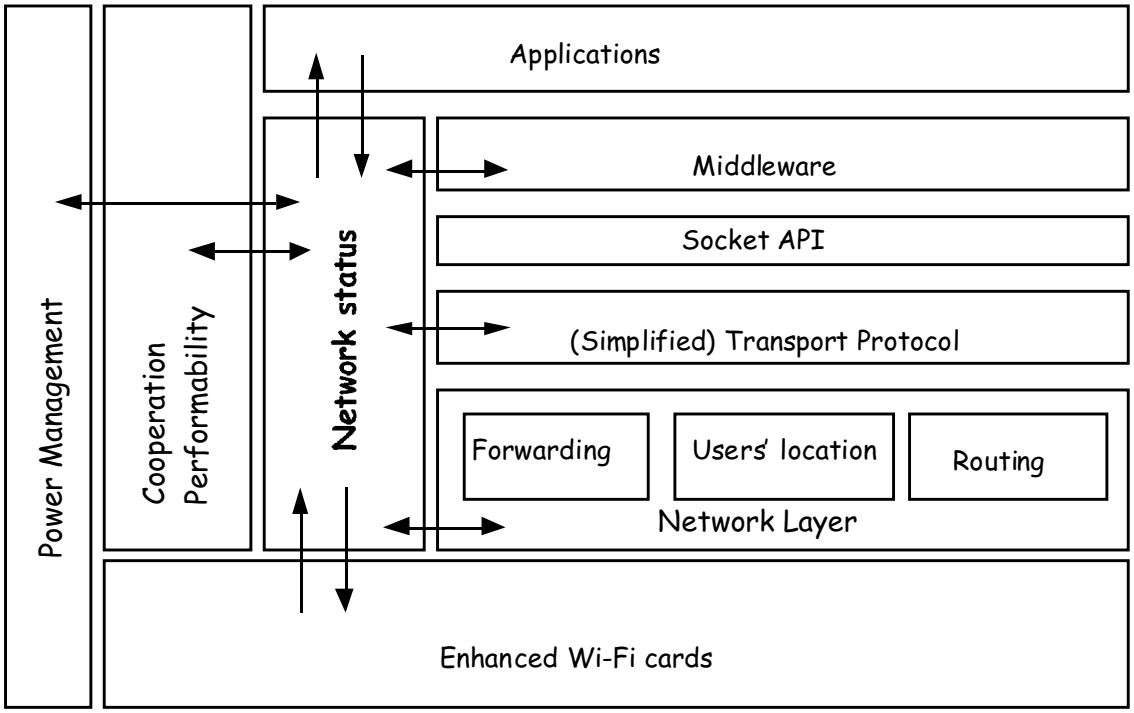


Figure 1.3: MobileMAN protocol stack

Transport Protocol. The main goal in the design of a Transport Protocol is to provide to the upper layers the TCP type of service, i.e., a reliable and connection-oriented service that minimizes useless data re-transmissions by analyzing and reacting appropriately to the different phenomena happening at the below layers (e.g., route failures, route changes, congestions). The efficient implementation of a reliable transport protocol in an ad hoc network requires a strict cooperation with lower layers [CCL03]. Therefore, the MobileMAN Transport Protocol will be designed to exploit information reported by the routing and Wi-Fi layers in the Network Status.

Middleware. The middleware layer generally provides context abstractions able to hide complex details to application programmers. In mobile (ad hoc) environments, this trend has to be reversed to context-awareness [MCE02]. The MobileMAN cross-layered architecture aims at supporting this aspect. At middleware layer, cross layering makes possible a full context awareness of the

[4] Note that this is fully in line with the MANET vision of a MANET as an autonomous system interconnected with the Internet, thus using an Interior Gateway Routing.

underlying network context, and hence to achieve better performance, see e.g., the design of a peer-to-peer system on top of ad hoc networks [GM01]. Specifically, we are currently investigating p2p information delivery based on efficient implementations of Pastry mechanisms in ad hoc networks [CP02]. In MANET, by exploiting cross layer information, the middleware layer can directly use the topology information collected by the routing protocol avoiding, at the p2p layer, the construction of an overlay network unaware of the topology of the underlying ad hoc network. In addition, our cross layer approach makes possible introducing location information and scope information so that content is initially placed and requests are routed to copies that have proximity on a number of QoS axes.

1.1. Interconnection to the Internet

The legacy layered architecture provides a straightforward solution to the interconnection of ad hoc networks to the Internet. As opposite, the cross-layering approach is not a direct solution and opens several research issues. Before discussing these issues it is useful to present our view of the relationship between Internet and MANET. As shown in Figure 1.4, an ad hoc network can either operate in isolation (virtual community network) but can also be interconnected to the Internet through one (or more) Internet access router, i.e., a node that has both a wireless interface to participate to the ad hoc network and a wired interface that connects it to the Internet.

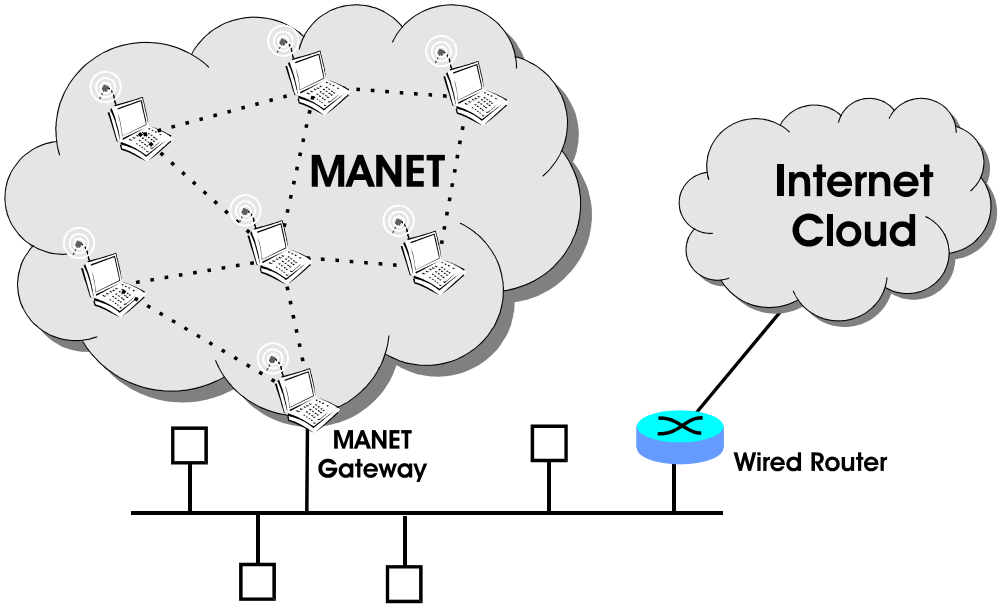


Figure 1.4: MANET as Internet Extension

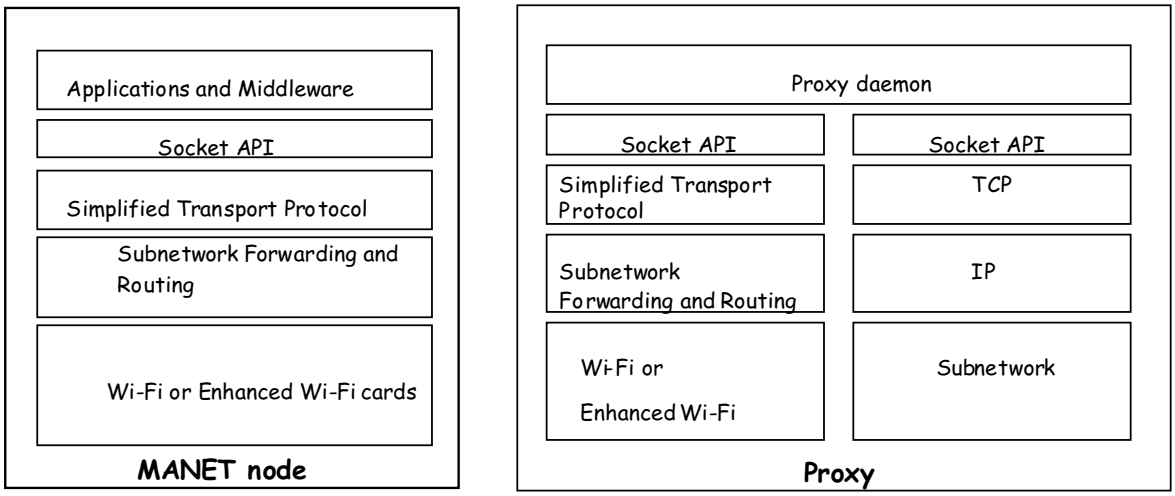


Figure 1.5: Interconnection to Internet: Proxy-based Architecture

In the former case using protocols specifically designed for MANET does not create any problem, while in the latter case solutions must be devised to compensate the differences between networking protocols in MANET and Internet. The easiest way, as shown in Figure 1.5 is to implement a proxy function on the Internet access router that compensates for the protocol differences. This is an effective solution that has already been proposed when wireless LANs are used to access the Internet, see the Indirect TCP model [BB97].

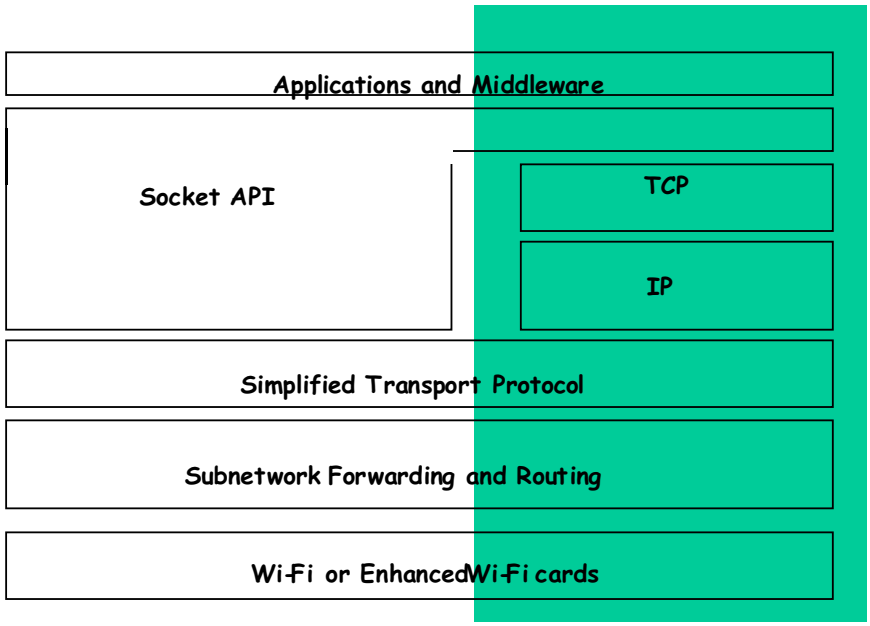


Figure 1.6: Interconnection to Internet: Ad hoc used as Subnetwork

In the case in which a proxy is not available, the ad hoc network can be seen from the Internet standpoint as a subnetwork technology (similarly to ATM, X.25, etc.) on top of which legacy TPC/IP protocols can be implemented. This is shown in Figure 1.6. Communications among MANET nodes are directly performed by exploiting the ad hoc network protocols, only. When, a node in the MANET needs to communicate with a remote Internet host, the ad-hoc-network

protocols are used to establish a "point-to-point" link between the ad hoc node and the Internet access router; on top of this link run the legacy TCP/IP protocols.

1.2. Design Methods and Tools

In the next sections of this deliverable, we will present and evaluate the main directions taken in the MobileMAN project for re-designing the protocol stack to exploit the cross layering potentialities. As explained before, legacy protocol can still be used inside the MobileMAN architecture, for this reason we will also analyze the performance of legacy solutions emerging in the framework of the IEFT MANET WG.

Performance studies will be used to analyze and compare alternative solutions. There are two main approaches in system performance evaluation: the first uses measurements; the second is based on a representation of the system behavior via a model [L83] [KM88]. Measurement techniques are applied to real systems, and thus they can be applied only when a real system, or a prototype of it, is available. Currently, only few measurements studies on real ad hoc testbeds can be found in the literature, see e.g., [BMJ00] [APE02].

Constructing a real ad hoc network test-bed for a given scenario is typically expensive and remains limited in terms of working scenarios, mobility models, etc. Furthermore, measurements are generally non-repeatable. For these reasons, protocols scalability, sensitiveness to users' mobility patterns and speeds are difficult to investigate on a real testbed. Using a simulation or analytic model, on the other hand, permits the study of system behavior by varying all its parameters, and considering a large spectrum of network scenarios.

Evaluating system performance via a model consists of two steps: i) defining the system model, and ii) solving the model using analytical and/or simulative techniques. Analytical methods are often not detailed enough for the ad hoc networks evaluation and in terms of accounting for mobility, in their infancy. On the other hand, simulation modeling is a more standardized, mature, and flexible tool for modeling various protocols and network scenarios, and allows (by running the simulation model) data collection and analyses that fully characterize the protocol performance in most cases.

A very large number of simulation models have been developed to study ad hoc network architectures and protocols under many network scenarios (number of nodes, mobility rates, etc.). Simulation studies have been extensively applied for instance to compare and contrast large number of routing protocols developed for MANETs, see e.g., [JLHM99] [DCY00] [DPR00] [BMJHJ98]. [FS03] presents a theoretical framework to compare ad hoc-network routing protocols (in an implementation independent manner) by measuring each protocol's performance relative to a theoretical optimum.

The use of simulation techniques in the performance evaluation of communication networks is a consolidated research area (see [CD99] and the references herein); however MANET simulation has several open research issues. An in depth discussion of methods and techniques for MANETs simulation can be found in [BB03c] [CCL03]; hereafter it is important to point one of the major problems in using simulation in MANET studies: the reliability of simulative results. Most MANET simulative studies are based on simulation tools. The main advantage of these tools is that they provide libraries containing pre-defined models for most communication protocols (e.g. 802.11, Ethernet, TCP, etc.). In addition, these tools often provide graphical interfaces that can be used both during the model development phase, and during simulation runs to simplify following dynamic protocol and network behaviors. Popular network simulators used in ad hoc networks include: OPNET [OPN], NS-2 [NS2], Glomosim [GLOM] and its commercial version QualNet [QUAL]. They all provide advanced simulation environments to test and debug different networking protocols, including collision detection modules, radio propagation and MAC protocols. Some recent results question however the validity of simulations based on these tools.

Specifically, [CSS02] presents the simulative results of the flooding algorithm using OPNET, NS-2 and Glomosim. Important divergences between the simulators results have been measured. The observed differences are not only quantitative (not the same absolute value), but also qualitative (not the same general behavior) making some past observation of MANET simulation studies an open issue. Similar problems have been observed, in the framework of the MobileMAN project, also by CNR researchers. Specifically, they compare and contrast the performance of two MANET routing protocols, DSR and AODV, by using NS-2 and Glomosim. Results obtained with the two simulators were often very (quantitative and qualitative) different [DC02].

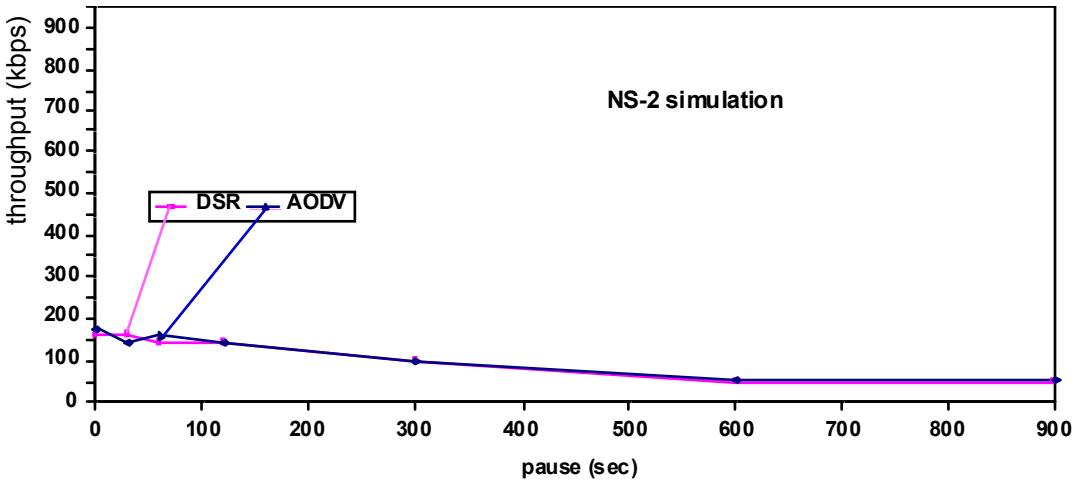


Figure 1.7: results of NS-2 simulator

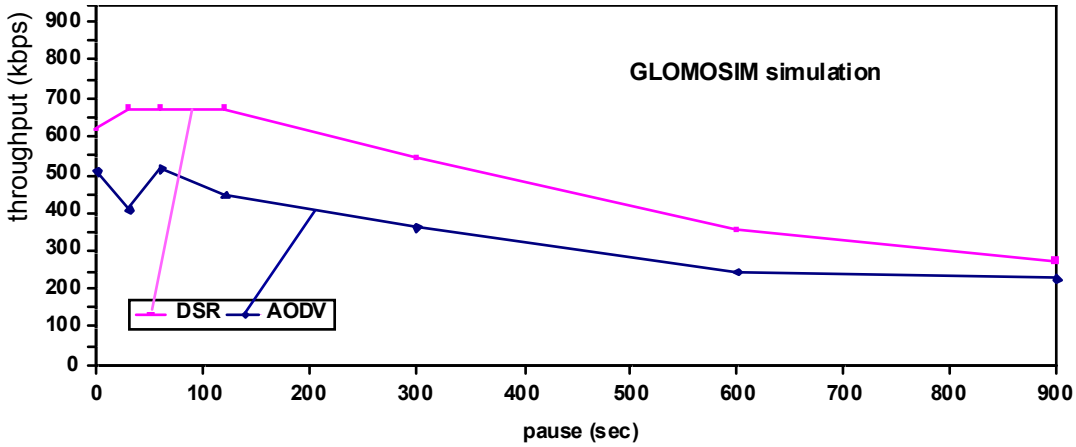


Figure 1.8: results of Glomosim simulator

An example of the obtained results is given in Figure 1.7 and Figure 1.8 showing the results obtained (in the same scenario) with NS-2 and Glomosim, respectively. Specifically, we considered an IEEE 802.11 network with 50 nodes moving according to the Random Waypoint mobility model in a 1500m x 300m closed rectangular area. The nodes maximum speed is equal to 1 m/sec. In the considered scenario there are 3 FTP sessions (1460 bytes per packet), and ten CBR sessions (512 bytes per packet). Figures 1.7 and 1.8 show the sum of the 3 FTP-session throughput

by varying the length of the pause in the Random Waypoint model.[5] As it clearly appears from the figures the obtained results are highly dependent on the simulation tool. For the above reasons, in this project we will use whenever possible measurement studies on real testbeds. Results from testbeds are also very important as they can point out problems that cannot be detected by simulation studies. For example, an important problem, related to the different transmission ranges for 802.11b control and data frames, is the so-called *communication gray zones* problem [LNT02]. The communication gray zone problem can not be revealed by most of commonly used simulation tools (e.g., NS-2, Glomosim) as in these 802.11 model both unicast and broadcast transmissions are performed at 2 Mbps, and hence have the same transmission range.

[5] The length of each simulative experiment is equal to 900 seconds, hence a 900-second pause implies a static configuration, i.e., the nodes do not move during the experiment.

1.3. References

[APE02] APE: Ad hoc Protocol Evaluation testbed. Department of Computer Systems at Uppsala, Sweden. http://apetestbed.sourceforge.net/

[BB97] A. V. Bakre and B. R. Badrinath, "Implementation and performance evaluation of indirect TCP," IEEE Trans. Computers, vol. 46, March 1997.

[BB03c] A. Boukerche, L. Bononi, "Simulation and Modeling of Wireless, Mobile and Ad Hoc Networks", in *Mobile Ad Hoc Networking*, S. Basagni, M. Conti, S. Giordano, I. Stojmenovic (Editors), IEEE Press and John Wiley and Sons, Inc., New York, 2003.

[BMJ00] Josh Broch, David A. Maltz, David B. Johnson, "Quantitative Lessons From a Full-Scale Multi-Hop Wireless Ad Hoc Network Testbed", Proceedings of the IEEE Wireless Communications and Network Conference 2000 (WCNC 2000).

[BMJHJ98] Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu, Jorjeta Jetcheva, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols", Proceedings of The Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM '98), October 25-30, 1998, Dallas, Texas, USA.

[C00] M. Scott Corson, "A Triggered Interface" <draft-corson-triggered-00.txt>

[CP02] Jon Crowcroft, Ian Pratt, "Peer to Peer: Peering into the Future" in Advanced Lectures on Networking, Enrico Gregori, Giuseppe Anastasi, Stefano Basagni (Editors) LNCS 2497, 2002.

[CCL03] I. Chlamtac, M. Conti, and J. Liu, "Mobile Ad Hoc Networking: Imperatives and Challenges", *Ad Hoc Networks*, No. 1(1) 2003.

[CD99] M. Conti, L. Donatiello, "Simulation modeling of local and metropolitan area networks", Chapter 6 in *Network Systems Design*, E. Gelenbe, K. Bagchi, and G. Zobrist (Editors), Gordon & Breach, Amsterdam, 1999, pp. 117-142.

[CSS02] David Cavin, Yoav Sasson, Andrè Schiper, "On the Accuracy of MANET Simulators", Proc. ACM POMC'02, Toulouse, France.October 2002,

[CZN02] K. Chen, S.H. Shah, K. Nahrstedt, "Cross-Layer Design for Data Accessibility in Mobile Ad Hoc Networks", *Wireless Personal Communications* 21: 49–76, 2002

[CRVP01] K. Chandran, S. Raghunathan, S. Venkatesan, R. Prakash, "A Feedback Based Scheme for Improving TCP Performance in Ad Hoc Wireless Networks", *IEEE Personal Communication Magazine*, Special Issue on Ad Hoc Networks, Vol. 8, N. 1, pp. 34-39, February 2001.

[DC02]. D. De Col, "Routing protocols for wireless ad hoc networks: performance evaluation of AODV and DSR", Computer Science Laura Thesis, University of Pisa, October 2002 (in Italian).

[DCY00] S. R. Das, R. Castaneda, J. Yan, "Simulation Based Performance Evaluation of Mobile, Ad Hoc Network Routing Protocols", ACM/Baltzer Mobile Networks and Applications (MONET) Journal, July 2000, pages 179-189.

[DPR00] Samir R. Das, Charles E. Perkins, Elizabeth M. Royer. "Performance Comparison of Two On-demand Routing Protocols for Ad Hoc Networks", Proceedings INFOCOM 2000, Tel Aviv, Israel, March 2000.

[FS03] Andras Farago, Violet Syrotiuk,,"MERIT: A Scalable Approach for Protocol Assessment", *ACM/Kluwer MONET* Vol. 8, No. 5 (Oct. 2003), Special issue on "Mobile Ad Hoc Network", A.T. Campbell, M. Conti, S. Giordano (Editors).

[FZX03] Zhenghua Fu, Petros Zerfos, Kaixin Xu, Haiyun Luo, Songwu Lu, Lixia Zhang, Mario Gerla, "The Impact of Multihop Wireless Channel on TCP Throughput and Loss", Proc. Infocom 2003, San Francisco, April 2003.

[GLOM] GloMoSim, Global Mobile Information Systems Simulation Library, http://pcl.cs.ucla.edu/projects/glomosim/.

[GM01] R. Gold, C. Mascolo, "Use of Context-Awareness in Mobile Peer-to-Peer Networks", in Proc. of the 8th IEEE Workshop on Future Trends of Distributed Computing Systems (FTDCS'2001), Bologna, Italy. October 2001.

[GW02] A.J. Goldsmith, S.B. Wicker, “Design Challenges for Energy-Constrained Ad Hoc Wireless Networks”, IEEE Wireless Communications, Volume 9, Number 4, August 2002. pp. 8- 27.

[HV02] Gavin Holland, Nitin H. Vaidya “Analysis of TCP Performance over Mobile Ad Hoc Networks”, ACM/Kluwer Journal of Wireless Networks 8(2-3), (2002) pp. 275-288.

[JLHM99] P. Johansson, T. Larsson, N. Hedman, B. Mielczarek “Routing Protocols for Mobile Ad-Hoc Networks – A Comparative Performance Analysis”, Proceedings of The Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM '99), August 15-19, 1999, Seattle, Washington, USA. pp. 195-206.

[KK03] Vikas Kawadia, P.R. Kumar, "A Cautionary Perspective on Cross Layer Design", Submitted to IEEE Wireless Communication Magazine. July, 2003.

[KM88] J.F. Kurose, H. Mouftah, “Computer-Aided Modeling of Computer Communication Networks”, IEEE Journal on Selected Areas in Communications, Vol. 6, No. 1 (January, 1988), pp. 130-145.

[L83] S.S. Lavenberg, *Computer Performance Handbook*, Academic Press, New York, 1983.

[LNT02] H. Lundgren, E. Nordstron, C. Tschudin, “Coping with Communication Gray Zones in IEEE 802.11 based Ad Hoc Networks”, Proceedings of the *ACM Workshop on Mobile Multimedia (WoWMoM 2002)*, Atlanta (GA), September 28, 2002, pp. 49-55.

[MC03] J.P. Macker, S. Corson, “Mobile Ad hoc Networks (MANET): Routing technology for dynamic, wirelessnetworking”, in *Mobile Ad hoc networking*, S. Basagni, M. Conti, S. Giordano, I. Stojmenovic (Editors), IEEE Press and John Wiley and Sons, Inc., New York, 2003.

[MCE02] Cecilia Mascolo, Licia Capra, Wolfgang Emmerich, “Middleware for Mobile Computing (A Survey)” in Advanced Lectures on Networking, Enrico Gregori, Giuseppe Anastasi, Stefano Basagni (Editors) LNCS 2497, 2002.

[MPR01] A. L. Murphy, G. P. Picco, G.-C. Roman, “Lime: A middleware for physical and logical mobility,” in Proceedings of the 21st International Conference on Distributed Computing Systems (ICDCS-21), Phoenix, AZ, USA, pp. 524–233, April 16-19 2001

[NS2] The Network Simulator - ns-2, http://www.isi.edu/nsnam/ns/index.html.

[OPN] OPNET Modeler. http://www.opnet.com/products/modeler/home.html.

[YLA02] Wing Ho Yuen, Heung-no Lee, Timothy D. Andersen, “A Simple and Effective Cross Layer Networking System for Mobile Ad Hoc Networks”, Proc. of IEEE PIMRC, 2002.

2. WIRELESS TECHNOLOGIES

A mobile ad hoc network (MANET) represents a system of wireless mobile nodes that can freely and dynamically self-organize into arbitrary and temporary network topologies, allowing people and devices to seamlessly internetwork in areas without any pre-existing communication infrastructure. While many challenges remain to be resolved before large scale MANETs can be widely deployed, small scale, mobile ad hoc networks will soon appear. Network cards for single-hop ad hoc wireless networks are already available on the market, and these technologies constitute the building blocks to construct small scale ad-hoc network that extend the range of the single-hop wireless technologies to few kilometers.

As shown in Figure 2.1, we can classify ad hoc networks, depending on their coverage area, into several classes: Body (BAN), Personal (PAN), Local (LAN), Metropolitan (MAN) and Wide (WAN) area networks.

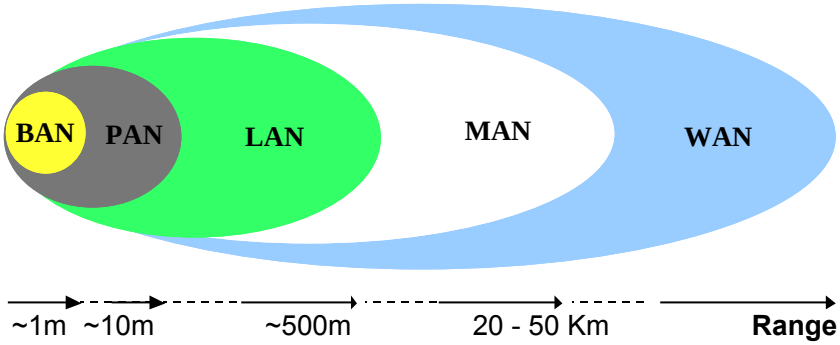


Figure 2.1: Ad Hoc Networks Taxonomy

The ad hoc network size in terms of the number of active nodes is the other metric used to classify MANETs. As defined in [MC03], we can classify the scale of an ad hoc network as small-scale (i.e., 2-20 nodes), moderate-scale (i.e., 20-100 nodes), large-scale (i.e., 100+ nodes), and very large-scale (i.e., 1000+ nodes). In [GK00], it was shown that in an ad hoc network with n nodes the per-node throughput is bounded by $c/\sqrt{n}$, where c is a constant. Unfortunately, experimental results [GGK01] indicate that with current technologies the per-node throughput decays as $c'/n^{1.68}$ and hence, with current technologies, only small- and moderate-scale can be implemented in an efficient way.

Wide- and Metropolitan-area ad hoc networks are mobile multi-hop wireless networks that present many challenges that are still to be solved (e.g., addressing, routing, location management, security, etc.), and their availability is not on immediate horizon. On the other hand, mobile ad hoc networks with smaller coverage can be expected to appear soon. Specifically, ad-hoc single-hop BAN, PAN and LAN wireless technologies are already common on the market [C03], and these technologies constitute the building blocks for constructing small/medium, multi-hop, ad hoc networks that extend their range over multiple radio hops [CMC99]. For these reasons, BAN, PAN and LAN technologies constitute the *Enabling Technologies* for ad hoc networking. A detailed discussion of Body, Personal, and Local Ad hoc Wireless Networks can be found in [C03]. Hereafter, the characteristics of these networks, and the technologies available to implement them, are summarized.

A body area network is strongly correlated with wearable computers. A wearable computer distributes on the body its components (e.g., head-mounted displays, microphones, earphones, etc.), and the BAN provides the connectivity among these devices. The communicating range of a BAN corresponds to the human body range, i.e., 1-2 meters. As wiring a body is generally cumbersome, wireless technologies constitute the best solution for interconnecting wearable devices.

Personal area networks connect mobile devices carried by users to other mobile and stationary devices. While a BAN is devoted to the interconnection of one-person wearable devices, a PAN is a network in the environment around the persons. A PAN communicating range is typically up to ten meters, thus enabling the interconnection of the BANs of persons close to each other, and the interconnection of a BAN with the environment around it.

The most promising radios for widespread PAN deployment are in the 2.4 GHz ISM band. Spread spectrum is typically employed to reduce interference and to increase bandwidth re-use.

Wireless LANs (*WLANs*) have a communication range typical of a single building, or a cluster of buildings, i.e., 100-500 meters. A WLAN should satisfy the same requirements typical of any LAN, including high capacity, full connectivity among attached stations, and broadcast capability. However, to meet these objectives, WLANs need to be designed to face some issues specific to the wireless environment, like security on the air, power consumption, mobility, and bandwidth limitation of the air interface [S96].

Two different approaches can be followed in the implementation of a WLAN: *infrastructure-based* or *ad hoc networking* [S96]. An infrastructure-based architecture imposes the existence of a centralized controller for each cell, often referred to as *Access Point*. The Access Point (AP) is normally connected to the wired network, thus providing the Internet access to mobile devices. In contrast, an ad hoc network is a peer-to-peer network formed by a set of stations within the range of each other, which dynamically configure themselves to set up a temporary network. In the ad hoc configuration, no fixed controller is required, but a controller may be dynamically elected among the stations participating in the communication.

The success of a network technology is connected to the development of networking products at a competitive price. A major factor in achieving this goal is the availability of appropriate networking standards. Currently, two main standards are emerging for ad hoc wireless networks: the IEEE 802.11 standard for WLANs [IEEE802], and the Bluetooth specifications[6] [BLU] for short-range wireless communications [B01][BLU01][MB00]. In addition to the IEEE standards, the European Telecommunication Standard Institute (ETSI) has promoted the HiperLAN (HIgh Performance Radio Local Area Network) family of standards for WLANs [ETSI]. Among these, the most interesting standard for WLAN is HiperLAN/2. The HiperLAN/2 technology addresses high-speed wireless network with data rates ranging from 6 to 54 Mbit/s. Infrastructure-based and ad hoc networking configurations are both supported in HiperLAN/2.To a large degree, HiperLAN is still at the prototype level, and hence we will not consider it more in detail. More details on this technology can be found in [EM02]. [ZD03] surveys the off-the-shelf technologies for constructing ad hoc networks.

Due to its extreme simplicity, the IEEE 802.11 standard is a good platform to implement a single-hop WLAN ad hoc network. Furthermore, multi-hop networks covering areas of several square kilometers can potentially be built by exploiting the IEEE 802.11 technology. Currently, the widespread use of IEEE 802.11 cards makes this technology the most interesting off-the-shelf enabler for ad hoc networks. For this reason in the MobileMAN project we use 802.11 as the reference technology.

The IEEE 802.11 standard defines two operational modes for WLANs: *infrastructure-based* and *infrastructure-less* or *ad hoc*. Network interface cards can be set to work in either of these modes but not in both simultaneously. The infrastructure-based is the mode commonly used to construct the so called Wi-Fi hotspots, i.e., to provide wireless access to the Internet. The standardization efforts concentrated on solutions for infrastructure-based WLANs, while little or no attention was given to the ad hoc mode. This section is therefore devoted to an in-depth investigation of the ad hoc features of the IEEE 802.11 standard to study its effectiveness to construct ad hoc networks,

[6] The Bluetooth specifications are released by the Bluetooth Special Interest Group.

and to propose and investigate solutions for enhancing this technology for a better support of the ad hoc networking paradigm.

2.1. IEEE 802.11 Architecture and Protocols

In 1997, the IEEE adopted the first wireless local area network standard, named IEEE 802.11, with data rates up to 2Mbps [IEEE97]. Since then, several task groups (designated by the letters from *a*, *b*, *c*, etc.) have been created to extend the IEEE 802.11 standard. The task groups 802.11b and 802.11a have completed their work by providing two relevant extensions to the original standard [IEEE802]. The 802.11b task group produced a standard for WLAN operations in 2.4 GHz band that extends IEEE 802.11 to data rates up to 11 Mbps. This standard, published in 1999, has been very successful. Currently, there are several IEEE 802.11b products available on the market. The 802.11a task group created a standard for WLAN operations in the 5 GHz band, with data rates up to 54 Mbps. Among the other task groups, it is worth mentioning the task group 802.11e (that attempts to enhance the MAC with QoS features to support voice and video over 802.11 networks), and the task group 802.11g that has just defined a higher speed extension to the 802.11b.

Currently, IEEE 802.11b, which is also known with the friendly name of Wireless Fidelity (Wi-Fi), is the de-facto reference technology for wireless LAN networks. For this reason hereafter we will focus on the IEEE 802.11 architecture and protocols as defined in the original standard [IEEE97], and then we will emphasize the differences between the 802.11b standard with respect to the original 802.11 standard.

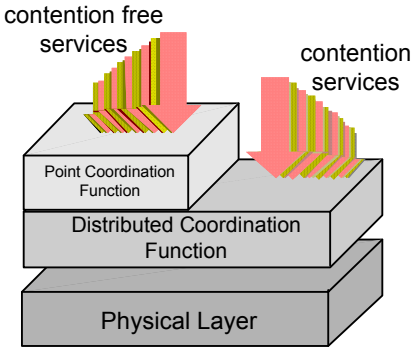


Figure 2.2: IEEE 802.11 Architecture

The IEEE 802.11 standard specifies both the MAC layer and the Physical Layer (see Figure 2.2). The MAC layer offers two different types of service: a contention-based service provided by the *Distributed Coordination Function* (DCF), and a contention-free service implemented by the *Point Coordination Function* (PCF). These service types are made available on top of a variety of physical layers. Specifically, three different technologies have been specified in the standard: Infrared (IF), Frequency Hopping Spread Spectrum (FHSS) and Direct Sequence Spread Spectrum (DSSS).

The DCF provides the basic access method of the 802.11 MAC protocol and is based on a *Carrier Sense Multiple Access with Collision Avoidance* (CSMA/CA) scheme. The PCF is implemented on top of the DCF and is based on a polling scheme. It uses a *Point Coordinator* that cyclically polls stations, giving them the opportunity to transmit. Since the PCF can not be adopted in ad hoc mode, it will not be considered hereafter.

2.1.1. Distributed Coordination Function (DCF)

According to the DCF, before transmitting a data frame, a station must sense the channel to determine whether any other station is transmitting. If the medium is found to be idle for an interval longer than the *Distributed InterFrame Space* (*DIFS*), the station continues with its

transmission[7] (see Figure 2.3). On the other hand (i.e., if the medium is busy), the transmission is deferred until the end of the ongoing transmission. A random interval, henceforth referred to as the *backoff time,* is then selected, which is used to initialize the *backoff timer.* The backoff timer is decreased for as long as the channel is sensed as idle, stopped when a transmission is detected on the channel, and reactivated when the channel is sensed as idle again for more than a DIFS (for example, the backoff timer of Station 2 in Figure 2.3 is disabled while Station 3 is transmitting its frame; the timer is reactivated a DIFS after Station 3 has completed its transmission). The station is enabled to transmit its frame when the backoff timer reaches zero. The backoff time is slotted. Specifically, the backoff time is an integer number of slots uniformly chosen in the interval (0, *CW*-1). *CW* is defined as the Backoff Window, also referred to as *Contention Window*. At the first transmission attempt $CW=CW_{min}$, and it is doubled at each retransmission up to CW_{max}. In the standard CW_{min} and CW_{max} values depend on the physical layer adopted. For example, for the FHSS Physical Layer Cw_{min} and Cw_{max} values are 16 and 1024, respectively [IEEE97].

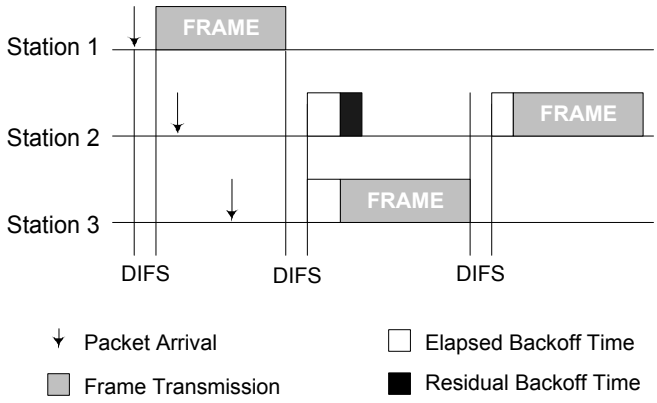


Figure 2.3. Basic Access Mechanism

Obviously, it may happen that two or more stations start transmitting simultaneously, and hence a collision occurs. In the CSMA/CA scheme, stations are not able to detect a collision by hearing their own transmissions (as in the CSMA/CD protocol used in wired LANs). Therefore, an immediate positive acknowledgement scheme is employed to ascertain the successful reception of a frame. Specifically, upon reception of a data frame, the destination station initiates the transmission of an acknowledgement frame (ACK) after a time interval called *Short InterFrame Space* (SIFS). The SIFS is shorter than the DIFS (see Figure 2.4) in order to give priority to the receiving station over other possible stations waiting for transmission. If the ACK is not received by the source station, the data frame is presumed to have been lost, and a retransmission is scheduled. The ACK is not transmitted if the received packet is corrupted. A Cyclic Redundancy Check (*CRC*) algorithm is used for error detection.

After an erroneous frame is detected (due to collisions or transmission errors), a station must remain idle for at least an *Extended InterFrame Space* (EIFS) interval before it reactivates the backoff algorithm. Specifically, the EIFS shall be used by the DCF whenever the physical layer has indicated to the MAC that a frame transmission was begun that did not result in the correct reception of a complete MAC frame with a correct FCS value. Reception of an error-free frame during the EIFS re-synchronizes the station to the actual busy/idle state of the medium, so the EIFS is terminated and normal medium access (using DIFS and, if necessary, backoff) continues following reception of that frame.

[7] To guarantee fair access to the shared medium, a station that has just transmitted a packet and has another packet ready for transmission must perform the backoff procedure before initiating the second transmission.

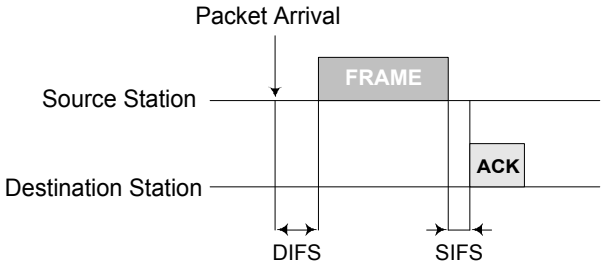


Figure 2.4: The SIFS is shorter than the DIFS

2.1.2. Common Problems in Wireless Ad Hoc Networks

In this section we discuss some problems that can arise in wireless networks, mainly in the ad hoc mode. The characteristics of the wireless medium make wireless networks fundamentally different from wired networks. Specifically, as indicated in [IEEE97]:

- the wireless medium has neither absolute nor readily observable boundaries outside of which stations are known to be unable to receive network frames;
- the channel is unprotected from outside signals;
- the wireless medium is significantly less reliable than wired media;
- the channel has time-varying and asymmetric propagation properties.

In a wireless (ad hoc) network that relies upon a carrier-sensing random access protocol, like the IEEE 802.11 DCF protocol, the wireless medium characteristics generate complex phenomena such as the hidden-station and exposed-station problems.

Figure 2.5 shows a typical "hidden station" scenario. Let us assume that station B is in the transmitting range of both A and C, but A and C cannot hear each other. Let us also assume that A is transmitting to B. If C has a frame to be transmitted to B, according to the DFC protocol, it senses the medium and finds it free because it is not able to hear A's transmissions. Therefore, it starts transmitting the frame but this transmission will result in a collision at the destination Station B.

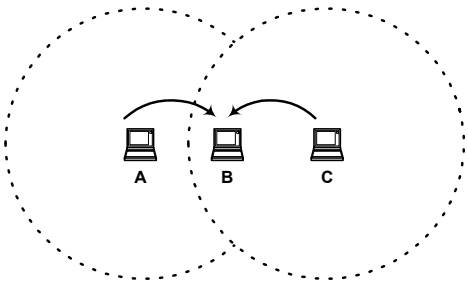


Figure 2.5: The "hidden station" problem

The hidden station problem can be alleviated by extending the DCF basic mechanism by a **virtual carrier sensing** mechanism (also referred to as floor acquisition mechanism) that is based on two control frames: *Request To Send* (*RTS*) and *Clear To Send* (*CTS*), respectively. According to this mechanism, before transmitting a data frame, the station sends a short control frame, named RTS, to the receiving station announcing the upcoming frame transmission (see Figure 2.6). Upon receiving the RTS frame, the destination station replies by a CTS frame to indicate that it is ready to receive the data frame. Both the RTS and CTS frames contain the total duration of the transmission, i.e., the overall time interval needed to transmit the data frame and the related ACK. This information can be read by any listening station that uses this information to set up a timer called *Network Allocation Vector* (*NAV*). While the NAV timer is greater than zero the station must

refrain from accessing the wireless medium. By using the RTS/CTS mechanism, stations may become aware of transmissions from hidden station and on how long the channel will be used for these transmissions.

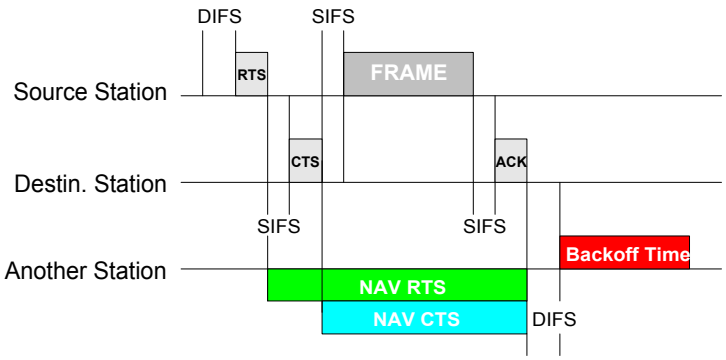


Figure 2.6: Virtual Carrier Sensing mechanism

Figure 2.7 depicts a typical scenario where the "exposed station" problem may occur. Let us assume that both Station A and Station C can hear transmissions from B, but Station A can not hear transmissions from C. Let us also assume that Station B is transmitting to Station A and Station C receives a frame to be transmitted to D. According to the DCF protocol, C senses the medium and finds it busy because of B's transmission. Therefore, it refrains from transmitting to C although this transmission would not cause a collision at A. The "exposed station" problem may thus result in a throughput reduction.

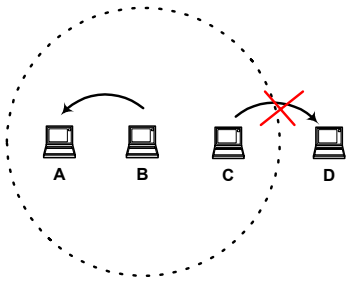


Figure 2.7: The "exposed station" problem

2.1.3. Ad Hoc Networking Support

In this section we will describe how two or more 802.11 stations set up an ad hoc network. In the IEEE 802.11 standard, an ad hoc network is named *Independent Basic Service Set* (IBSS). An IBSS enables two or more 802.11 stations to communicate each other without the intervention of either a centralized AP, or an infrastructure network. Hence, the IBSS can be considered as the support provided by the 802.11 standard for mobile ad hoc networking.[8]

Due to the flexibility of the CSMA/CA protocol, to receive and transmit data correctly it is sufficient that all stations within the IBSS are synchronized to a common clock. The standard specifies a Timing Synchronization Function (TSF) to achieve clock synchronization between stations. In an infra-structured network the clock synchronization is provided by the AP and all

[8] To uniquely identify a IBSS it is necessary to associate to it an identification number (IBSSID) that is locally administered and that will be used by any other Station to join the IBSS, i.e., the ad hoc network. When a station starts a new IBSS, it generates a 46-bit random number in a manner that minimizes the probability that the same number is generated by another station.

stations synchronizes their own clock to the AP's clock. In an IBSS, due to the lack a centralized station, clock synchronization is achieved through a distributed algorithm. In both cases synchronization is obtained by transmitting special frames, called *beacons,* containing timing information.

The TSF requires two fundamental functionalities, namely *synchronization maintenance* and *synchronization acquirement*, that will be sketched below. We only focus on IBSS.

Synchronization maintenance

Each station has a *TSF timer* (clock) with modulus 2^{64} counting in increments of microseconds. Stations expect to receive beacons at a nominal rate defined by the BeaconPeriod parameter. This parameter is decided by the station initiating the IBSS, and is then used by any other station joining the IBSS. Stations use their TSF timers to determine the beginning of beacon intervals or periods. At the beginning of a beacon interval each station performs the following procedure:

- it suspends the decrementing of the backoff timer for any pending (non-beacon) transmission;
- it generates a random delay interval uniformly distributed in the range between zero and twice the minimum value of the Contention Window.
- it waits for the random delay;
- if a beacon arrives before the random delay timer has expired, it stops the random delay timer, cancel the pending beacon transmission, and resumes the backoff timer;
- if the random delay timer has expired and no beacon has been received, it sends a beacon frame.

The sending station sets the beacon timestamp to the value of its TSF timer at the time the beacon is transmitted. Upon reception of a beacon, the receiving station looks at the timestamp. If the beacon timestamp is later than the station's TSF timer, the TSF timer is set to the value of the received timestamp. In other words, all stations within the IBSS synchronize their TSF timer to the quickest TSF timer.

Synchronization acquirement

This functionality is necessary when a station wants to join an already existing IBSS. The discovery of existing IBSSs is the result of a scanning procedure of the wireless medium during which the station receiver is tuned to different radio frequencies, looking for particular control frames. Only if the scanning procedure does not result in finding any IBSS, the station may start with the creation of a new IBSS. The scanning procedure can be either passive or active.

In a passive scanning the station listens to the channels for hearing a beacon frame. It is worth reminding that a beacon frame contains not only timing information for synchronization, but also the complete set of IBSS parameters. This set includes the IBSS identifier IBSSID, the BeaconPeriod parameter, the data rates that can be supported, the parameters relevant to IBSS management functions (e.g., power saving management).

Active scanning involves the generation of Probe frames, and the subsequent processing of received Probe Response frames. The station that decides to start an active scanning procedure has a ChannelList of radio frequencies that will be scanned during the procedure. For each channel to be scanned a probe with broadcast destination is sent by using the DCF access method. At the same time a ProbeTimer is started. If no response to the probe is received before the ProbeTimer reaches the MinChannelTime the next channel of the list is considered. Otherwise, the station continues to scan the same channel until the timer reaches the MaxChannelTime. Then, the station processes all received Probe responses.

Probe responses are sent using normal frame transmission rules as directed frames to the address of the station that generated the Probe request. In an IBSS, only the station that generated the last beacon transmission will respond to a probe request, in order to avoid the waste of bandwidth with repetitive control frames. In each IBSS, at least one station must be awake at any given time to respond to Probe request. Therefore, the station that sent the last beacon remains in the awake state in order to respond to Probe requests, until a new beacon is received. There may be more than one station in a IBSS that responds to a given probe request, particularly in the case where more than one station transmitted a beacon, either due to not receiving successfully a previous beacon, or due to collision between beacon transmissions.

2.1.4. Power Management

In a mobile environment, portable devices have limited energetic resources since they are powered through batteries. Power management functionalities are thus extremely important both in the infrastructure-based and in the ad hoc modes. Obviously, in the ad hoc mode, i.e., inside an IBSS, Power Saving (PS) strategies need to be completely distributed in order to preserve the self-organizing nature of the IBSS. A station may be in one of two different power states: *awake* (station is fully powered) or *doze* (the station is not able to transmit or receive). Multicast and/or directed frames destined to a power-conserving station are first announced during a period when all stations are awake. An Ad hoc Traffic Indication Map (ATIM) frame does the announcement. A station operating in the PS mode listens to these announcements and, based on them, decides whether it has to remain awake or not.

ATIM frames are transmitted during the ATIM Window, a specific period of time following the beginning of a Beacon period whose length is defined by the ATIMWindow parameter (an IBSS parameter included in the beacon content). During the ATIM Window, only beacon and ATIM frames can be exchanged and all stations must remain awake. Directed ATIM frames are to be acknowledged by the destination station, while multicast ATIMs are not to be acknowledged. Hence a station sends a directed ATIM frame and waits for the acknowledgement. If this acknowledgement does not arrive it executes the backoff procedure for re-transmitting the ATIM frame.

A station receiving a directed ATIM frame must send the acknowledgement and remain awake for the entire duration of the beacon interval, waiting for the announced data frame. Data frames are transmitted at the end of the ATIM Window according to the DCF access method (see Figure 2.8). If a station does not receive any ATIM frame during the ATIM Window it, can enter the doze state at the end of the ATIM window.

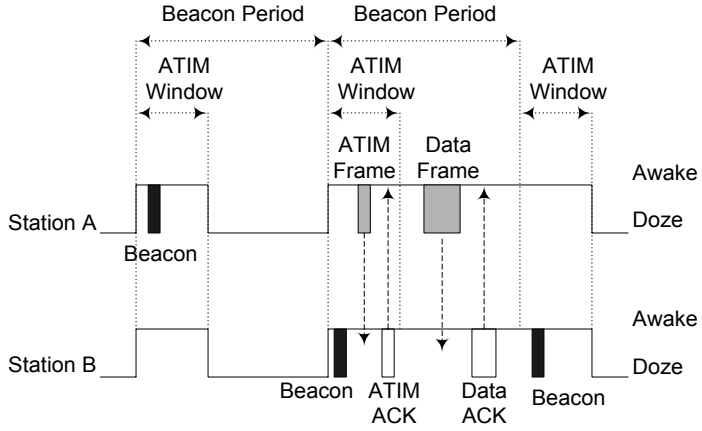


Figure 2.8: A data exchange between stations operating in PS mode in an ad hoc network

2.1.5. IEEE802.11and IEE802.11b

The 802.11b standard extends the 802.11 standard by introducing a higher-speed Physical Layer in the 2.4 GHz frequency band still guaranteeing the interoperability with 802.11 cards. Specifically, 802.11b enables transmissions at 5.5 Mbps and 11 Mbps, in addition to 1 Mbps and 2 Mbps. 802.11b cards may implement a dynamic rate switching with the objective of improving performance. To ensure coexistence and interoperability among multirate-capable stations, and with 802.11 cards, the standard defines a set of rules that must be followed by all stations in a WLAN. Specifically, for each WLAN is defined a *basic rate set* that contains the data transfer rates that all stations within the WLAN will be capable of using to receive and transmit.

To support the proper operation of a WLAN, all stations must be able to detect control frames. Hence, RTS, CTS, and ACK frames must be transmitted at a rate included in the basic rate set. In addition, also frames with multicast or broadcast destination addresses must be transmitted at a rate belonging to the basic rate set. These differences in the rates used for transmitting (unicast) data and control frames have a big impact on the system behavior as clearly pointed out in [E02]. Our experimental results presented below show the impact of different transmission rates on the IEEE 802.11b behavior.

2.2. Analysis of 802.11 performance

Two main performance indices are used to analyze the performance of a technology for wireless LANs: the throughput and the delay.

As far as throughput is concerned, a special attention will be deserved to the Medium Access Control (MAC) protocol *capacity* ([KSY84], [CGL97]) defined as: *the maximum fraction of channel bandwidth used by successfully transmitted messages*. This performance index is important as wireless networks deliver much lower bandwidth than wired networks, e.g. 1-11 Mbps vs. 100-1000 Mbps [S96]. Since a WLAN relies on a common transmission medium, the transmissions of the network stations must be coordinated by the MAC protocol. This coordination can be achieved by means of control information that is carried explicitly by control messages traveling along the medium (e.g. ACK messages), or can be provided implicitly by the medium itself using the carrier sensing to identify the channel being either active or idle. Control messages, or message retransmissions due to collision, remove channel bandwidth from that available for successful message transmission. Therefore, the capacity gives a good indication of the overheads required by the MAC protocol to perform its coordination task among stations, or in other words of the effective bandwidth that can be used on a wireless link for data transmission.

The delay can be defined in several forms (access delay, queuing delay, propagation delay, etc.) depending on the time instants considered during its measurement, see [CGL97]. In computer networks the response time (i.e., the time between the generation of a message at the sending station, and its reception at the destination station) is the best figure to measure the Quality of Service (*QoS*) perceived by the users. However, the response time depends on the amount of buffering inside the network, and it is not always meaningful to evaluate a LAN technology. For example, during congested periods, the buffers fill up, and thus the response time does not depend on the LAN technology but it is mainly a function of the buffer length. For this reason, hereafter, the MAC delay index is used. The MAC delay of a station in a LAN is defined as: *the time between the instant at which a packet comes to the head of the station transmission queue and the end of the packet transmission* [CGL97].

2.2.1. 802.11 Protocol Capacity

The IEEE 802.11 protocol capacity was extensively investigated in [CCG00]. Hereafter, the main results of that analysis are summarized. In [CCG00] to study the protocol capacity it was defined a

p-persistent IEEE 802.11 protocol. This protocol differs from the standard protocol only in the selection of the backoff interval. Instead of the binary exponential backoff used in the standard, the backoff interval of the *p*-persistent IEEE 802.11 protocol is sampled from a geometric distribution with parameter *p*. Furthermore, in [CCG00], it was shown that a *p*-persistent IEEE 802.11 protocol closely approximates the standard protocol with the same average backoff window size. By developing an analytical model for the *p*-persistent IEEE 802.11 protocol, in [CCG00] it is derived the *p* value corresponding to the *theoretical upper bound*, i.e. the *p* value ($p_{\min}$) that maximizes the capacity of the *p*-persistent IEEE 802.11 protocol. Due to the correspondence (from the capacity standpoint) between the standard protocol and the *p*-persistent one, the theoretical upper bound constitutes also a throughput limit for tuning the IEEE 802.11 protocol. Specifically, the throughput limit is achieved by an IEEE 802.11 protocol whose average backoff window size (hereafter, *optimal average backoff window size*) is equal to the average backoff of the *p*-persistent IEEE 802.11 protocol using the *optimal p* value, $p_{\min}$.

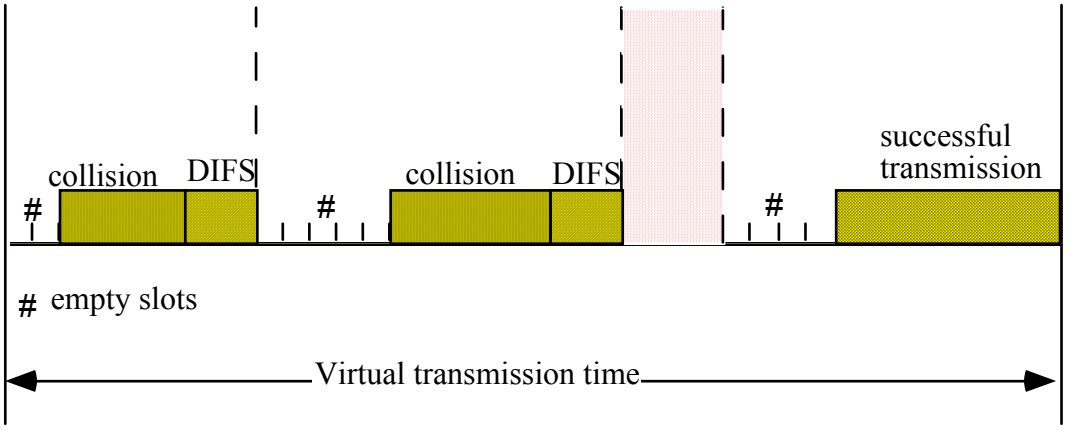


Figure 2.9: Structure of a virtual transmission time

In the following, for ease of reading, we briefly summarize the procedure used to derive the $p_{\min}$ value. For more details see [CCG00]. The IEEE 802.11 MAC protocol capacity is analytically estimated by developing a model with a finite number, *M*, of stations operating in *asymptotic conditions*. This means that all the *M* network stations always have a packet ready for transmission. The computation of the protocol capacity, presented in [CCG00], is performed by observing the system at the end of each successful transmission assuming that packet lengths are i.i.d. random variables sampled from a geometric distribution with parameter *q*. The time interval between two successful transmissions is referred to as *virtual transmission time*. A virtual transmission time includes a successful transmission and may include several collision intervals (see Figure 2.9). In the rest of this section we will use the following notation:

- $Idle\_p$ is the number of consecutive empty slots;
- $Coll$ is the time the channel is busy due to a collision given that a transmission attempt occurs, also referred to as *collision cost*. Obviously, $Coll$ is equal to zero if the transmission attempt is successful, otherwise it is equal to the collision length;
- $p_{collision}$ is the probability of a collision given that a transmission attempt occurs;
- t_{slot} is the time duration of a slot;
- $E[\ ]$ denotes the average operator, i.e., given a random variable X, $E[X]$ is its average;

From the geometric backoff assumption all the processes that define the occupancy pattern of the channel (i.e. empty slots, collisions and successful transmissions) are regenerative with respect to the sequence of time instants corresponding to the completion of a successful transmission. The protocol capacity is thus [HS82]:

$$\rho_{\max} = \frac{\overline{m}}{E[t_v]} \tag{2.1}$$

where $E[t_v]$ is the average virtual transmission time, and $\overline{m}$ is the average message length. As shown in [CCG00], see also $E[t_v]$ can be written as:

$$E[t_v] = E[N_c]\{E[Coll]_{|Collision} + \tau + DIFS\} + E[Idle\_p]\cdot(E[N_c]+1) + E[S] \quad , \tag{2.2}$$

where $E[Coll]_{|Collision}$ is the average collision length given that a collision occurs, $E[N_c]$ is the average number of collisions in a virtual transmission time, $E[Idle\_p]$ is the average number of consecutive idle slots, τ is the propagation delay, and $E[S]$ is the time required to complete a successful transmission (including all the protocol overheads).

By taking into consideration the protocol behavior, it can be verified that $E[S] \approx \overline{m} + 2\tau + SIFS + ACK + DIFS$. The analytical formulas for the other unknown quantities of Equation (2.1) are defined in Lemma 1 whose proof can be found [CCG00].

LEMMA 1. *In a network with M active stations, by assuming that for each station i) the backoff interval is sampled from a geometric distribution with parameter p, and ii) packet lengths are i.i.d. random variables sampled from a geometric distribution with parameter q:*

$$E[N_c] = \frac{1-(1-p)^M}{Mp(1-p)^{M-1}} - 1$$

$$E[Coll]_{|collision} = \frac{t_{slot}}{1-\left[(1-p)^M + Mp(1-p)\right]^{M-1}} \cdot \left[\sum_{h=1}^{\infty}\left\{h\cdot\left[\left(1-pq^h\right)^M - \left(1-pq^{h-1}\right)^M\right]\right\} - \frac{Mp(1-p)^{M-1}}{1-q}\right]$$

$$E[Idle\_p] = \frac{(1-p)^M}{1-(1-p)^M}$$

From Equation (2.2) and Lemma 1, it results that $E[t_v]$ is a function of the parameters *M*, *p* and *q*. Hence, for a given network configuration (i.e. number of active stations, *M*) and for a given traffic configuration (i.e. the value of *q* that characterizes the average message length), $E[t_v]$ is only a function of the *p* value, and (with standard procedures) we can compute the value of *p*, say $p_{\min}$, which minimizes the $E[t_v]$. As $\overline{m}$ does not depend on *p*, from Equation (2.1) it follows that $p_{\min}$ is also the *p* value that maximizes the protocol capacity.

Since the exact $p_{\min}$ derivation is expensive from a computational standpoint, in [BCGG03], it is shown that a close approximation of $p_{\min}$ is given by the *p* value that satisfies the following relationship:

$$E[Coll]_{|Collision} \cdot E[N_c] = (E[N_c]+1)\cdot E[Idle\_p]\cdot t_{slot} \tag{2.3}$$

Equation (2.3) expresses the following condition: $p_{\min}$ is the *p* value for which, inside a virtual transmission time, the average time the channel is idle equates the average time the channel is busy due to the collisions.

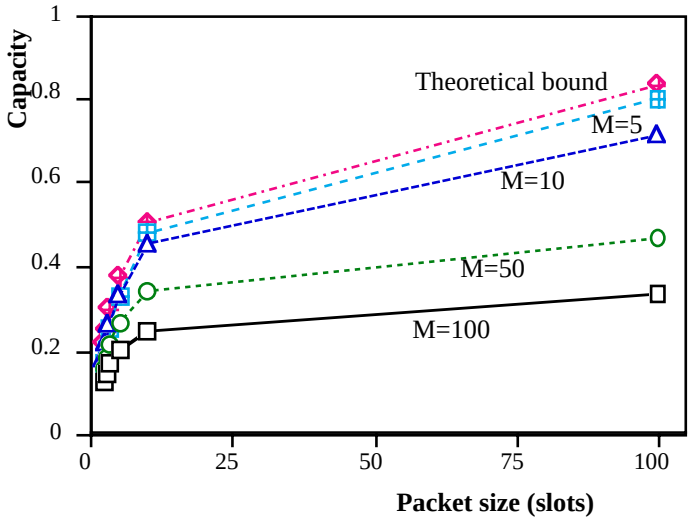


Figure 2.10. IEEE 802.11 performance: protocol capacity

By applying the above formulas in [CCG00] it was analytically derived the theoretical throughput limit for IEEE 802.11 networks,[9] and this limit was compared with the simulative estimates of the real protocol capacity. Results show that, depending on the network configuration, the standard protocol can operate very far from the theoretical throughput limit. These results, summarized in Figure 2.10, indicate that the distance between the IEEE 802.11 and the analytical bound increases with the number of active stations, *M*. In the IEEE 802.11 protocol, due to its backoff algorithm, the average number of stations that transmit in a slot increases with *M*, and this causes an increase in the collision probability. A significant improvement of the IEEE 802.11 performance can thus be obtained by controlling the number of stations that transmit in the same slot.

As the optimal *p* value (and hence the optimal average backoff window size in the standard protocol) depends on the traffic conditions, the optimal protocol capacity can only be achieved if the backoff window is dynamically tuned at run-time following the evolution of the network traffic conditions. In [CCG00] it was shown that, if a station has an exact knowledge of the network status it is able to tune its backoff algorithm to achieve a protocol capacity very close to its theoretical bound. Unfortunately, in a real case, a station does not have an exact knowledge of the network and load configuration but it, at most, can estimate it. Several works have shown that an appropriate tuning of the IEEE 802.11 backoff algorithm can significantly increase the protocol capacity (e.g., [BFO96], [CCG00a]). In particular, in [CCG00a], it was presented and evaluated a distributed algorithm, named *Dynamic IEEE 802.11 protocol*, to tune at run time the size of the backoff window. Specifically, by observing the status of the channel, each station gets an estimate of both the number of active stations, and the characteristics of the network traffic. By exploiting these estimates, each station then applies a distributed algorithm to tune its backoff window size in order to achieve the *theoretical throughput limit* for the IEEE 802.11 network. The Dynamic IEEE 802.11 protocol is complex due to the interdependencies among the estimated quantities [CCG00a]. To avoid this complexity, the *Simple Dynamic IEEE 802.11 Protocol* (*SDP*) was designed [BCG01]. The major difference with the Dynamic IEEE 802.11 protocol is that SDP does not need to estimate the number of active stations. Specifically, SDP only needs an estimate of the average time the channel is idle and the average time the channel is busy due to collisions. These two quantities can be directly estimated from the carrier sensing mechanism implemented in each station.

[9] That is, the maximum throughput that can be achieved by adopting the IEEE 802.11 MAC protocol, and using the optimal tuning of the backoff algorithm.

The Simple Dynamic IEEE 802.11 Protocol (SDP)

Before presenting the backoff-tuning algorithm we need to introduce some definitions. We denote as *transmission interval* the time interval between two consecutive transmission attempts. Hence a transmission interval, from the channel status standpoint, is made up of two components (see Figure 2.9: an idle period and a busy period. An idle period is made up of consecutive idle slots, while the busy period is an interval in which the channel is busy due to either a collision or a successful transmission.

As explained before, in [BCG03] it is shown that the optimal network operating point corresponds to the point in which the time wasted on idle periods is equal to the time spent on collisions:

$$E[Coll] = E[Idle\_p] \cdot t_{slot} \tag{2.4}$$

By exploiting classical probabilistic reasoning, closed form expressions can be derived for the above quantities contained in Equation (2.4). Specifically, in a system with M active stations, all adopting the same p -value for the backoff algorithm, it can be proved that (see for example [CCG00]):

$$E[Idle\_p] = \frac{(1-p)^M}{1-(1-p)^M}$$

$$P_{Collision} = \mathrm{P}\{\text{Transmitting Stations} \geq 2 \mid \text{Transmitting Stations} \geq 1\} = \tag{2.5}$$

$$= \frac{1-(1-p)^M - Mp(1-p)^{M-1}}{1-(1-p)^M} \quad ,$$

and $E[Coll] = E[Coll]_{|collision} \cdot p_{collision}$, ,

where by assuming that the messages' length (expressed as a number of consecutive t_{slot} necessary for the message transmission) is distributed according to a geometrical distribution with parameter q :

$$E[Coll]_{collision} = \frac{t_{slot}}{1-\left[(1-p)^M + Mp(1-p)\right]^{M-1}} \cdot \left[\sum_{h=1}^{\infty}\left\{h \cdot \left[\left(1-pq^h\right)^M - \left(1-pq^{h-1}\right)^M\right]\right\} - \frac{Mp(1-p)^{M-1}}{1-q}\right]$$

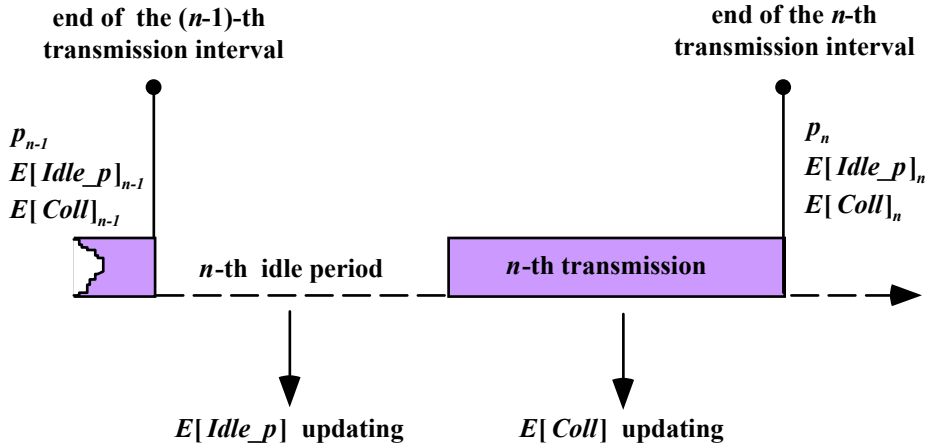


Figure 2.11: SDP updating points

The SDP protocol operates at the boundaries of the transmission intervals with the target to adjust the *p*-value so that Equation *(2.4*) holds.

Equation (2.4) provides the criteria that must be satisfied, after each transmission attempt, to approach the theoretical capacity. To achieve this, SDP updates the estimates of the network status (i.e. $E[Idle\_p]$ and $E[Coll]$) at the end of each (successful or colliding) transmission attempt. To better clarify the operations performed by a station let us refer to Figure 2.11. Specifically, the figure represents a station behavior during the *n*-th transmission interval by assuming that at the beginning of that interval, i.e. the end of the (*n*-1)th transmission interval, it has the following information:

- p_{n-1}, is the optimal value of *p*;
- $E[Idle\_p]_{n-1}$ is the average number of consecutive empty slots;
- $E[Coll]_{n-1}$ is the average collision cost.

Each station by using the carrier sensing mechanism can observe the channel status[10] and measure the length of both the last idle period and the last transmission attempt. From these two values the average idle period length and the average collision cost are approximated by exploiting a moving averaging window:

$$E[Idle\_p]_n = \alpha \cdot E[Idle\_p]_{n-1} + (1-\alpha) \cdot Idle\_p_n$$
$$E[Coll]_n = \alpha \cdot E[Coll]_{n-1} + (1-\alpha) \cdot Coll_n$$

where $E[Idle\_p]_n$ and $E[Coll]_n$ are the approximations, at the end of the *n-th* transmission attempt, of $E[Idle\_p]$ and $E[Coll]$, respectively; $Idle\_p_n$ is the length of the *n-th* idle period, $Coll_n$ is zero if the *n-th* transmission attempt is successful or it is the collision length; α $(\alpha \in [0,1])$ is a smoothing factor.

Let us assume that at the end of the *n*-th transmission interval, after computing $E[Idle\_p]_n$ and $E[Coll]_n$, Equation (1) does not hold. Hence the aim of SDP is to compute a new *p* -value, say p_{comp}, to balance in the future the idle-period length and the collision cost. As in the future, the stations will adopt the new *p*-value, by applying the Taylor formula, we express $E[Idle\_p]_{n+1}$ and $E[Coll]_{n+1}$ as follows:

$$E[Idle\_p]_{n+1} \approx \frac{1 - Mp_{comp}}{Mp_{comp}} \quad , \quad E[Coll]_{n+1} \approx l \cdot \frac{Mp_{comp}}{2} \tag{2.6}$$

where *l* is the average collision length given that two stations collide. *l* is an approximation of $E[Coll]_{|collision}$. This approximation is based on the results presented in [CCG03] indicating that, when a network operates close to its optimal behavior, it is almost negligible the probability that more than two stations collide. Therefore we can consider the approximation $E[Coll]_{|collision} = \max\{L_1, L_2\}$ where L_i is a packet size sampled from a geometric distribution with parameter *q*.

By exploiting Equation (2.6), SDP increases or decreases the *p*-value to have $E[Idle\_p]_{n+1} = E[Coll]_{n+1}$. Firstly, p_{comp} is expressed as a function of an unknown quantity *x*, such that

$$p_{comp} = p_{n-1}(1+x)$$

[10] In a CSMA protocol a station observes all the channel busy periods. A busy period is assumed to be a collision if an ACK does not immediately follow.

Then, by assuming $E[Idle\_p]_{n+1} = E[Coll]_{n+1}$, from (2.6) after some algebraic manipulations, we obtain:

$$x = -1 + \sqrt{\frac{E[Idle\_p]_n \cdot t_{slot}}{E[Coll]_n}}$$

Finally, to avoid harmful fluctuations of the *p*-value, we utilize a smoothing factor, and hence p_n is:

$$p_n = \alpha \cdot p_{n-1} + (1-\alpha) \cdot p_{comp}$$

Figure 2.12 summarizes the steps performed independently by each station at the end of each n-th transmission interval to compute the optimal *p*-value for the current network and load conditions, given the p_{n-1} value.

Begin

step 1: $Idle\_p_n$ = measure of the *n*-th the idle period;

step 2: $Coll_n$ = measure of the *n*-th collision cost;

step 3: $E[Idle\_p]_n = \alpha \cdot E[Idle\_p]_{n-1} + (1-\alpha)_n \cdot Idle\_p$

step 4: $E[Coll]_n = \alpha \cdot E[Coll]_{n-1} + (1-\alpha) \cdot Coll_n$

step 5: $p_{comp} = p_{n-1} \cdot \sqrt{\frac{E[Idle\_p]_n \cdot t_{slot}}{E[Coll]_n}}$

step 6: $p_n = \alpha \cdot p_{n-1} + (1-\alpha) \cdot p_{comp}$

End.

Figure 2.12. SDP backoff tuning algorithm

Simulation results [BCG01] indicate that the capacity of the enhanced protocol approaches the theoretical capacity limits of the IEEE 802.11 protocol in all the configurations analyzed. Studies of transient conditions show that, when the load changes, the protocol quickly re-tunes the backoff to the new optimal one. Unfortunately, to be effective the SDP algorithm assumes that each station is able to measure the average idle and collision lengths. As shown in the next section in a real 802.11 network phenomena occurring at the physical layer make this assumption difficult to verify. To cope with this problem we have designed a distributed mechanism, named *Asymptotically Optimal Backoff* (*AOB*) that dynamically adapts the backoff window size to the current load. AOB guarantees that an IEEE 802.11 WLAN asymptotically (i.e. for a large number of active stations) achieves its optimal channel utilization. The AOB mechanism adapts the backoff window to the network contention level by using two simple load estimates: the slot utilization, and the average size of transmitted frames. These estimates are simple and can be obtained with no additional costs or overheads. The AOB mechanism is presented in Section 2.5. It is worth noting that, in [BCG02] it is shown that the optimal capacity state and the optimal energy consumption state almost coincide. Hence, the AOB mechanism also guarantee a quasi-optimal behavior from the energy consumption standpoint (i.e., minimum energy consumption).

2.2.2. MAC delay

The IEEE 802.11 capacity analysis presented in the previous section is performed by assuming that the network operates in asymptotic conditions (i.e. each LAN station always has a packet ready for

transmission). However, LANs typically operate in normal traffic conditions, i.e. the network stations generate an aggregate traffic that is lower (or slightly higher) than the maximum traffic the network can support. In these load conditions, the most meaningful performance figure is the MAC delay, see [CGL97]. Two sets of MAC delay results are presented hereafter, corresponding to a traffic generated by 50 stations, made up of short (2 slots) and long (100 slots) messages, respectively. Stations alternate between idle and busy periods. In the simulative experiments, the channel utilization level is controlled by varying the idle periods' lengths.

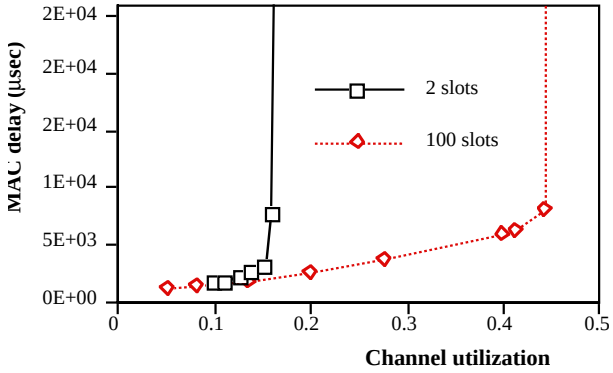


Figure 2.13. IEEE 802.11 performance: Average MAC delay

Figure 2.13 (which plots the average MAC delay vs. the channel utilization) highlights that, for light load conditions, the IEEE 802.11 exhibits very low MAC delays. However, as the offered load approaches the capacity of the protocol (see Figure 2.10), the MAC delay sharply increases. This behavior is due to the CSMA/CA protocol. Under light-load conditions the protocol introduces almost no overhead (a station can immediately transmit as soon as it has a packet ready for transmission). On the other hand, when the load increases, the collision probability increases as well, and most of the time a transmission results in a collision. Several transmissions attempts are necessary before a station is able to transmit a packet, and hence the MAC delay largely increases**.** It is worth noting that the AOB algorithm developed for optimizing the protocol capacity also contributes to avoid that MAC delays become unbounded when the channel utilization approaches the protocol capacity, see Section 2.2.1.

2.3. IEEE 802.11b Measurements

Most of 802.11 performance analyses were (and are) carried by simulation and, hence, the results observed are highly dependent on the physical layer model implemented in the simulation tool used in the analysis (e.g., GloMosim [Glo02], ns-2 [Ns02], Qualnet [Qua02]). Hereafter, we present and discuss measurements obtained from a real testbed constructed with 802.11b nodes operating in ad hoc mode (similar results related to the original 802.11 standard can be found in [ACG03]). From these results we have derived a more accurate channel model for 802.11b ad hoc networks.

To better understand the results presented below, it is useful to provide a model of the relationships existing among stations when they transmit or receive. In particular, it is useful to make a distinction between the transmission range, the interference range and the carrier sensing range. The following definitions can be given.

The *Transmission Range* (*TX_range*) is the range (with respect to the transmitting station) within which a transmitted packet can be successfully received. The transmission range is mainly determined by the transmission power and the radio propagation properties.

The *Physical Carrier Sensing Range* (*PCS_range*) is the range (with respect to the transmitting station) within which the other stations detect a transmission. It mainly depends on the sensitivity of the receiver (the receive threshold) and the radio propagation properties.

The *Interference Range* (*IF_range*) is the range within which stations in receive mode will be "interfered with" by a transmitter, and thus suffer a loss. The interference range is usually larger than the transmission range, and it is a function of the distance between the sender and receiver, and of the path loss model. It is very difficult to predict the interference range as it strongly depends on the ratio between power of the received "correct" signal and the power of the received "interfering" signal. Both these quantities heavily depend on several factors (i.e., distance, path, etc.) and hence to estimate the interference we must have a detailed snapshot of the current transmission and relative station position.

Table 2.1. IEEE 802.11b parameter values

$Slot\_Time$	τ	PHY_{hdr}	MAC_{hdr}	*Bit Rate (Mbps)*
20 µsec	≤1 µsec	192 bits (9.6 t_{slot})	272 bits	1, 2, 5.5, 11
DIFS	*SIFS*	*ACK*	CW_{MIN}	CW_{MAX}
50 µsec	10 µsec	112 bits + PHY_{hdr}	32 t_{slot}	1024 t_{slot}

In the simulation studies the following relationship has been generally assumed: $TX\_range \le IF\_range \le PCS\_range$. For example, in the ns-2 simulation tool [Ns02] the following values are used to model the characteristics of the physical layer: $TX\_range = 250m$, $IF\_range = PCS\_range = 550m$.

The numerical results presented in the next sections depend on the specific setting of the IEEE 802.11b protocol parameters. Table 2.1 gives the values for the protocol parameters used hereafter.

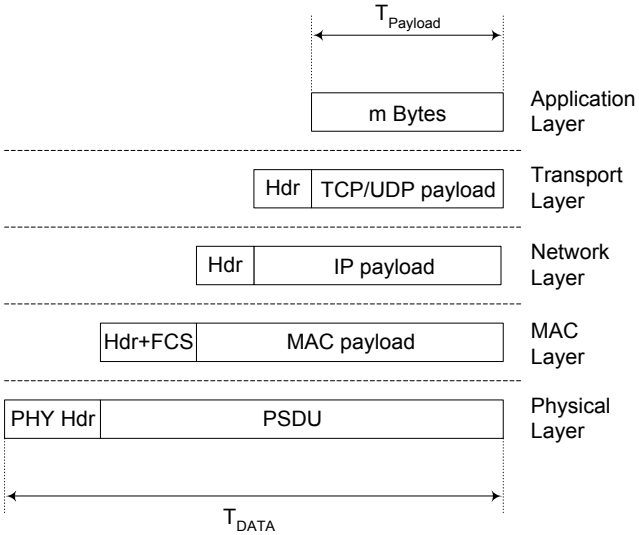


Figure 2.14: Encapsulation overheads

2.3.1. Available Bandwidth

To discuss the measurement results presented hereafter it is useful to investigate the amount of bandwidth that is available in an IEEE 802.11b network to transmit user data. In this section we will show that only a fraction of the 11 Mbps nominal bandwidth of the IEEE 802.11b cards can be used for data transmission. To this end we need to carefully analyze the overheads associated with the transmission of each packet (see Figure 2.14). Specifically, each stream of *m* bytes generated by a legacy Internet application is encapsulated in the TCP/UDP and IP protocols that add their headers before delivering the resulting IP datagram to the MAC layer for transmission over the wireless medium. Each MAC data frame is made up of: *i)* a *MAC header*, say MAC_{hdr}, containing MAC addresses and control information,[11] and *ii)* a variable length *data payload*, containing the upper layers data information. Finally, to support the physical procedures of transmission (carrier sense and reception) a *physical layer preamble* (PLCP preamble) and a *physical layer header* (PLCP header) have to be added to both data and control frames. Hereafter, we will refer to the sum of PLCP preamble and PLCP header as PHY_{hdr}.

It is worth noting that these different headers and data fields are transmitted at different data rates to ensure the interoperability between 802.11 and 802.11b cards. Specifically, the standard defines two different formats for the PLCP: Long PLCP and Short PLCP. Hereafter, we assume a Long PLCP that includes a 144-bit preamble and a 48-bit header both transmitted at 1 Mbps while the MAC_{hdr} and the $MAC_{payload}$ can be transmitted at one of the NIC data rates: 1, 2, 5.5, and 11 Mbps. In particular, control frames (RTS, CTS and ACK) can be transmitted at 1 or 2 Mbps, while data frame can be transmitted at any of the NIC data rates.

By taking into considerations the above quantities Equation *(2.7*) defines the maximum expected throughput for a single active session (i.e., only a sender-receiver couple active) when the basic access scheme (i.e., DCF and no RTS-CTS) is used. Specifically, Equation *(2.7*) is the ratio between the time required to transmit the user data and the overall time the channel is busy due to this transmission:

$$Th_{noRTS/CTS} = \frac{T_{payload}}{DIFS + T_{DATA} + SIFS + T_{ACK} + \frac{CW\min}{2} * Slot\_Time} \tag{2.7}$$

where

$T_{payload}$ is the time required to transmit only the *m* bytes generated by the application; $T_{payload}$ is therefore equal to $m / data\_rate$, where $data\_rate$ is the data rate used by the NIC to transmit data, i.e., 1, 2, 5.5, or 11 Mbps.

T_{DATA} is the time required to transmit a MAC data frame; this includes the PHY_{hdr}, MAC_{hdr}, $MAC_{payload}$ and FCS bits for error detection.

T_{ACK} is the time required to transmit a MAC ACK frame; this includes the PHY_{hdr}, and MAC_{hdr}.

$\frac{CW\min}{2} * Slot\_Time$ is the average back off time

[11] Without any loss of generality we have considered the *frame error sequence* (FCS), for error detection, as belonging to the MAC header.

When the RTS/CTS mechanism is used, the overheads associated with the transmission of the RTS and CTS frames must be added to the denominator of *(2.7)*. Hence, in this case, the maximum throughput $Th_{RTS/CTS}$, is defined as

$$Th_{RTS/CTS} = \frac{T_{payload}}{DIFS + T_{RTS} + T_{CTS} + T_{DATA} + T_{ACK} + 3*SIFS + \frac{CW\ \min}{2} * Slot\_Time} \quad (2.8)$$

where T_{RTS} and T_{CTS} indicate the time required to transmit the RTS and CTS frames, respectively. Equations *(2.7)* and *(2.8)* are used to obtain the theoretical throughput for a single session with UDP traffic. Indeed, when using the TCP protocol, overheads due to the TCP_ACK transmission must be considered. More precisely, the technique of cumulative ACK answering to two consecutive TCP_DATA is used, so a TCP handshake is composed by TCP_DATA1, TCP_DATA2 and TCP_ACK.

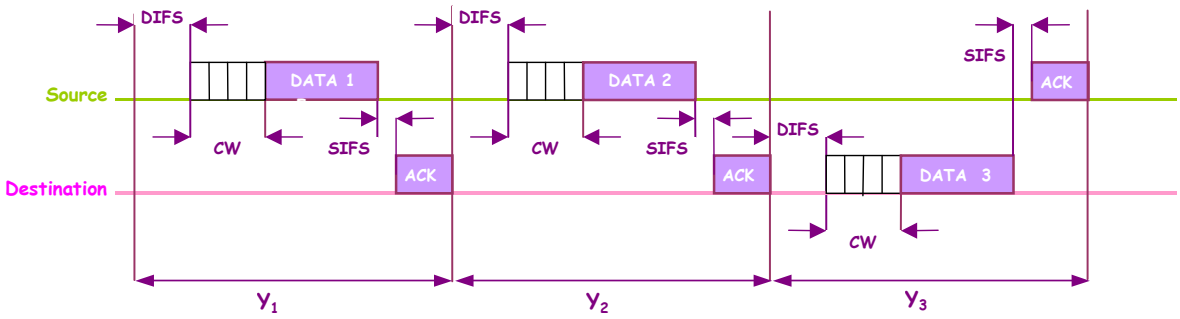


Figure 2.15: TCP handshake with basic mechanism

Figure 2.15 and Figure 2.16 show the TCP handshake on the channel; in particular, DATA1 and DATA2 packets are obtained by the encapsulation of TCP_DATA1 and TCP_DATA2, instead DATA3 is obtained by the encapsulation of the TCP_ACK.

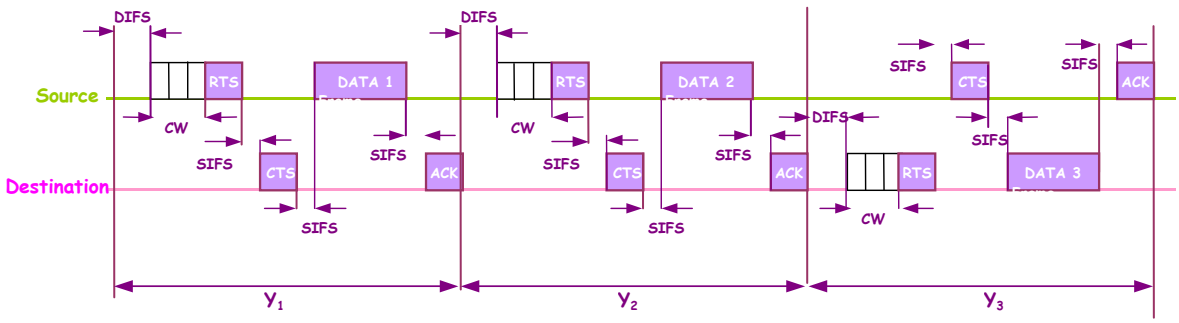


Figure 2.16: TCP handshake with RTS/CTS mechanism

Thus, the theoretical throughput is the ratio between the time to transmit two user data and the overall time for the complete transmission on the channel:

$$Th = \frac{2 * T_{payload}}{y_1 + y_2 + y_3} \tag{2.9}$$

where $T_{payload}$ is equal to $m / data\_rate$ and y_i represents the time required to the transmission of the DATAi packet on the channel.

In Table 2.2 and Table 2.3 we report the expected throughputs (with and without the RTS/CTS mechanism) by assuming that the NIC is transmitting at a constant data rate equal to 1, 2, 5.5, or 11 Mbps, respectively for a UDP and TCP connection. These results are computed by applying Equations *(2.7)*, *(2.8)* and *(2.9)*, and assuming a data packet size at the application level equal to *m*=512 and *m*=1024 bytes.

Table 2.2. Maximum UDP throughput at different data rate.

	m= 512 Bytes		m=1024 Bytes	
	No RTS/CTS	RTS/CTS	No RTS/CTS	RTS/CTS
11 Mbps	3.337 Mbps	2.739 Mbps	5.120 Mbps	4.386 Mbps
5.5 Mbps	2.490 Mbps	2.141 Mbps	3.428 Mbps	3.082 Mbps
2 Mbps	1.319 Mbps	1.214 Mbps	1.589 Mbps	1.511 Mbps
1 Mbps	0.758 Mbps	0.738 Mbps	0.862 Mbps	0.839 Mbps

Table 2.3. Maximum throughputs in Mbit/sec (Mbps) at different data rates for a TCP connection

	m= 512 Bytes		m=1024 Bytes	
	No RTS/CTS	RTS/CTS	No RTS/CTS	RTS/CTS
11 Mbps	2.456 Mbps	1.979 Mbps	4.015 Mbps	3.354 Mbps
5.5 Mbps	1.931 Mbps	1.623 Mbps	2.858 Mbps	2.507 Mbps
2 Mbps	1.105 Mbps	0.997 Mbps	1.423 Mbps	1.330 Mbps
1 Mbps	0.661 Mbps	0.620 Mbps	0.796 Mbps	0.766 Mbps

As shown in Table 2.2, only a small percentage of the 11 Mbps nominal bandwidth can be really used for data transmission. This percentage increases with the payload size. However, even with large packets sizes (e.g., m=1024 bytes) the bandwidth utilization is lower than 44%.

The above theoretical analysis has been complemented with the measurements of the actual throughput at the application level. Specifically, we have considered two types of applications: ftp and CBR. In the former case the TCP protocol is used at the transport layer, while in the latter case the UDP is adopted. In both cases the applications operate in asymptotic conditions (i.e., they always have packets ready for transmission) with constant size packets of 512 bytes.

The results obtained from this experimental analysis are reported in the Figure 2.17.

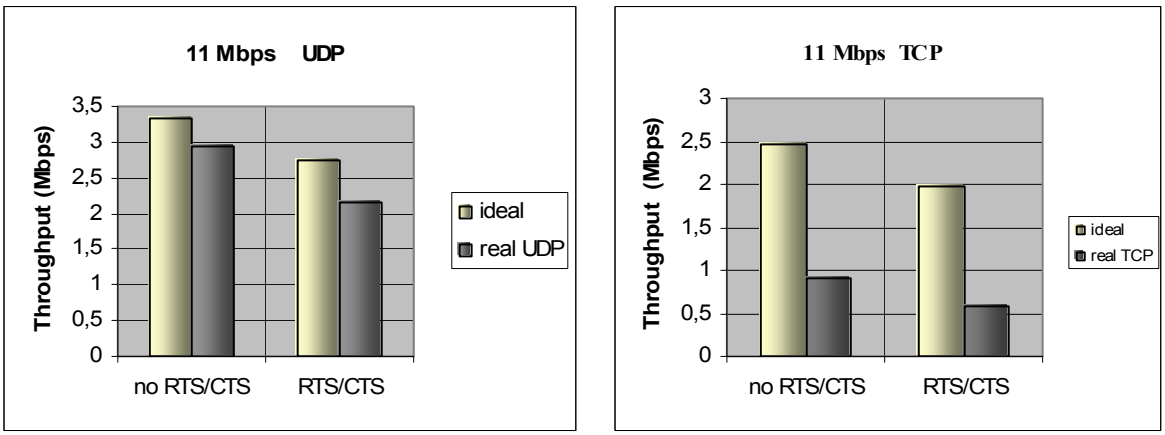


Figure 2.17: Comparison between the theoretical maximum throughput and the actual throughput achieved by TCP/UDP applications

The experimental results related to the UDP traffic are very close to the maximum throughput computed analytically. As expected, in the presence of TCP traffic the measured throughput is much lower than the theoretical maximum throughput. Similar results have been also obtained when the NIC data rate is set to 1, 2 or 5.5 Mbps.

2.3.2. Transmission Ranges

The dependency between the data rate and the transmission range was investigated by measuring the packet loss rate experienced by two communicating stations whose network interfaces transmit at a constant (preset) data rate. Specifically, four sets of measurements were performed corresponding to the different data rates: 1, 2, 5.5, and 11 Mbps. In each set of experiments the packet loss rate was recorded as a function of the distance between the communicating stations. The resulting curves are presented in Figure 2.18. Figure 2.19 shows the transmission-range curves derived in two different days (the data rate is equal to 1 Mbps). This graph highlights the variability of the transmission range depending on the weather conditions.

The results presented in Figure 2.18 are summarized in Table 2.4 where the estimates of the transmission ranges at different data rates are reported. These estimates point out that, when using the highest bit rate for the data transmission, there is a significant difference in the transmission range of control and data frames, respectively. For example, assuming that the RTS/CTS mechanism is active, if a station transmits a frame at 11 Mbps to another station within its transmission range (i.e., less then 30 m apart) it reserves the channel for a radius of approximately 90 (120) m around itself. The RTS frame is transmitted at 2 Mbps (or 1 Mbps), and hence, it is correctly received by all stations within station transmitting range, i.e., 90 (120) meters.

Again, it is interesting to compare the transmission range used in the most popular simulation tools, like ns-2 and Glomosim, with the transmission ranges measured in our experiments. In these simulation tools it is assumed $TX\_range = 250m$. Since the above simulation tools only consider

a 2-Mbps bit rate we make reference to the transmission range estimated with a NIC data rate of 2 Mbps. As it clearly appears, the value used in the simulation tools (and, hence, in the simulation studies based on them) is 2-3 times higher that the values measured in practice. This difference is very important for example when studying the behavior of routing protocols: the shorter is the *TX_range*, the higher is the frequency of route re-calculation when the network stations are mobile.

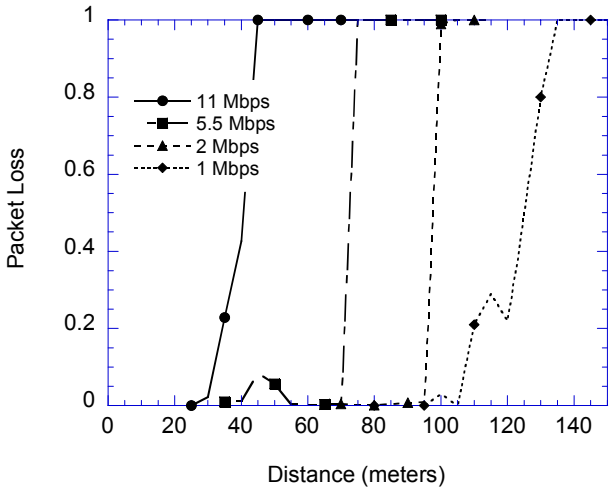


Figure 2.18: Packet loss rate as a function of the distance between communicating stations for different data rates

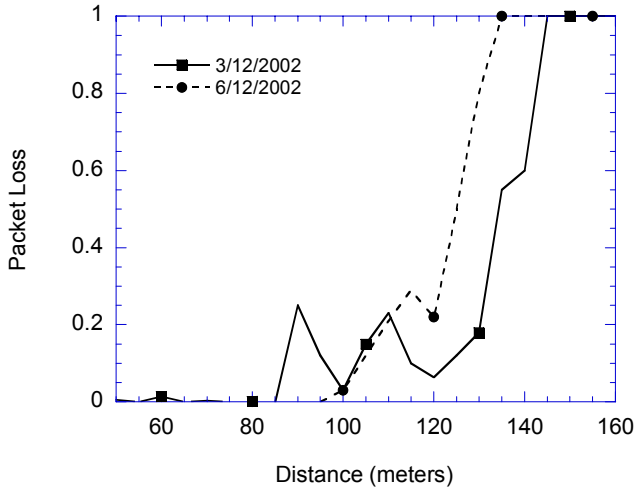


Figure 2.19: 1 Mbps transmission ranges in different days

Table 2.4. Estimates of the transmission ranges at different data rates.

	11 Mbps	5.5 Mbps	2 Mbps	1 Mbps
Data *TX_range*	30 meters	70 meters	90-100 meters	110-130 meters
Control *TX_range*			≈ 90 meters	≈ 120 meters

2.3.3. Transmission Ranges and the Mobile devices' Height

During the experiments we performed to analyze the transmission ranges at various data rates, we observed a dependence of the transmission ranges from the mobile devices' height from the ground. Specifically, in some case we observed that while the devices were not able to communicate when located on the stools, they started to exchange packets by lifting them up. In this section we present the results obtained by a careful investigation of this phenomenon. Specifically, we studied the dependency of the transmission ranges from the devices height from the ground. To this end we measured the throughput between two stations[12] as a function of their height from the ground: four different heights were considered: 0.40 m, 0.80 m, 1.2 m and 1.6 m. The experiments were performed with the Wi-Fi card set at two different transmission rates: 2 and 11 Mbps. In each set of experiments the distance among the two devices was set close to guarantee that the receiver is always inside the sender transmission range. Specifically, the sender-receiver distance was equal to 30 and 70 m when the cards operated at 11 and 2 Mbps, respectively.

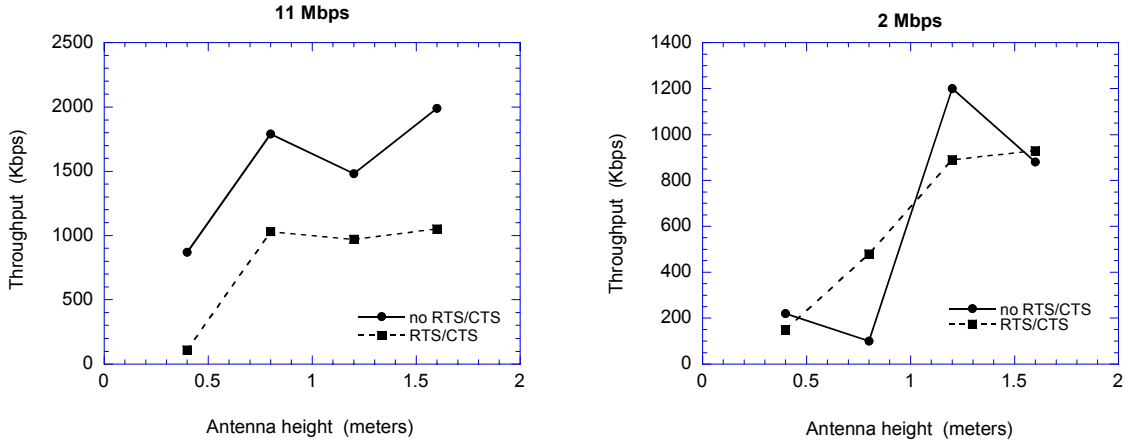


Figure 2.20: Relationship between throughput and devices' height

As it clearly appears in Figure 2.20, the ground height may have a big impact on the quality of the communications between the mobile devices. For example, at 11 Mbps, by lifting up the devices from 0.40 meters to 0.80 meters the throughput doubles, while further increasing the height does not produce significant throughput gains. A similar behavior is observed with a 2 Mbps transmission rate, however in this case the major throughput gain is obtained lifting up the devices from 0.80 meters to 1.20 meters. A possible explanation of this difference is related to the distances, in the two cases, between the communicating devices. This intuition is confirmed by the work presented in [OG02] that provides a theoretical framework to explain the height impact on IEEE 802.11 channel quality. Specifically, the channel power loss depends on the contact between the Fresnel zone and the ground. The Fresnel zone for a radio beam is an elliptical area with foci

[12] In these experiments UDP is used as the transport protocol.

located in the sender and the receiver. Objects in the Fresnel zone cause diffraction and hence reduce the signal energy. In particular, most of the radio-wave energy is within the First Fresnel Zone, i.e., the inner 60% of the Fresnel zone. Hence, if this inner part contacts the ground (or other objects) the energy loss is significant. Figure 2.21 shows the Fresnel zone (and its inner 60%) for a sender-receiver couple at a distance D. In the figure, R1 denotes the height of the First Fresnel Zone. As shown in [OG02] R1 is highly dependent on the stations distance. For example, when the sender and the receiver are at an height of 1 meter from the ground, the First Fresnel Zone has a contact with the ground only if $D > 33$ meters. While at heights of 1.5 and 2 meters the First Fresnel Zone contacts the ground only if D is greater than 73 and 131 meters, respectively. These theoretical computations are aligned with our experimental results.

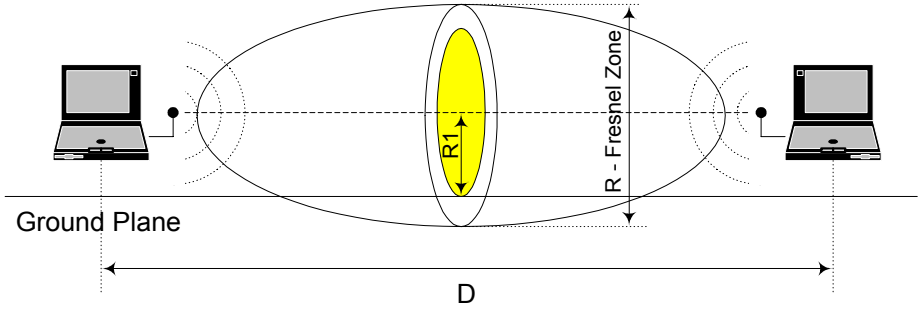


Figure 2.21: The Fresnel Zone

2.3.4. Four-Stations Network Configurations

The results presented in the previous sections show that the IEEE 802.11b behavior is more complex than the behavior of the IEEE 802.11 standard. Indeed the availability of different transmission rates may cause the presence of several transmission ranges inside the network. In particular, inside the same data transfer session there may be different transmission ranges for data and control frame (e.g., RTS, CTS, ACK). Hereafter, we show that the superposition of these different phenomena makes very difficult to understand the behavior of IEEE 802.11b ad hoc networks. To reduce this complexity, in the experiments presented below the NIC data rate is set to a constant value equal to 11 Mbps for the entire duration of the experiment.[13]

The network configuration is shown in Figure 2.22, and the related results are presented in Figure 2.23.

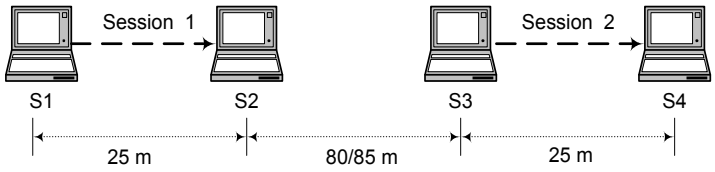


Figure 2.22: Network configuration at 11 Mbps

[13] It is worth pointing out that we experienced a high variability in the channel conditions thus making a comparison between the results difficult.

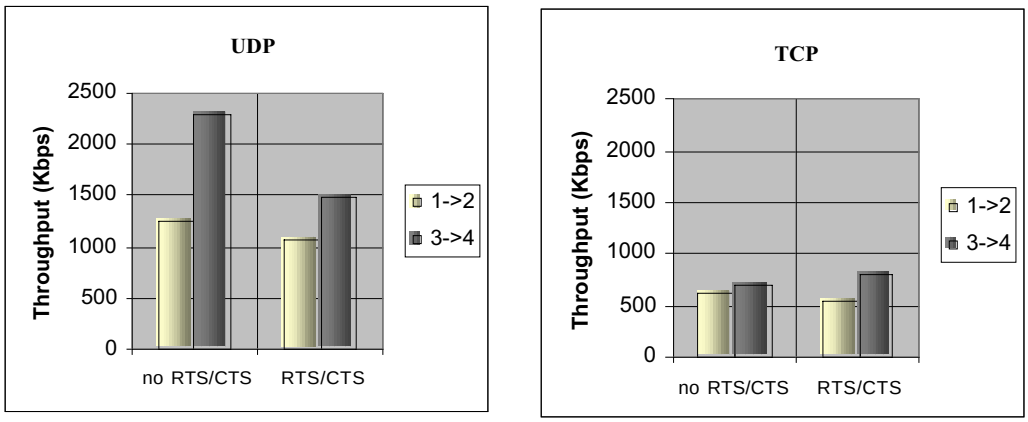


Figure 2.23: Throughputs at 11 Mbps

The results show that dependencies exist between the two connections even though the transmission range is smaller than the distance between stations S1 and S3. In detail, the throughput experienced by each session is much smaller than the throughput obtained by a session in isolation, e.g., about 3.3 Mbps with UDP (see Figure 2.17).

Furthermore, the dependency exists also when the basic mechanism (i.e., no RTS/CTS) is adopted.[14] To summarize, these experiments show that i) interdependencies among the stations extends beyond the transmission range; ii) our hypothesis is that the physical carrier sensing range, including all the four stations, produces a correlation between active connections and its effect is similar to that achieved with the RTS/CTS mechanism (virtual carrier sensing). The difference in the throughputs achieved by the two sessions when using the UDP protocol (with or without RTS/CTS) can be explained by considering the asymmetric condition that exists on the channel: station S2 is exposed to transmissions of station S3 and, hence, when station S1 sends a frame to S2 this station is not able to send back the MAC ACK. Therefore, S1 reacts as in the collision cases (thus re-scheduling the transmission with a larger backoff). It is worth pointing out that also S3 is exposed to S2 transmissions but the S2's effect on S3 is less marked given the different role of the two stations. When using the basic access mechanism, the S2's effect on S3 is limited to short intervals (i.e., the transmission of ACK frames). When adopting the RTS/CTS mechanism, the S2 CTS forces S3 to defer the transmission of RTS frames (i.e., simply a delay in the transmission), while RTS frames sent by S3 forces S2 to not reply with a CTS frame to S1's RTS. In the latter case, S1 increases the back off and reschedules the transmission. Finally, when the TCP protocol is used the differences between the throughput achieved by the two connections still exist but are reduced. The analysis of this case is very complex because we must also take into consideration the impact of the TCP mechanisms that: i) reduces the transmission rate of the first connection, and ii) introduces the transmission of TCP-ACK frames (from S2 and S4) thus contributing to make the system less asymmetric.

2.3.5. Physical Carrier Sensing Range

Results presented in the previous section seem to indicate that dependencies among the stations extend far beyond the transmission range. For example, taking as a reference the scenario presented in Figure 2.22, the distance between the two couples of transmitting stations is about three times the transmission range. The hypothesis is that dependencies are due to a large physical carrier sensing that includes all the stations. To validate this hypothesis and to better understand the system behavior we designed some experiments to estimate the physical carrier sensing range.

[14] A similar behavior is observed (but with different values) by adopting the RTS/CTS mechanism.

A direct measure of this quantity seems difficult to achieve because the 802.11b cards we utilized do not provide to the higher layers information about the channel carrier sensing. Therefore, we defined an indirect way to perform these measurements. We utilized the scenario shown in Figure 2.24 with fixed distance between each couple of communicating stations (d(1,2)=d(3,4)=10 meters), and variable distance between the two couples, i.e., d(2,3), is variable.

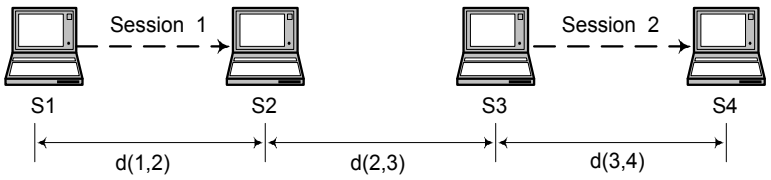


Figure 2.24: Reference network scenario

The idea is to investigate the correlation among the two sessions while increasing the distance d(2,3). To measure the correlation degree, just before running each experiment we performed some preliminary measurements. Specifically, we measured the throughput of each session in isolation, i.e., when the other session is not active. Then, we measured the throughput of each session when both sessions are active. Hereafter, $Th_i(x)$ denotes the throughput of session i (i=1,2) when both sessions are active and d(2,3)=x. Obviously, $Th_i(\infty)$ denotes the throughput of session i (i=1,2), when d(2,3)= ∞, and hence the two sessions are independent. By exploiting these measurements we estimated the correlation existing between the two sessions by the following index:

$$D_1(x) = 1 - \frac{Th_1(x) + Th_2(x)}{Th_1(\infty) + Th_2(\infty)} \quad .$$

The $D_1(x)$ index takes the value 0 if the two sessions are independent. Taken a session as a reference, the presence of the other session may have two possible effects on the performance of the reference session: 1) if the two sessions are within the same physical carrier sensing range, they share the same physical channel; 2) if they are outside the physical carrier sensing range the radiated energy from one session may still affect the quality of the channel observed by the other session. As the radiated energy may travel over unlimited distances, we can expect that $D_1(x)$ may be equal to zero only for very large distances among the sessions [E02].

When the $D_1(x)$ value is greater than zero, the index does not indicate how strong the correlation is. To measure this second aspect we introduce the $D_2(x)$ index:

$$D_2(x) = \frac{Th_1(0) + Th_2(0)}{Th_1(x) + Th_2(x)} \quad .$$

$D_2(x)$ compares the throughput of the two sessions when they are active at the same time and d(2,3)=x, with respect to the two-session throughput when all the stations are inside the same transmission range, i.e., d(2,3)=0. A $D_2(x)$ value equal to 1 indicates the maximum correlation that exists when all stations are in the same transmission range.

By varying the distance d(2,3) we performed several experiments to estimate the above indexes. The results were obtained with the cards transmission rates set to 11 Mbps, and are summarized in Table 2.5. As it clearly appears from the table, the correlation among session is still marked when d(2,3) is less than or equal to 250 meters, noticeably decreases around 300 meters, and further reduces (but not disappears) when the inter-session distance is about 350 meters.

Table 2.5: Throughput values (Card rate =11 Mbps, payload size=512 Bytes)

Access Mechanism	Distance	Throughput of Session 1		Throughput of Session 2		$D_1(x)$	$D_2(x)$
		$Th_1(\infty)$ Kbps	$Th_1(x)$ Kbps	$Th_2(\infty)$ Kbps	$Th_2(x)$ Kbps		
No RTS/CTS	x=0	2780	1849	2981	1768	0.37	1.00
	x=150	1950	1500	2950	2250	0.23	0.96
	x=180	2920	2210	3040	1580	0.36	0.95
	x=200	2290	1930	3160	2660	0.16	0.78
	x=250	2820	1700	3170	2760	0.25	0.81
	x=300	2980	2800	3060	2750	0.08	0.65
	x=350	2730	2590	3250	3230	0.03	0.62

To estimate the size of the physical carrier sensing range we have analytically computed the throughput of the two sessions when they are both within the same physical carrier sensing range.

Since in this scenario the probability that both the session start transmitting on the same slot is very small (i.e., the collision probability is almost negligible) to simplify the analysis we assumed that stations always use a 32-slot contention window (i.e., the smallest congestion window). Under this assumption, the average backoff time between two consecutive transmission attempts is equal to $Cw_{min}/4$. Taken into consideration the overheads involved in a frame transmission (see Section 2.3.1 for details) the maximum aggregate throughput of the two sessions is equal to 3.66 Mbps and 5.45 Mbps when the frame payload is equal to 512 and 1024 bytes, respectively.

By observing from Table 2.5 that the aggregate throughput for the two sessions is about 3.7 Mbps when the d(2,3) is less than 200 meters, we assume that 200 m is approximately the size of the physical carrier sensing range. After this distance the correlation among the two sessions is due to the mutual impact of the two sessions on the channel quality. A set of measurements is currently ongoing to further verify the exact size of the physical carrier sensing range.

The results obtained confirm the hypotheses we made in the previous section to justify the apparent dependencies existing among the two couples of transmitting stations even if the distance among them is about three times greater than the transmission range.

It is worth noting that the ideal value for $D_1(0)$ is 0.5, i.e., each session gets half of the throughput of the session in isolation. This is not true for CSMA MAC protocol as $Th_1(0)$ ($Th_2(0)$) is greater than $Th_1(\infty)/2$ ($Th_2(\infty)/2$). This results is caused by a smaller overhead of the backoff algorithm in the experiments with d(2,3)=0.

2.4. Channel Model for an IEEE 802.11b Ad Hoc Network

The results presented in the previous section indicate that for correctly understanding the behavior of an 802.11b network operating in ad hoc mode, several different ranges must be considered.

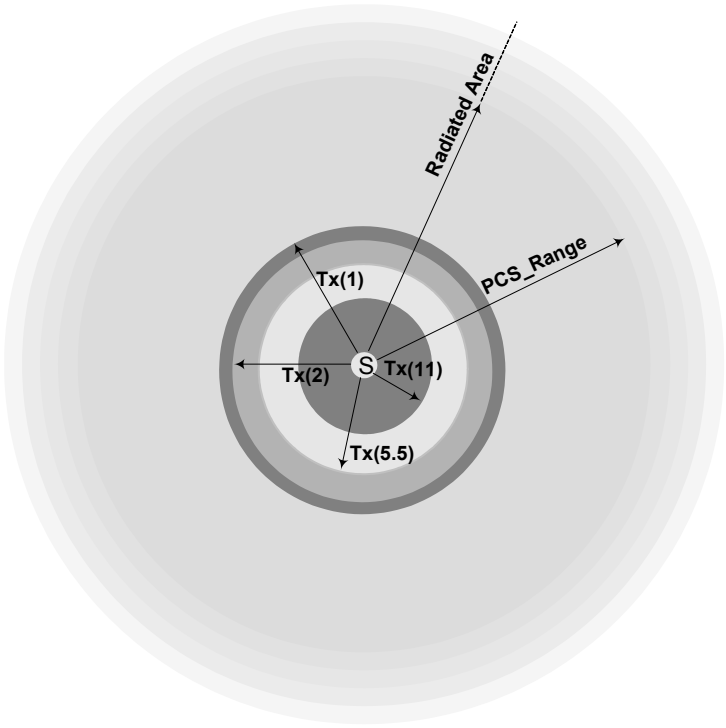


Figure 2.25: Channel model for an 802.11 ad hoc network

Specifically, as shown in Figure 2.25, given a transmitting station S, the stations around will be affected by the station S transmissions in a different way depending on the distance from S and the rate used by S for its transmissions.

Specifically, assuming that S is transmitting with a rate x $\left(x \in \{1,\ 2,\ 5.5,\ 11\}\right)$ stations around it can be partitioned into three classes depending on their distance, d, from S:

i. Stations at a distance d < TX_Range(x) are able to correctly receive data from S, if S is transmitting at a rate lower or equal to x;

ii. Stations at a distance d, where TX_Range(x) < d < PCS_Range, are not able to receive data correctly from station S. However, as they are in the S physical carrier sensing range, when S is transmitting they observe the channel busy and thus they defer their transmissions.

iii. Stations at a distance d > PCS_Range do not measure any significant energy on the channel when S is transmitting, therefore they can start transmitting contemporarily to S; however, the quality of the channel they observe may be affected by the energy radiated by S. In addition, if d < PCS_Range + TX_Range(x) some interference phenomena may occur (see below). This interference depends on the IF_Range value. This value is difficult to model and evaluate as it depends on several factors (mainly the power at the receiving site) but as explained before TX_Range(1) < IF_Range < PCS_Range.

Several interesting observations can be derived by taking into consideration points i-iii above. Firstly, the hidden station phenomenon, as it is usually defined in the literature (see Section 2.1.2), is almost impossible with the ranges measured in our experiments. Indeed, the PCS_Range is more than twice TX_Range(1), i.e., the larger transmission range. Furthermore, two stations, say S1 and S2, that can start transmitting towards the same receiver, R, must be at a distance ≤ 2•TX_Range(1), and thus they are inside the physical carrier sensing range of each other. Hence, if S1 has an ongoing transmission with R, S2 will observe a busy channel and thus will defer its own transmission. This means that, in this scenario, virtual carrier sensing is not necessary and the RTS/CTS mechanism only introduces additional overhead.

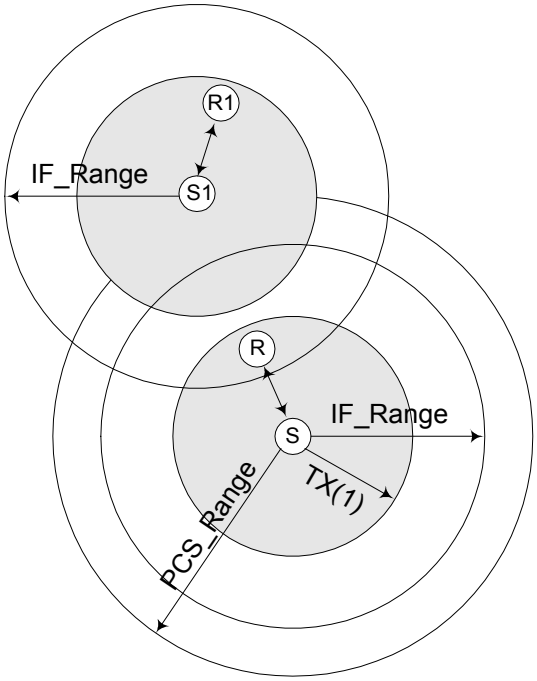


Figure 2.26: Interference-based hidden station phenomenon

While the hidden station phenomenon, as defined in the literature, seems not relevant for this environment point iii above highlights that packets cannot be correctly received due to the interference caused by a station that is "hidden" to the sending station. An example of this type of *hidden station phenomenon* is presented in Figure 2.26. In this figure we have two transmitting stations, S and S1 that are outside their respectively PCS_Range and hence they are hidden to each other. In addition we assume that the receiver of station S (denoted by R in the figure) is inside the interference range (IF_Range) of station S1. In this scenario S and S1 can be simultaneously transmitting and, if this occurs, station R cannot receive data from S correctly. Also in this case the RTS/CTS mechanism does not provide any help and new coordination mechanisms need to be designed to extend the coordination in the channel access beyond the PCS_Range.

It is worth noting that, in our channel model, the exposed station definition (see Figure 2.7) must be modified too. In this scenario, exposed stations are those station at a distance PCS_Range-TX_Range(1) < d < PCS_Range. Indeed, these stations are exposed to station S transmissions, while they are in the transmission range of stations with d > PCS_Range .

The following example outlines problems that may occur in this case. Let us denote with S1 a station at a distance d from S: PCS_Range < d < PCS_Range+TX_Range(x). Station S1 can start transmitting, with a rate x, towards a station E that is inside the physical carrier sensing of S; station E cannot reply because it observes a busy channel due to the ongoing station S transmissions, i.e., E is exposed to station S. Since station S1 does not receive any reply (802.11 ACK) from E, it assumes an error condition (collision or CRC error condition), hence it backoffs and then tries again. If this situation repeats for several times (up to 7), S1 assumes that E is not anymore in its transmission range, gives up the transmission attempt and (wrongly) signals to the higher layer a link breakage condition, thus forcing higher layers to attempt a recovery action (e.g., new route discovery, etc.).

To summarize, results obtained in the configuration we analyzed indicate that the hidden station and exposed station definitions must be extended. These new hidden-station and exposed-station phenomena may produce undesirable effects that may degrade the performance of an ad hoc network, mainly if the TCP protocol is used. Extending the coordination in the channel access beyond the PCS_Range seems to be the correct direction for solving the above problems.

2.5. Burtsy MAC definition

As it is pointed out in the previous section, the physical carrier sensing range is very large (compared to the transmission range) and, in average, it contains most of the stations around a transmitting one. Therefore a back off tuning algorithm must be effective inside the physical carrier sensing range, and it must use a very simple feedback from the channel. Stations inside the physical carrier sensing range may be not able to measure the idle and collision lengths. To cope with this problem the algorithm presented in this section utilizes a very simple estimate of the channel conditions which can be measured by all stations inside the physical carrier sensing range of the transmitting one. Specifically, our mechanism, named Asymptotically Optimal Backoff (AOB), dynamically adapts the backoff window size to the current network contention level, and guarantees that an IEEE 802.11 WLAN asymptotically achieves its optimal channel utilization. The AOB mechanism measures the network contention level by using two simple estimates: the slot utilization, and the average size of transmitted frames. These estimates are simple and can be obtained by exploiting information that is already available in the standard protocol. AOB can be used to extend the standard 802.11 access mechanism without requiring any additional hardware.

To better understand the basic ideas of the AOB mechanism it is useful to observe the behavior of a standard 802.11 system running in DCF mode, with respect to the contention level, i.e. the number of active stations with continuous transmission requirements. By analyzing the behavior of the 802.11 DCF mechanisms some problems could be identified. Specifically, the results presented in [BCG03a] show that the channel utilization is negatively affected by the increased contention level. These results can be explained since, in the IEEE 802.11 backoff algorithm, a station selects the initial size of the Contention Window by assuming a low level of congestion in the system. This choice avoids long access delays when the load is light. Unfortunately, this choice causes efficiency problems in bursty arrival scenarios, and in congested systems, because it concentrates the accesses in a reduced time window, and hence may cause a high collision probability. In high-congestion conditions each station reacts to the contention taking into consideration only the number of collisions already experienced while transmitting the current frame. Every station performs its attempts blindly, with a late collision reaction performed (increasing *CW_Size)*. Each increase of the *CW_Size* is obtained at the cost of a collision. It is worth noting that, as a collision detection mechanism is not implemented in the IEEE 802.11, a collision implies that the channel is not available for the time required to transmit the longest colliding packet. Furthermore, after a successful transmission the *CW_Size* is set again to the minimum value without maintaining any knowledge of the current contention level. To summarize, the IEEE 802.11 backoff mechanism has two main drawbacks: *i*) the increase in the *CW_Size* is obtained at the cost of a collision, and *ii*) after a successful transmission no memory of the actual contention level is maintained.

The drawbacks of the IEEE 802.11 backoff algorithm indicate a direction for improving the performance of a random access scheme, by exploiting the information on the current network congestion level that is already available at the MAC level. Specifically, the utilization rate of the slots (*Slot Utilization*) observed on the channel by each station is used as a simple and effective estimate of the channel congestion level. The estimated Slot Utilization must be frequently updated. For this reason in [BCD00] it was proposed that an estimate be updated by each station in every *Backoff interval*, i.e., the defer phase that precedes a transmission attempt.

A simple and intuitive definition of the slot utilization (*S_U*) is then given by:

$$S\_U = \frac{Num\_Busy\_Slots}{Num\_Available\_Slots} \quad ,$$

where *Num_Busy_Slots* is the number of slots, in the backoff interval, in which a transmission attempt starts, hereafter referred to as *busy slots*. A transmission attempt can be either a successful transmission or a collision; and *Num_Available_Slots* is the total number of slots available for transmission in the backoff interval, i.e. the sum of idle and busy slots.

In the standard 802.11 every station performs a Carrier Sensing activity and thus the proposed *S_U* estimate is simple to obtain. The information required to estimate *S_U* is already available to an IEEE 802.11 station and no additional hardware is required.

The current *S_U* estimate can be used by each station (before trying a "blind" transmission) to evaluate the opportunity to either perform or defer its scheduled transmission attempt. In other words, if a station knows that the probability of a successful transmission is low, it should defer its transmission attempt. This can be achieved in an IEEE 802.11 network by exploiting the DCC mechanism proposed in [Bon00]. According to DCC, each IEEE 802.11 station performs an additional control (beyond carrier sensing and backoff algorithm) before any transmission attempt. This control is based on a new parameter named *Probability of Transmission P_T(...)* whose value depends on the current contention level of the channel, i.e., *S_U*. The heuristic formula proposed in [Bon00] for *P_T*(...) is:

$$P\_T(S\_U,\ N\_A) = 1 - S\_U^{N\_A}$$

where, by definition, S_U assumes values in the interval [0,1], and *N_A* is the number of attempts already performed by the station for the transmission of the current frame.

The *N_A* parameter is used to partition the set of active stations in such a way that each station's subset is associated with a different level of privilege to access the channel. Stations that have performed several unsuccessful attempts have the highest transmission privilege [Bon00].

The *P_T* parameter allows filtering the transmission attempts. When, according to the standard protocol, a station is authorized to transmit (backoff counter is equal to zero and channel is idle) in the protocol extended with the Probability of Transmission a station will perform a real transmission with probability *P_T*, otherwise (i.e. with probability 1-*P_T*) the transmission is re-scheduled, since a collision would have occurred, i.e. a new backoff interval is sampled.

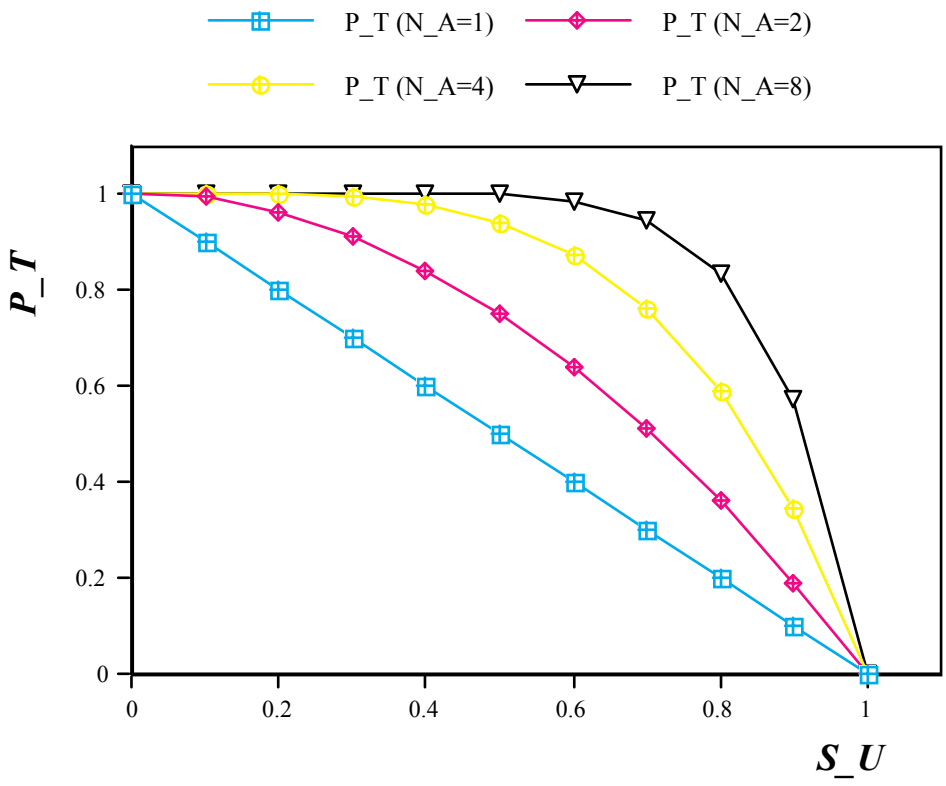


Figure 2.27: DCC Probability of Transmission

To better understand the relationship between the *P_T* definition and the network congestion level, we can observe Figure 2.27. In Figure 2.27 we show the *P_T* curves (for users with different *N_A*), with respect to the estimated *S_U* values. Assuming *S_U* close to zero, we can observe that each station, independently of its number of performed attempts, obtains a Probability of Transmission (*P_T*) close to 1. This means that the proposed mechanism has no effect on the system, and each user performs its accesses as in the standard access scheme, without any additional contention control. This point is significant, as it implies the absence of overhead introduced in low-load conditions. The differences in the users' behavior as a function of their levels of privilege (related to the value of the *N_A* parameter) appear when the slot utilization grows. For example, assuming slot utilization close to 1, say 0.8, we observe that the stations with the highest *N_A* value obtain a Probability of Transmission close to 1 while stations during the first transmission attempt transmit with a probability equal to 0.2.

It is worth noting a property of the DCC mechanism: the slot utilization of the channel never reaches the value 1. Assuming *S_U* close to or equal to 1, the DCC mechanism reduces the Probabilities of Transmission for all stations close to zero thus reducing the network contention level. This effect was due to the *P_T* definition, and in particular to the explicit presence of the upper bound 1 for the slot utilization estimate. The DCC choice to use 1 as the asymptotic limit for the *S_U* is heuristic and does not guarantee the maximum channel utilization. To achieve the maximum channel utilization we need to know the optimal congestion level, i.e. the optimal upper bound for the *S_U* value *(opt_S_U)*. It is worth noting that, if *opt_S_U* is known, the *P_T* mechanism can be easily tuned to guarantee that maximum channel utilization is achieved. Intuitively, if the slot-utilization boundary value (i.e. the value one for DCC) is replaced by the *opt_S_U* value, we reduce all the probabilities of transmission to zero in correspondence with slot utilization values greater than or equal to the *opt_S_U*. This can be achieved by generalizing the definition for the Probability of Transmission:

$$P\_T(opt\_S\_U, S\_U, N\_A) = 1 - \min\left(1, \frac{S\_U}{opt\_S\_U}\right)^{N\_A} \tag{2.10}$$

Specifically, by applying this definition of the transmission probability we obtain the *P_T* curves shown in Figure 2.28. These curves were obtained by applying the generalized *P_T* definition with *opt_S_U*=0.80. As expected, the curves indicate the effectiveness of the generalized *P_T* definition to limit *S_U* to the *opt_S_U* value.

The generalized Probability of Transmission provides an effective tool for controlling the congestion inside an IEEE 802.11 WLAN in an optimal way, provided that the *opt_S_U* value is known. In the following we will present a simple mechanism to set the *opt_S_U* value. Our mechanism is named Asymptotically Optimal Backoff as it guarantees that the optimal utilization is asymptotically achieved, i.e. for large *M* values.

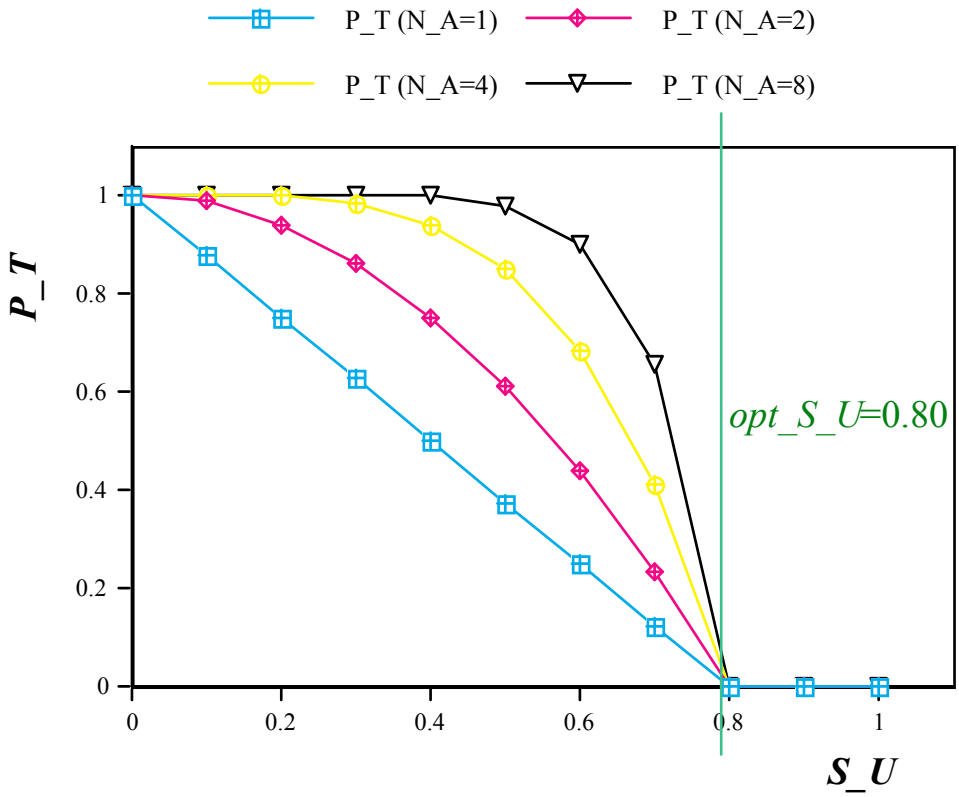


Figure 2.28: Generalized Probability of Transmission

2.5.1. Asymptotically Optimal Backoff (AOB) Mechanism

The aim of the AOB mechanism is to dynamically tune the backoff window size to achieve the theoretical capacity limit of the IEEE 802.11 protocol. The AOB mechanism is simpler, more robust and has lower costs and overhead introduced than the contention mechanisms proposed in ([CCG00] and [CCG00a]). Specifically, the AOB mechanism requires no estimate of the number M of active stations. An accurate M estimate may be very difficult to obtain because M may be highly variable in WLANs.

As explained in Section 2.2.1, the IEEE 802.11 theoretical capacity is identified by $p_{\min}$. Hereafter, we will show the relationship between $p_{\min}$ and the *opt_S_U* value of the AOB mechanism and we show that i) the product $Mp_{\min}$ is an invariant figure to characterize the theoretical capacity limits; ii) $Mp_{\min}$ provides an upper bound to *opt_S_U*.

To this end let start observing results presented in Table 2.6 that are numerically derived by computing the optimal p value, i.e. $p_{\min}$ according to formulas presented in [CCG00] Specifically, in this table we report for various network and traffic configurations (defined by the (M, q) couples) the value $Mp_{\min}$. It is worth noting that, given a q-value (or correspondingly the Mean Frame Size, *MFS*), while $p_{\min}$ is highly affected by the M value, given a q-value, the product $Mp_{\min}$ is almost constant. Specifically, results indicate that for a given message length, the product $Mp_{\min}$ has an asymptotic value with respect to M. Furthermore, when $M \geq 4$, the $Mp_{\min}$ values are very close to the asymptotic value. This is the reason for calling $Mp_{\min}$ an *invariant figure*, i.e., for a given *MFS* it is almost constant. Figure 2.29 clearly points out the $Mp_{\min}$ invariant behavior.

Table 2.6: Optimal p values

q values	MFS (Slots)	$M=2$		$M=4$		$M=10$		$M=50$		$M=100$	
		$p_{\min}$	$Mp_{\min}$	$p_{\min}$	$Mp_{\min}$	$p_{\min}$	$Mp_{\min}$	$p_{\min}$	$Mp_{\min}$	$p_{\min}$	$Mp_{\min}$
0.5	2	.26160	.52321	.11679	.46715	.04430	.44304	.00864	.43206	.00431	.43076
0.9	10	.18260	.36521	.07880	.31520	.02945	.29448	.00570	.28518	.00284	.28409
0.96	25	.13293	.26586	.05638	.22552	.02091	.20914	.00404	.20186	.00201	.20101
0.98	50	.10053	.20106	.04221	.16883	.01559	.15591	.00300	.15018	.00149	.14952
0.99	100	.07434	.14868	.03097	.12388	.01140	.11403	.00219	.10968	.00109	.10918

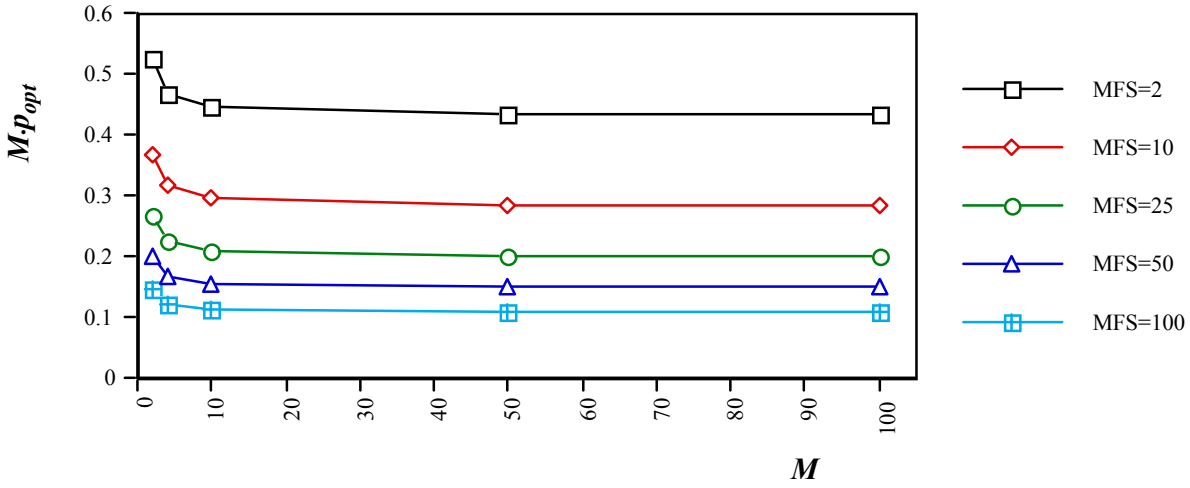


Figure 2.29: $Mp_{\min}$ "invariant behavior"

In [BCG03a] by exploiting an analytical model it is explained the rationale behind the $M \cdot p_{\min}$ quasi-constant value (for a given *MFS*).

We now investigate the relationship between $M \cdot p_{\min}(q)$ and the optimal S_U. As said before, $M \cdot p_{\min}(q)$ are derived assuming M active stations scheduling their transmission attempts in a slot selected according to a geometric distribution with parameter *p*. Furthermore, for each configuration an optimal value of parameter *p*, say $p_{\min}$, exists that guarantees a balance on the channel idle periods and collisions. Furthermore, in previous section we introduced the *slot utilization* parameter, S_U, to estimate the network contention level. Let us now investigate the relationship between S_U and $M \cdot p_{\min}$.

To this end, let us consider a *p*-persistent IEEE 802.11 protocol in which each station uses the optimal value $p_{\min}$. We denote with N_{tr} the number of stations that make a transmission attempt in a slot. Hence, $P\{N_{tr}=i\}$ is the probability that exactly *i* stations transmit in a slot, and $P\{N_{tr}=0\}$ is the probability that a slot remains empty. Let us now observe that $M \cdot p_{\min}$ is the average number of stations transmits in a slot:

$$M \cdot p_{\min} = \sum_{i=1}^{M} i \cdot P\{N_{tr}=i\} \geq \sum_{i=1}^{M} P\{N_{tr}=i\} = 1 - P\{N_{tr}=0\} = S\_U \qquad (2.11)$$

hence, $Mp_{\min}$ is an upper bound on the probability to observe a busy slot, i.e., $M \cdot p_{\min} \geq S\_U$. Furthermore, by noting that

$$S\_U = 1 - P\{N_{tr} = 0\} = 1 - (1-p)^M = -\sum_{k=1}^{M} \binom{M}{k} (-p)^k \geq Mp - \frac{M \cdot (M-1)p}{2}$$

and by observing the optimal $M \cdot p_{\min}$ values reported in Table 2.6, we can conclude that $Mp_{\min}$ is a tight upper bound of S_U. The accuracy of this approximation increases with the increase of the network congestion, i.e. number of active stations and/or message length. Results presented in Table 2.6 indicate that, for each *q*-value, the $M \cdot p_{\min}$ product results quasi-constant for *M* greater than 2, and hence it is possible to define a single quasi-optimal value for the $M \cdot p_{\min}$. To sum up, for each IEEE 802.11 physical layer parameters setting, it is possible to define a function of *q*, named *Asymptotic Contention Limit, ACL(q),* such that $ACL(q) = M \cdot p_{\min}(q)$. This function would represent the optimal slot-utilization level the system should obtain to guarantee its optimal behavior from the channel utilization viewpoint. The *ACL(q)* function can be computed off-line using the analytical model presented in [CCG00]. It is worth noting that *ACL(q)* identifies the optimal contention level without requiring any knowledge of the number of active stations in the system. This is important, because it is the basis for implementing an optimal window-tuning mechanism that does not require estimating the number of active stations in the system. The basic idea, is *i*) to estimate *S_U*, and *ii*) to disable the stations' transmissions when *S_U* > *ACL(q)*. To this end, in the AOB, we use the generalized *P_T* mechanism (see Equation (2.10)), and we set *opt_S_U(q)* equal to *ACL(q)*. Hence, in the AOB, the *P_T* formula is:

$$P\_T(ACL(q), S\_U, N\_A) = 1 - \min\left(1, \frac{S\_U}{ACL(q)}\right)^{N\_A} \tag{2.12}$$

Fixed a given *ACL(q)* value, the *P_T* values obtained fluctuates among 0 and 1. We named *Asymptotically Optimal Backoff* (AOB) a mechanism that, by using the *P_T* defined by Equation (2.12), guarantees a S_U value below the given *ACL(q)* value. The optimal slot utilization value (associated to ACL) can be only asymptotically achieved, for this reason the mechanism is named Asymptotically Optimal Backoff.

To be effective, the AOB mechanism requires the knowledge of the *q* value to identify the *ACL(q)*. The *ACL(q)* value together with the current *S_U* estimate determines the transmission probability.

The *q* value depends on the characteristics of the traffic that is transmitted in the network. Two approaches exist to compute its value: static vs. dynamic approach. In the static approach the *q* value is computed by taking into consideration the characteristics of the Internet traffic [S94]: on a byte count basis, 90% of the traffic is made up of maximum size packets (i.e. 512 bytes of user data) while the remaining 10% consists of very short packets (i.e. 10 bytes of user data). This approach is very simple but can introduce severe approximations on the real traffic that is transmitted in the network. Therefore, we prefer to adopt a dynamic approach: the *q* value is estimated by each station by observing the status of the channel through its carrier sensing mechanism. A dynamic approach, based on the channel status monitoring, is also used to obtain the current *S_U* estimate. The *q* and *S_U* estimation algorithms used hereafter to derive numerical results are reported in [BCG02a]. In that report we also discuss the *ACL(q)* computation.

It is worth noting that from a logical standpoint the AOB can be seen as a simple extension of the IEEE 802.11 mechanism (see Figure 2.30): each IEEE 802.11 transmission is deferred in a probabilistic way. However, this does not imply that AOB can be implemented on top of the standard protocol; indeed the mechanism is embedded within the standard protocol, since it requires information that is available inside the IEEE 802.11 network interface (backoff counter

value, carrier sensing information, etc.). Hence its implementation is a simple extension of the IEEE 802.11 protocol implemented in existing products.

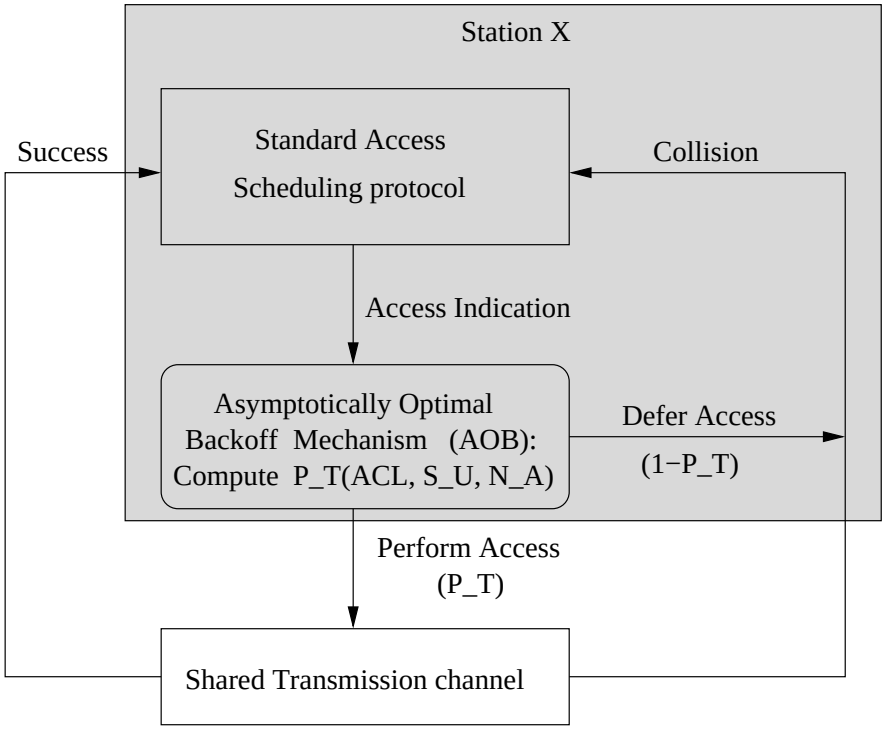


Figure 2.30:AOB and IEEE 802.11

2.5.2. AOB Performance Analysis

In this section, by means of the discrete event simulation, we show the effectiveness of the AOB mechanism. The main target of this performance study is to investigate the relationship between the channel utilization level and the network contention.

To perform this study we run a set of simulative experiments in which we change the number *M* of active stations. Active stations are assumed to operate in asymptotic conditions (i.e., with continuous transmission requirements). We use a maximum number of 200 active stations because the number of stations expected in the future for such a system could raise the order of hundreds [C94]. Using up to 200 active stations enable us to emphasize the system's characteristics, adaptiveness and scalability.

It is also interesting to note that other interesting performance indices as the Throughput and the Mean Access Delay are strongly correlated with the channel utilization level.

The Figure 2.31 shows the channel utilization level of the standard 802.11 DCF with respect to the optimal values calculated by means of the analytical model sketched in Section 2.2.1. For a given system configuration, the optimal channel utilization value has been obtained by computing p_{opt} as explained in Section 2.2.1. It is immediate to verify that the performance of the IEEE 802.11 standard protocol are negatively (low channel utilization) affected by high-contention situations. In fact, with the standard protocol the channel utilization level decreases when the contention grows; this implies that collisions and retransmissions reduce the amount of user data which is possible to deliver. Note that this problem occurs for each possible value of the mean payload size considered.

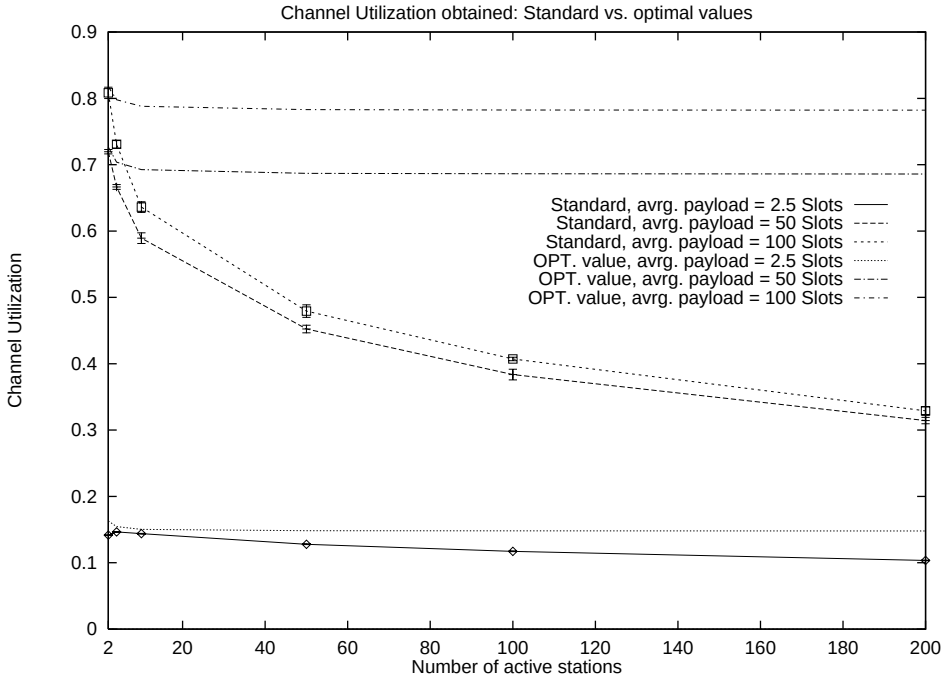


Figure 2.31: IEEE 802.11 channel utilization vs. optimal value

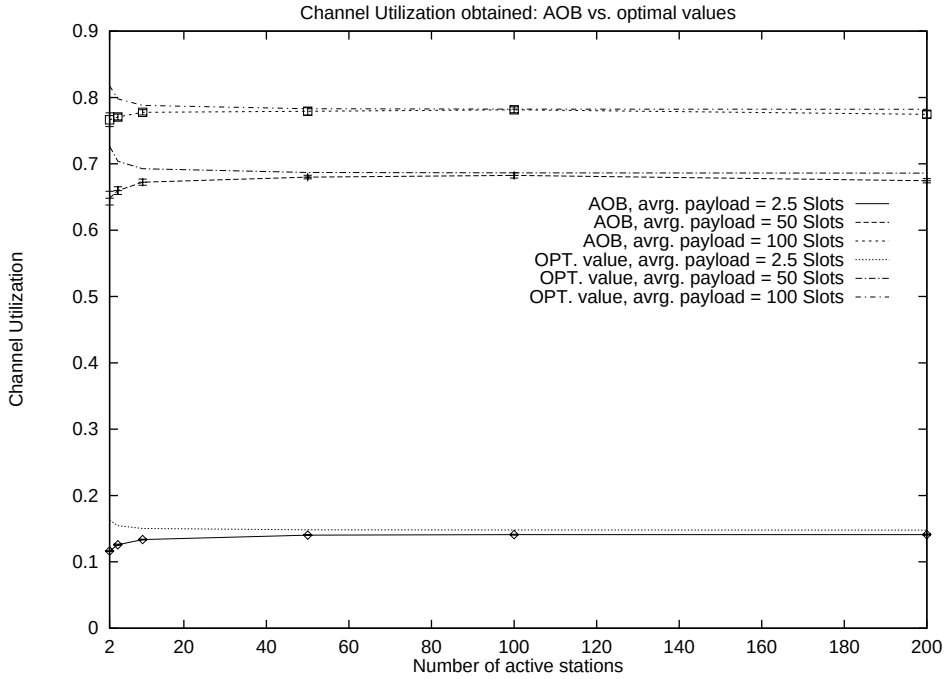


Figure 2.32: Channel utilization of the IEEE 802.11 protocol with the AOB mechanism vs. optimal value

The effectiveness of the proposed AOB mechanism is shown in Figure 2.32. This figure shows the channel utilization level achieved by adopting the AOB system and compares this index with the analytically-defined optimal utilization levels. The results show that the AOB mechanism leads an IEEE 802.11 network near to its optimal behavior at least from the channel utilization viewpoint.

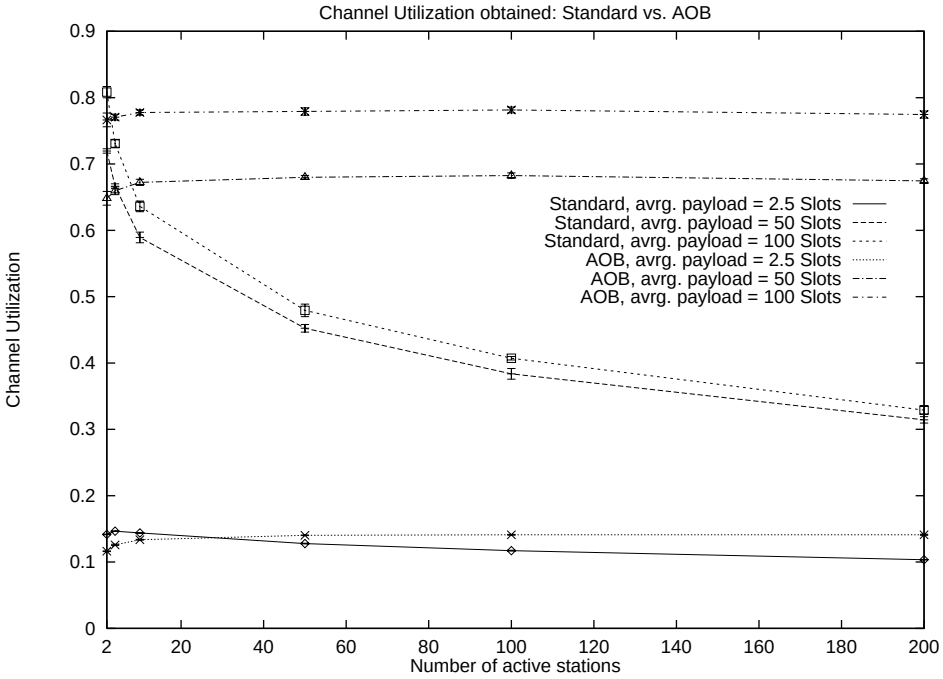


Figure 2.33:IEEE 802.11 channel utilization with or without the AOB mechanism

Only a little overhead is introduced when only few stations are active, as we can see in the direct comparison presented in Figure 2.33. Moreover, with the AOB mechanism, the channel utilization remains close to its optimal value even in high-contention situations. In such cases, AOB almost doubles the channel utilization with respect to the standard protocol.

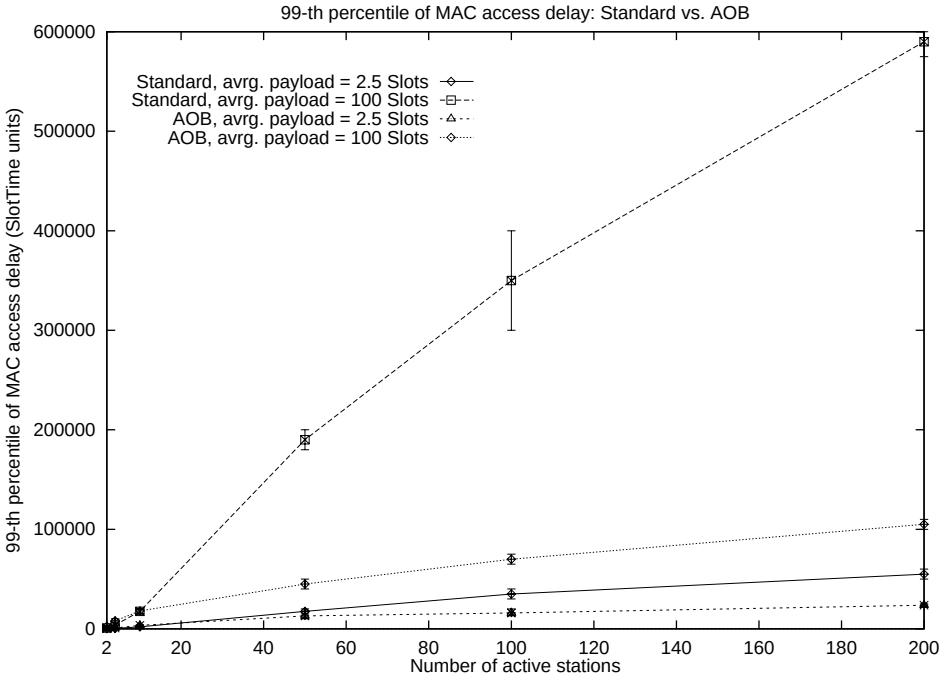


Figure 2.34:99-th percentile of MAC delay

The channel utilization provides information about the efficiency of a MAC protocol in sharing the channel among several stations. However, to measure the network QoS from the users' standpoint other performance indices must be used. The delay a user experience to transmit a packet is generally used to estimate the QoS a user can rely upon. In this section we utilize the MAC delay, i.e. the time interval between the first time a packet is scheduled for transmission and the instant at which its successful transmission is completed.

In Figure 2.34 we report the 99-th percentile of the MAC delay vs. contention level (i.e. number of active stations) for various average sizes of the transmitted frames. It results that the AOB mechanism leads to a great reduction of the worst case MAC delay with respect to the standard access scheme alone. This gives also a good indication of the reduced risk of starvation for transmissions. The AOB mechanism, by exploiting the priority effect induced by the *N_A* parameter used in the probability of transmission, increases the stations' *P_T* with the increase of MAC delay. This behavior enhances the fairness and the queue-emptying behavior of the system.

By a careful observation of Figure 2.33 and Figure 2.34 it is clear that the *N_A* priority mechanism is really effective in reducing the tail of the MAC Delay. For example, for average payload equal to 100 slots, the ratio between the 99-th percentile of the MAC Delay with or without the AOB mechanism is about 6 while the ratio between the average MAC Delay is about 2. Note that the average MAC Delay ratio is exactly the inverse of the channel utilization ratio (see Figure 2.33).

A detailed performance analysis of the IEEE 802.11 protocol, with and without the AOB mechanism, can be found in [BCG03a]. Simulation results indicate that the AOB mechanism is very effective both in steady-state, and under transient conditions. Furthermore, simulative results indicate that the mechanism is robust to errors and has potential for traffic differentiation.

2.6. Enhanced card novel mechanisms

A new medium access technology is necessary in order to provide the hardware and corresponding low-level software for the experimental (lab and field) verification and test (proof-of-concept) of MobileMAN. This aims to distinguish MobileMAN project by other existing projects in Ad Hoc networking. During this first project year, most of the work has been concentrated on the hardware architectural aspects, while software is being currently specified and will be developed in the next project phase.

2.6.1. State-of-the-art investigation

This preliminary investigation has been conducted and has been further broken down into:

- Study of both the 802.11/802.11a/802.11b standard; comparative analysis of the standard and the proposed modified MAC algorithm (still under development by CNR-IIT).
- Analysis of the available solutions for 802.11 cards; silicon manufactures and integrators have been investigated. This study did allow the pre-selection of the technology on which the final solution will be based. In Appendix B we present our detailed investigation of existing technologies.

Thanks to this work, the critical parts in the MAC algorithms (both the standard 802.11 and the proposed one) have been spotted. This did allow the choice of a well-adapted hardware/software platform.

2.6.2. Choice of an hardware medium-access platform for MobileMAN

In order to define realistic hardware architecture, an important choice has been taken preliminarily: the access technology was further broken down into BB/RF part and MAC parts. This allows choosing the most suited technology for both parts and gives more freedom in choosing the basic components (integrated circuits and chipsets). Thanks to this breakdown, a very flexible architecture has been reached. More in details, the technology for the BB/RF part has been firstly chosen; based on this, a system architecture and a processor for the MAC part have been selected:

- For the BB/RF part, the Intersil Prism-I chipset has been chosen. Although the Prism-I chipset is not the most powerful on the market today, it has the advantage of being extremely modular: MAC, BB and RF chips are distinct and their interface is clear and well documented.
- For the MAC part, a very powerful Texas Instruments floating point DSP (TMS6701) has been chosen.

Having chosen the technologies for the medium access card, 2 commercial products have been selected, which offered simple interfaces:

- For the BB/RF part, a product by the company Elektrobit AG has been chosen (http://www.elektrobit.ch/products/sequence/index.html). The chosen Sequence™ modem is relatively easy to adapt to the MobileMAN requirements (in contrast to off-the-shelf 802.11 cards, which are virtually impossible to modify). A modified version of the modem has been ordered to Elektrobit and has already been delivered.
- For the MAC processor, an off-the-shelf OEM module from company Orsys has been selected, namely the *micro-line® C6713Compact* (www.orsys.de). The TI C67x processor family is very powerful (almost over-dimensioned for the task), but having a very

powerful processor ensures enormous advantages in the future project phases (simple software development using C language, quick tryouts). Several variants of the modified MAC may be programmed and checked without having to worry about the computing resources. The software tools are based on TI Code Composer Studio for C6000 (www.ti.com).

2.6.3. The medium-access level software architecture

For the software, a flexible architecture is under development, i.e. an architecture, which not only allows the implementation of the MAC software alone, but which also allows flexibility and future extensions. Figure 2.35 shows a preliminary simplified scheme of the architecture of the MAC card software.

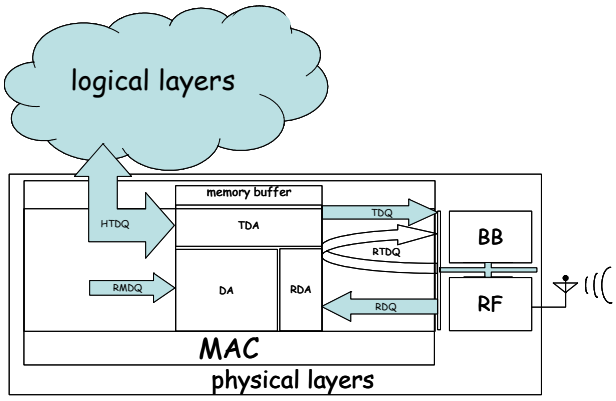


Figure 2.35: MobileMAN (enhanced) MAC card software architecture (simplified view)

The processor unit (PU), which supports this new MAC, has to manage the data flow between the communication channel and the host (logical layers) in both directions and channel-to-channel in case of atomic operation (e.g. RTS/CTS). This means that the PU needs a specialized data structure. Since the communication must also satisfy the stringent time constraints and frames hierarchy imposed by the IEEE 802.11 standard, the data structure must be optimized in terms of numbers of memory access, numbers of data swap and storage mechanism.

It has been chosen to implement a buffer management scheme with a descriptor mechanism; this allows to efficiently process receive- and transmit data packets in place, and to eliminate packet copy or swapping. The buffer memory is configured in three different areas:

- data area (DA) for storing data from the logical layers to the PHY, frames ready to be transmitted, frames received from the channel and data from the PHY to the logical layers;
- transmission descriptor area (TDA) for storing descriptors pointing to the data packets ready for transmission (to the channel or to the logical layers);
- receive descriptor area (RDA) for storing all descriptors pointing to each received data packet.

Each descriptor points to a specific data packet/frame and all descriptors are arranged in different queues: transmission descriptor queue (TDQ) for the transmission on the channel, host transmission descriptor queue (HTDQ) for the transmission to the logical layers, receive descriptor queue (RDQ) for the received frames, receive transmission descriptor queue (RTDQ) for those frames of an atomic operation, and reusable memory descriptor queue (RMDQ) for the reusable

data area. In this way, the PU manages priorities, hierarchy and sequence of the frames/data packets.

Thanks to the descriptor mechanism, the PU doesn't need to move, swap, copy or erase data in memory, but it has only to change some flags in the control register or to change the value of the address register that are stored in each descriptor. This data structure is easily improvable for new MAC features (which are currently being investigated by other MobileMAN from a theoretical point of view), as for instance:

- cross layering: several mechanisms can profit by the knowledge of some parameters that are typically confined at the MAC layer, like transport, power management, cooperation, etc.;
- MAC-level routing: a packet received at the wireless interface must be passed up to the routing layer (in order to discover the next hop), and further down to the same wireless interface for transferring it to the next hop; this adds undesirable delay and overhead at both MAC and routing layer.

2.7. References

[ACG03] G. Anastasi, M. Conti, E. Gregori, "IEEE 802.11 Ad Hoc Networks: Protocols, Performance and Open Issues", *Mobile Ad hoc networking*, S. Basagni, M. Conti, S. Giordano, I. Stojmenovic (Editors), IEEE Press and John Wiley and Sons, Inc., New York, 2003.

[B01] C. Bisdikian, "An Overview of the Bluetooth Wireless Technology", *IEEE Communication Magazine,* December 2001.

[BCD00] L. Bononi, M. Conti, L. Donatiello, "Design and Performance Evaluation of a Distributed Contention Control (DCC) Mechanism for IEEE 802.11 Wireless Local Area Networks", *Journal of Parallel and Distributed Computing*, Accademic Press Vol.60 N.4, April 2000.

[BCGG03] R. Bruno , M. Conti, E. Gregori, "Optimal Capacity of p-persistent CSMA Protocols", IEEE Comm. Letters, March 2003

[BCG03a] L. Bononi, M. Conti, E. Gregori, ""Run-Time Optimization of IEEE 802.11 Wireless LANs performance", *IEEE Transaction on Parallel and Distributed Systems* Vol 14, N. 12 December 2003.

[BCG01] R. Bruno, M. Conti, E. Gregori, "A Simple Protocol for the Dynamic tuning of the backoff mechanism in IEEE 802.networks", *Computer Networks*, Vol 37 Iss 1. pp 33-44

[BCG02] R. Bruno, M. Conti, and E. Gregori, Optimization of Efficiency and Energy Consumption in p-persistent CSMA-based Wireless LANs, *IEEE Transactions on Mobile Computing*, Vol 1. N.1 (2002), pp. 10-31.

[BCG02a]. Technical report L. Bononi, M. Conti, E. Gregori, "Run-Time Optimization of IEEE 802.11 Wireless LANs performance", *IIT Internal report*, July 2002

[BFO96] G.Bianchi, L.Fratta, M. Oliveri, "Performance Evaluation and Enhancement of the CSMA/CA MAC Protocol for 802.11 Wireless LANs", *Proceedings of PIMRC* 1996, October 1996, Taipei, Taiwan, pp. 392-396.

[BLU] Web site of the Bluetooth Special Interest Group: http://www.bluetooth.com/

[BLU01] Specification of the Bluetooth System, Version 1.1, February 2001.

[C03] M. Conti, "Body, Personal, and Local Wireless Ad Hoc Networks", Chapter 1 in *Handbook of Ad Hoc Networks* (M. Ilyas Editor), CRC Press, New York, 2003.

[C94] Chen K.C., "Medium Access Control of Wireless LANs for Mobile Computing", IEEE Networks, 9-10/1994.

[CCG00] F. Calì, M. Conti, E. Gregori, "Dynamic Tuning of the IEEE 802.11 Protocol to Achieve a Theoretical Throughput Limit", *IEEE/ACM Transactions on Networking*, Volume 8 , No. 6 (Dec. 2000), pp. 785 - 799.

[CCG00a] F. Calì, M. Conti, E. Gregori, "Dynamic IEEE 802.11: design, modeling and performance evaluation", *IEEE Journal on Selected Areas in Communications*, 18(9), September 2000, pp. 1774-1786.

[CGL97] M. Conti, E. Gregori, L. Lenzini, "Metropolitan Area Networks" Springer Limited series on Telecommunication Networks and Computer Systems, November 1997.

[CMC99] M. S. Corson, J.P. Maker, J.H. Cernicione, "Internet-based Mobile Ad Hoc Networking", IEEE Internet Computing, July-August 1999, pp. 63-70.

[E02] A. Ephremides, "Energy Concerns in Wireless Networks", IEEE Wireless Communications, August 2002, pp. 48-59.

[EM02] E. Mingozzi, "QoS Support By The HiperLAN/2 MAC Protocol: A Performance Evaluation", *Cluster Computing Journal ,* Vol. 5, No. 2 (April 2002).

[ETSI] ETSI Technical Report 101 683, V1.1.1, "Broadband Radio Access Networks (BRAN): HIgh PErformance Local Area Network (HiperLAN) Type 2; System Overview".

[Glo02] [GLOM] GloMoSim, Global Mobile Information Systems Simulation Library, http://pcl.cs.ucla.edu/projects/glomosim/.

[GGK01] Piyush Gupta, Robert Gray, P. R. Kumar, "An Experimental Scaling Law for Ad Hoc Networks." http://black.csl.uiuc.edu/~prkumar/postscript_files.html, 2001.

[GK00] Piyush Gupta, P. R. Kumar, "The Capacity of Wireless Networks," IEEE Transactions on Information Theory, vol. IT-46, no. 2, pp. 388-404, March 2000.

[HS82] D. P. Heyman, M. J. Sobel, "Stochastic models in operations research" Vol. I, McGraw-Hill Book Company, 1982.

[IEEE802] Web site of the IEEE 802.11 WLAN: http://grouper.ieee.org/grups/802/11/main.html

[IEEE97] IEEE standard for Wireless LAN- Medium Access Control and Physical Layer Specification, P802.11, November 1997. See also IEEE P802.11/D10, 14 January 1999.

[KSY84] J.F. Kurose, M. Schwartz, Y. Yemini, "Multiple access protocols and time constraint communications", *ACM computing Surveys*, Vol. 16, pp.43-70.

[MB00] B.A. Miller, C. Bisdikian, *Bluetooth Revealed,* Prentice Hall, 2000.

[MC03] J.P. Macker, S. Corson, "Mobile Ad hoc Networks (MANET): Routing technology for dynamic, wireless networking", in *Mobile Ad hoc networking*, S. Basagni, M. Conti, S. Giordano, I. Stojmenovic (Editors), IEEE Press and John Wiley and Sons, Inc., New York, 2003.

[Ns02] The Network Simulator - ns-2, http://www.isi.edu/nsnam/ns/index.html.

[OG02] Mohammad S. Obaidat, David G. Green, "An Accurate Line of Sight Propagation Performance Model for Ad Hoc 802.11 Wireless LAN (WLAN) Devices", Proc. ICC 2002, New York City, April 28 – May 2, 2002.

[Qua02] Qualnet simulator, http://www.qualnet.com/.

[S94] W.R.Stevens, *TCP/IP Illustrated*, Vol 1, Addison Wesley, 1994.

[S96] W. Stallings, *Local & Metropolitan Area Networks,* Prentice Hall, 1996.

[ZD03] G. Zaruba, S. Das, "Off-the-Shelf Enablers of Ad Hoc Networks", in *Mobile Ad hoc networking*, S. Basagni, M. Conti, S. Giordano, I. Stojmenovic (Editors), IEEE Press and John Wiley and Sons, Inc., New York, 2003.

[Bernasconi2003] Ralph Bernasconi, Ivan Defilippis, Silvia Giordano and Alessandro Puiatti; "An enhanced MAC architecture for multi-hop wireless networks"; to appear in proceeding of PWC2003, September 23-25, 2003, Venice, Italy.

3. NETWORKING

To cope with the self-organizing, dynamic, volatile, peer-to-peer communication environment in a MANET, most of the main functionalities of the *Networking protocols* (i.e., network and transport protocols in the Internet architecture) need to be re-designed. In this section, we survey the existing literature, we pointed out of the main the research issues, and we present and discuss MobileMAN solutions.

The aim of the networking protocols is to use the one-hop transmission services provided by the enabling technologies to construct end-to-end (reliable) delivery services, from a sender to one (or more) receiver(s). To establish an end-to-end communication, the sender needs to locate the receiver inside the network. The purpose of a *location service* is to dynamically map the logical address of the (receiver) device to its current location in the network. Current solutions generally adopted to manage mobile terminals in infrastructure networks are generally inadequate, and new approaches have to be found.

Once, a user is located, *routing and forwarding algorithms* must be provided to route the information through the MANET. Finally, the low reliability of communications (due to wireless communications, users' mobility, etc.), and the possibility of network congestion require a *re-design of Transport Layer mechanisms.*

Hereafter, we analyze the various aspects of the research on networking protocols with reference to the MobileMAN context, i.e., location service, routing and forwarding, and transport protocols.

3.1. Nodes' Location

A Location Service is a system able to maintain nodes positions inside a database, implementing two fundamental operations:

- *Position Update*: to update the database with the current position of a specified node
- *Position Lookup*: to retrieve the current (i.e. last submitted) position of a specified node

In legacy mobile networks [LC00] (e.g. GSM, Mobile IP), the presence of a fixed infrastructure led to the diffusion of two-tier schemes to track the position of mobile nodes. Examples are the Home Location Register/Visitor Location Register approach used in GSM networks, and the Home Agent/Foreign Agent approach for Mobile IP networks. Efficient implementations of these approaches make a heavy use of the network infrastructure use in both position database maintenance and usage. In a mobile ad hoc network, the lack of infrastructure makes these solutions not useful, and new approaches have to be found for nodes location management.

On one hand a *reactive* location service scheme would represent a simple solution, based on flooding location lookups throughout the network, and using no position database. Of course, flooding does not scale, and hence this approach is only suitable for limited size networks, where frequently flooded packets have only a limited impact on network performance. Controlling the flooding area can help to refine the technique. This can be achieved by gradually increasing, until the node is located, the number of hops involved in the flooding propagation. In this approach all the complexity is associated with lookup operations.

On the other hand, proactive location services subdivide the complexity in the two phases. Proactive services construct and maintain inside the network, data structures (i.e. database) that store the position information of each node. By exploiting the data structures, the query operations are highly simplified.

DREAM [BCSW98] is an example of a proactive location service in which all the complexity is in the first phase. All the network nodes maintain the location information of all the other nodes. To this end, each node uses the flooding technique to broadcast its location. To reduce the overhead, a node can control the frequency with which its sends its position-update messages, and the area (number of hops) to which the update messages are delivered. In this way, the location information

accuracy decreases with the distance from the node but this shortcoming is balanced by the *distance effect*: "the greater the distance separating two nodes, the slower they appear to be moving with respect to each other" [BCSW98].

The location services presented in [GH99][IS99][LJD99][PH01] select for each node a subset of network nodes that are designed to store its location. These works follow two main approaches: *virtual home* and *grid.*

[GH99] and [IS99] use a similar approach to identify the location servers of a node by distributing the duty on several nodes inside the ad hoc network. Specifically, each node is univocally associated with an area inside the ad hoc network (i.e., its *virtual home*) in which its position information is stored. The association between a node and its virtual home area is obtained through a mathematical function (known to all nodes) working on the node identifier. The query related to a node location is therefore directed to its *virtual home*, where the node information is stored.

[LJD99] and [PH01] assume that a grid-like structure is superposed on the ad hoc network. By exploiting the grid structure the location service is organized in a hierarchy of squares that simplifies the update and query operations. For example, in [LJD99], the grid hierarchy and the node identifiers define for each mobile node a small set of other nodes (its location servers) designed to contain its current location. A node has no knowledge about the identity of its location servers, but the protocol defines a distributed and independent procedure to identify them. A node only forwards its position updates toward grid squares. Then, locally to each selected grid square, the distributed procedure finds one location server for that node. The same distributed procedure is also used to locate the node location server to solve the queries. It is worth of note that again, the duty of maintaining the position information of a given node, is associated to some nodes in the network (and not all of them), but instead of having the location servers all co-located in the same small area (*virtual home*) they are distributed following the grid partitioning. If on one hand this approach increases reliability, on the other hand produces more communicational complexity.

[GT03] contains an updated overview of Location Services for ad hoc networks. In the next sections we present solutions currently designed for Location services in the context of the MobileMAN project.

3.1.1. Location Services in MobileMAN

Node locations for small medium size networks will be directly based on link state routing information. For large scale MANET we propose an approach which is *context-aware*. The key aspect exploited in this approach, is to use knowledge about areas in the network where network density is usually good. We call these areas *hot spots* (with no reference to infrastructure wireless networks).

On a metropolitan area, people do not uniformly distribute, but tend to assemble around hot spots. These are for example offices, shopping malls, leisure centers or similar places where people are plunged in well defined contexts. Beside that, a user tends to frequent the same contexts, where often the same people can be met. Clearly, daily habits change, but this happens over long periods of time comparing with the lifetime of ad hoc networks.

If thousands of PDA-equipped users are connected on a metropolitan Ad Hoc network, the facts pointed out above will result in users possibly aware of places where connectivity and node density are usually good. People can identify and easily remember these locations, which we call *hot spots*. Moreover, this process could be directly supported using mobile hosts. For example each mobile host could build and maintain a ranking of positions where it passed over and where connectivity and node density was good. A user would than been able to consult and use the ranking to specify hot spot locations.

Once a user identifies a hot spot context, the system can exploit it to co-locate its location servers.

Finally, this information could be distributed along with the unique identifier of the node, resulting in a composed address structure similar to current e-mail addresses:

$$Unique\text{-}ID@(X_{hot\text{-}spot}, Y_{hot\text{-}spot})$$

In this way users would distribute *context-aware* addresses as present e-mail addresses are exchanged today, giving seekers nodes a way to find the location servers of sought-after nodes.

In describing the internal of the context-aware location service, we first give the necessary assumption and definitions.

Each node is able to sense its current position through an on-board GPS device. Beside that, we do not assume a uniform distribution of the nodes on the network area. This assumption is taken in both [GH99] and [LJD99], but do not represent a realistic scenario even if it helps demonstrating formal properties of the systems.

Consider a node N and a hot spot with coordinates $(X_{hot\text{-}spot}, Y_{hot\text{-}spot})$ specified by its user. The system potentially identifies the location servers of N, as those nodes currently located around the hot spot. More formally, the location servers of N belong to its *friends set* (F_N), defined as the set of nodes located at distance less than or equal to a value *r* from the point $(X_{hot\text{-}spot}, Y_{hot\text{-}spot})$:

$$F_N = \{\text{all nodes } k \text{ such that, } | \text{CurrentPos(k)} - (X_{hot\text{-}spot}, Y_{hot\text{-}spot}) | \leq r\}$$

Note that not all the F_N nodes are location servers for N. Some of them might be unreachable, while others might have just entered the area, without having yet received a position update. In fact, node N has no control over its F_N area, and no knowledge about its location servers. The node periodically sends a position update request towards the friends set, simply asking to cache the position information for a limited amount of time. We designed the context-aware location service to be a soft state system: positions are only temporarily cached on location servers. This eliminates the problem of handling stale entries.

We now describe the internals of a Position Update and subsequently the procedure.

Node N periodically updates its current position to its location servers, as the system is soft state.

The caching time, and respectively the update frequency, varies accordingly to the node's mobility pattern: the faster the node moves, the more frequent it has to update its position (see [T03] [CGT03] for details). A position update consists in the following steps:

1. An update request packet is geographically forwarded towards F_N. By definition, the forwarding procedure will deliver the packet to the reachable node L that closest to the F_N coordinates
2. If L is at distance greater than r, then a failure will return back to N. Otherwise, L floods the position update request bounding to the F_N area. Each node receiving the position update request caches N 's position and broadcasts again the packet only if it belongs to F_N otherwise it simply drops it
3. Node L collects acknowledgements from caching nodes and sends back a cumulative packet to N

Nodes distribution inside F_N can be far from uniform. In this case, the packet flooding described above, does not guarantee N to have its position cached in a satisfying number of nodes spanning the F_N area. To minimize this condition, node L could drive the flooding to start also from other three points located inside F_N, identified as projections of the current position (X_L, Y_L) of node L. Having the position (X_L, Y_L) and the F_N coordinates $(X_{hot\text{-}spot}, Y_{hot\text{-}spot})$, points P_1, P_2 and P_3 are identified by:

$$P_1 = (X_{hot\text{-}spot} - Y_L, Y_{hot\text{-}spot} + X_L - X_{hot\text{-}spot})$$
$$P_2 = (2X_{hot\text{-}spot} - X_L, 2Y_{hot\text{-}spot} - Y_L)$$
$$P_3 = (X_{hot\text{-}spot} + Y_L - Y_{hot\text{-}spo}, Y_{hot\text{-}spot} - X_L + X_{hot\text{-}spot})$$

A Position Lookup procedure is initiated by a node M willing to communicate with node N, that has the address $N_{Unique\text{-}ID}@(X_{hot\text{-}spot}, Y_{hot\text{-}spot})$.

There are two cases:

1. N is sufficiently close to M, so that they directly see each other. In this case there is no need to query the location service
2. N is far away from M, the two nodes do not directly see each other, and M needs to query the location service to obtain N's position

A position lookup is needed in the second case. Node M geographically forwards a position lookup request packet towards $(X_{hot\text{-}spot}, Y_{hot\text{-}spot})$. Eventually a node selected by the forwarding procedure, caching N's position, will send back a lookup answer packet. If none exists, an error is reported to M. It is worth of note, that in order to lookup the current position of a node, the size r of the F_N area is not needed.

Finally, we introduce a mathematical model to have *friends set areas* of variable size. The model works again using context information like the network density (number of nodes in F_N) around the hot spot coordinates. Intuitively, the higher is the density around $(X_{hot\text{-}spot}, Y_{hot\text{-}spot})$ the smaller will the F_N radius be, and vice versa (see [T03] [CGT03] for details).

3.2. Routing

The highly dynamic nature of a mobile ad hoc network results in frequent and unpredictable changes of network topology, adding difficulty and complexity to routing among the mobile nodes. The challenges and complexities, coupled with the critical importance of routing protocol in establishing communications among mobile nodes, make routing area the most active research area within the MANET domain. Numerous routing protocols and algorithms have been proposed, and their performance under various network environments, and traffic conditions have been studied and compared.

Several surveys and comparative analysis of MANET routing protocols have been published [RT99], [ER03]. [P00] provides a comprehensive overview of routing solutions for ad hoc network, while an updated and in depth analysis of routing protocols for mobile ad hoc network is presented in [ER03].

A preliminary classification of the routing protocols can be done via the type of cast property, i.e. whether they use a *Unicast, Geocast, Multicast, or Broadcast* forwarding [PK].

Broadcast is the basic mode of operation over a wireless channel; each message transmitted on a wireless channel is generally received by all neighbors at one-hop from the sender. The simplest implementation of the broadcast operation to all network nodes is by naive flooding, but this may cause the *broadcast storm problem* due to redundant re-broadcast [NTCS99]. Schemes have been proposed to alleviate this problem by reducing redundant broadcasting. [SW03] surveys existing methods for flooding a wireless network intelligently.

Unicast forwarding means a one-to-one communication, i.e., one source transmits data packets to a single destination. This is the largest class of routing protocols found in ad hoc networks.

Multicast routing protocols come into play when a node needs to send the same message, or stream of data, to multiple destinations. Geocast forwarding is a special case of multicast that is used to deliver data packets to a group of nodes situated inside a specified geographical area. Nodes may join or leave a multicast group as desired, on the other hand, nodes can only join or leave a geocast group only by entering or leaving the corresponding geographical region. From an implementation standpoint, geocasting is a form of “restricted” broadcasting: messages are delivered to all the nodes that are inside a given region. This can be achieved by routing the packets from the source to a node inside the geocasting region, and then applying a broadcast transmission inside the region. Position-based (or location-aware) routing algorithms, by providing an efficient solution for forwarding packets towards a geographical position, constitute the basis for constructing geocasting delivery services. Hereafter, we surveyed the characteristics of unicast routing protocols that a relevant for MobileMAN, a comprehensive analysis of MANET routing protocols can be found in [ER03][CCL03].

3.2.1. Unicast Routing Protocols

A primary goal of unicast routing protocols is the correct and efficient route establishment and maintenance between a pair of nodes, so that messages may be delivered reliably and in a timely manner.

MANET routing protocols are typically subdivided into two main categories: *proactive routing protocols and reactive on-demand routing protocols* [RT99]. Proactive routing protocols are derived for legacy Internet distance-vector and link-state protocols. They attempt to maintain consistent and updated routing information for every pair of network nodes by propagating, proactively, route updates at fixed time intervals. As the routing information is usually maintained in tables, these protocols are sometimes referred to as Table-Driven protocols. Reactive on demand routing protocols, on the other hand, establish the route to a destination only when there is a demand for it. The source node through the route discovery process usually initiates the route requested. Once a route has been established, it is maintained until either the destination becomes inaccessible (along every path from the source), or until the route is no longer used, or expired [RT99][ER03].

Most work on routing protocols is being performed in the framework of the IETF MANET working group, where four routing protocols are currently under active development. These include two reactive routing protocols, AODV and DSR, and two proactive routing protocols, OLSR and TBRPF. There has been good progress in studying the protocols' behavior (almost exclusively by simulation), as can be seen in the large conference literature in this area, but the absence of performance data in non-trivial network configurations continues to be a major problem. The perception is that of a large number of competing routing protocols, a lack of WG-wide consensus, and few signs of convergence [MAN02]. To overcome this situation, a discussion is currently ongoing to focus the activities of the MANET WG towards the design of IETF MANET standard protocol(s), and to split off related long-term research work from IETF. The long term research work may potentially move to the IETF's sister organization, the IRTF (Internet Research Task Force) that has recently established a group on "Ad hoc Network Scaling Research".

PROACTIVE ROUTING PROTOCOLS. The main characteristic of these protocols is the constant maintaining of a route by each node to all other network nodes. The route creation and maintenance are performed through both periodic and event-driven (e.g., triggered by links breakages) messages. MANET IETF proactive protocols are: Optimized Link State Routing (OLSR), and Topology Dissemination Based on Reverse-Path Forwarding (TBRPF).

OLSR protocol [JMQ98] is an optimization for MANET of legacy link-state protocols. The key point of the optimization is the *multipoint relay* (MPR). Each node identifies (among its neighbors) its MPRs. By flooding a message to its MPRs, a node is guaranteed that the message, when retransmitted by the MPRs, will be received by all its two-hop neighbors. Furthermore, when exchanging link-state routing information, a node lists only the connections to those neighbors that have selected it as MPR, i.e., its Multipoint Relay Selector set. The protocol selects bi-directional links for routing, hence avoiding packet transfer over unidirectional links.

Like OLSR, TBRPF [BOT01] is a link-state routing protocol that employs a different overhead reduction technique. Each node computes a shortest-path tree to all other nodes, but to optimize bandwidth only part of the tree is propagated to the neighbors, for details see [ER03].

REACTIVE ROUTING PROTOCOLS. These protocols depart from the legacy Internet approach. To reduce the overhead, the route between two nodes is discovered only when it is needed. Representative reactive routing protocols include: Dynamic Source Routing (DSR), Ad hoc On Demand Distance Vector (AODV).

DSR is a loop-free, source based, on demand routing protocol [JM96], where each node maintains a route cache that contains the source routes learned by the node. The route discovery process is only initiated when a source node do not already have a valid route to the destination in its route cache; entries in the route cache are continually updated as new routes are learned. Source routing is used for packets' forwarding.

AODV is a reactive improvement of the DSDV protocol. AODV minimizes the number of route broadcasts by creating routes on-demand [PR99], as opposed to maintaining a complete list of routes as in the DSDV algorithm. Similar to DSR, route discovery is initiated on-demand, the route request is then forward by the source to the neighbors, and so on, until either the destination or an intermediate node with a fresh route to the destination, are located.

DSR has a potentially larger control overhead and memory requirements than AODV since each DSR packet must carry full routing path information, whereas in AODV packets only contain the destination address. On the other hand, DSR can utilize both asymmetric and symmetric links during routing, while AODV only works with symmetric links (this is a constraint that may be difficult to satisfy in mobile wireless environments). In addition, nodes in DSR maintain in their cache multiple routes to a destination, a feature helpful during link failure. In general, both AODV and DSR work well in small to medium size networks with moderate mobility.

Despite the large volume of research activities and rapid progress made in the MANET routing protocols in the past few years, this research area still harbor many open issues. There has been good progress in studying the protocols' behavior almost exclusively by simulation. Currently, only few measurements studies on real ad hoc testbeds can be found in the literature, see e.g., [BMJ00] [APE02]. The results from these testbeds are very important as they are pointing out problems that were not detected by preceding simulation studies, see e.g., the so-called *communication gray zones* problem [LNT02].

For this reason in the framework of the MobileMAN project we have set up a testbed to measure the performance of one reactive (AODV) and one proactive (OLSR) routing protocols.

Hybrid routing protocols integrate the characteristics of proactive and reactive routing protocols and exhibit proactive behavior given a certain set of circumstances, while exhibiting reactive behavior given a different set of circumstances. These protocols allow for flexibility based on the characteristics of the network. For these reasons, we included in the MobileMAN testbed also a hybrid routing protocol: the Zone Routing Protocol (ZRP) [PH01].

The Zone Routing Protocol (ZRP) integrates both proactive and reactive routing components into a single protocol. Around each node, ZRP defines a zone whose radius is measured in terms of hops. Each node utilizes proactive routing within its zone and reactive routing outside of its zone. Hence, a given node knows the identity of and a route to all nodes within its zone. When the node has data packets for a particular destination, it checks its routing table for a route. If the destination lies within the zone, a route will exist in the route table. Otherwise, if the destination is not within the zone, a search to find a route to that destination is needed.

3.2.2. A Testbed for Experimenting MANET IETF and Novel Routing protocols

As explained in the previous sections, experimenting existing (and novel) routing protocols in real testbeds is very important to deeply understand their behavior in a real environment. Existing testbeds [LLNN01] have demonstrated un-expected behavior of routing protocols that were not identified with simulations. "Gray zones" [LNT03] and route instabilities are behaviors detected with implementations on real nodes since the simulations do follow predefined mobility models and stable radio propagation. Therefore, in order to validate the final protocol design that corroborates the simulated protocol behavior, an implementation is required.

The state of the art in terms of Ad Hoc frameworks and test beds define a routing layer working with a single protocol. The routing layer may change the routing protocol but still there is a single protocol executed at any time.

In the framework of the MobileMAN project we decided to design and develop a novel "multiprotocol" architecture to investigate the routing protocols behavior, both in isolation and when different routing algorithms running simultaneously in the same node. This architecture requires a node taxonomy [CKN01] that differentiates nodes with low resources running a single IETF MANET protocol, versus "multiprotocol" nodes with enough resources that assist the overall routing process in the network. This architecture allows having a huge diversity of nodes with different routing protocols. The MobileMAN "multiprotocol" nodes will interact with those native IETF MANET nodes and will cooperate with them in order to extend the network lifetime.

The limitations of Ad Hoc nodes running a single routing protocol is well known but before moving forward and exploring new routing paradigms it is worth to experiment the routing behavior of nodes running a combination of different protocols. Thus, we provide backward compatibility with the Ad Hoc nodes that implement purely MANET IETF routing protocols, but in addition we incorporate the benefits of new routing algorithms.

Figure 3.1 presents the proposed MobileMAN node architecture for allocating the legacy MANET IETF protocols (at least the protocols proposed RFC experimental) but also providing room for including the protocols designed within the MobileMAN project. The proposed architecture allows including new "independent routing modules" that will access the common elements such as the routing tables and other lower layer functionality. This approach enables the possibility of having proactive and reactive routing protocols in addition to new novel routing proposals co-operating in the routing process. The novel routing protocols (which will be designed and implemented for medium to large Ad Hoc networks) are depicted as "independent routing modules" with gray color. In addition to these novel proposals for the routing algorithms, other paradigms such as cross-layer routing architecture can be added into the proposed MobileMAN node architecture. Thus, the proposed Ad hoc framework should provide modular and flexible node architecture for including and removing routing modules in order to design the suitable routing protocol but still supporting legacy protocols for backward compatibility. Figure 3.1 shows this modular Ad Hoc framework including the "independent routing modules" but also the "common module", which contains a "Registry" and a "Common Cache". The "Registry" keeps information about the protocols that are running simultaneously in the node. The "Common Cache" is a placeholder for routing information collected by the "independent routing modules" such as IP addresses, hostname, geographical location information, services provided by the node, etc. This information is relevant for the routing modules and it requires a common place accessible from the multiple modules. However, adding new information to the routing table does not have to overload the kernel from its normal routing functionality. Therefore, the "Common Cache" provides a flexible mechanism for adding new functionality to the framework and storing new data required by novel routing protocols but still keeping the performance of the kernel.

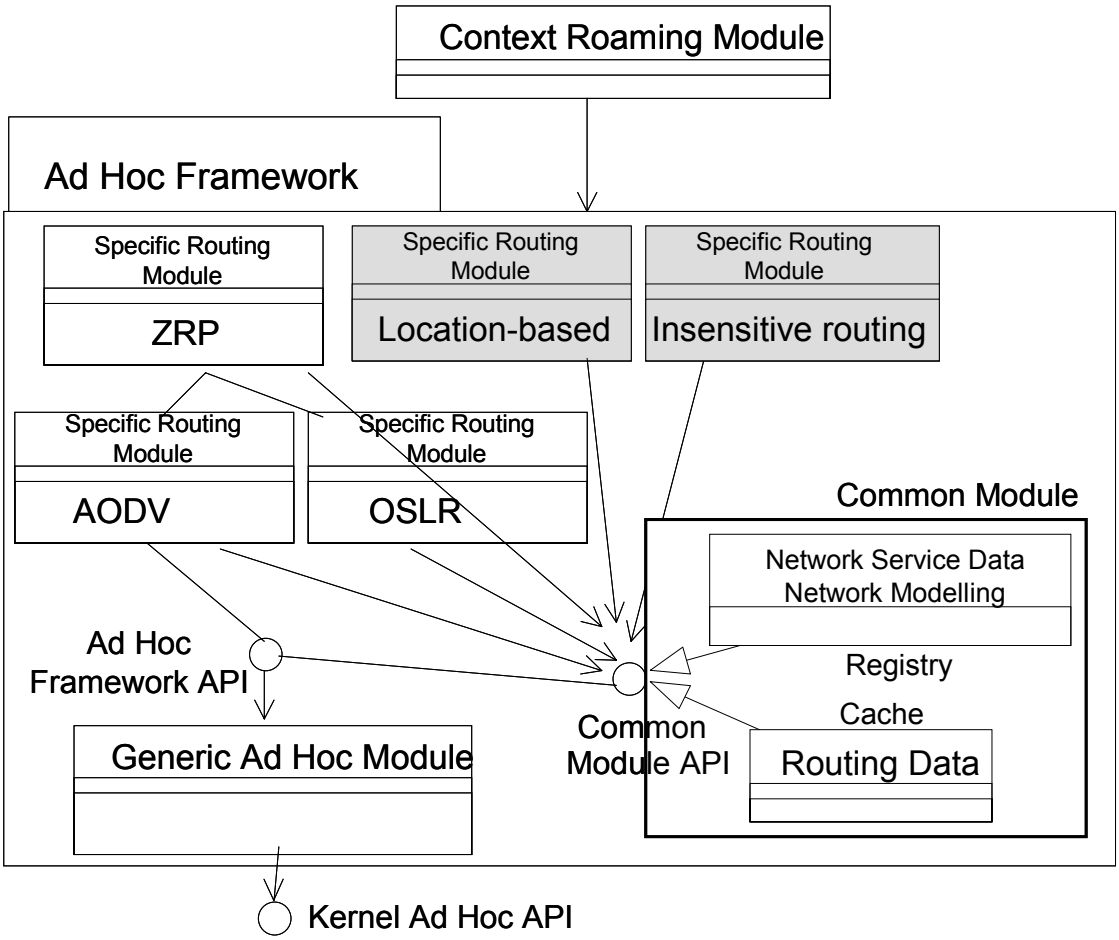


Figure 3.1. MobileMAN node routing architecture

The final output of this effort is the implementation of an Ad Hoc framework and the integration on real nodes. This target will include in the next phase a set of new novel proposals to overcome the routing problems in medium to large networks.

At this stage the Ad Hoc framework contains the modules depicted in Figure 3.1, excluding the modules in gray, which will be part of the next MobileMAN goal to overcome the limitations from legacy MANET routing protocols. The Ad Hoc framework has been implemented with Linux OS (Familiar [LINUX]) and integrated in real mobile nodes (Personal Digital Assistants model iPAQ [iPAQ]). Having the framework integrated into Laptops would have been straightforward way for testing the Ad Hoc framework. However, by selecting PDA for integrating the Ad Hoc framework it was included the restrictions that limited devices such the PDA would add to the network behavior.

The original operating system in iPAQ is PocketPC 2002 and it was changed to Linux in order to have the flexibility of adding new components to the kernel, implement and integrate the Ad Hoc framework. Linux supports the ARM architecture that is used by iPAQ and Familiar [LINUX] is a tailored version of Linux OS, which fits into mobile devices with limited resources. However, due to the very little space for the file system in the iPAQ, we cannot install the Linux kernel source code and compile directly into iPAQ. This means we cannot build native executables on iPAQ directly. Therefore, we have to compile everything in a Linux PC to make ARM executables and afterwards transfer them to iPAQ for testing. This required a lot of effort when implementing integrating and testing the Ad Hoc framework in the real nodes.

Nevertheless, the final result is an Ad Hoc framework running on limited nodes that are used for testing the behavior of legacy and new routing algorithms. The Ad Hoc framework was tested in local premises (Figure 3.2) and some results from implemented legacy protocols were obtained.

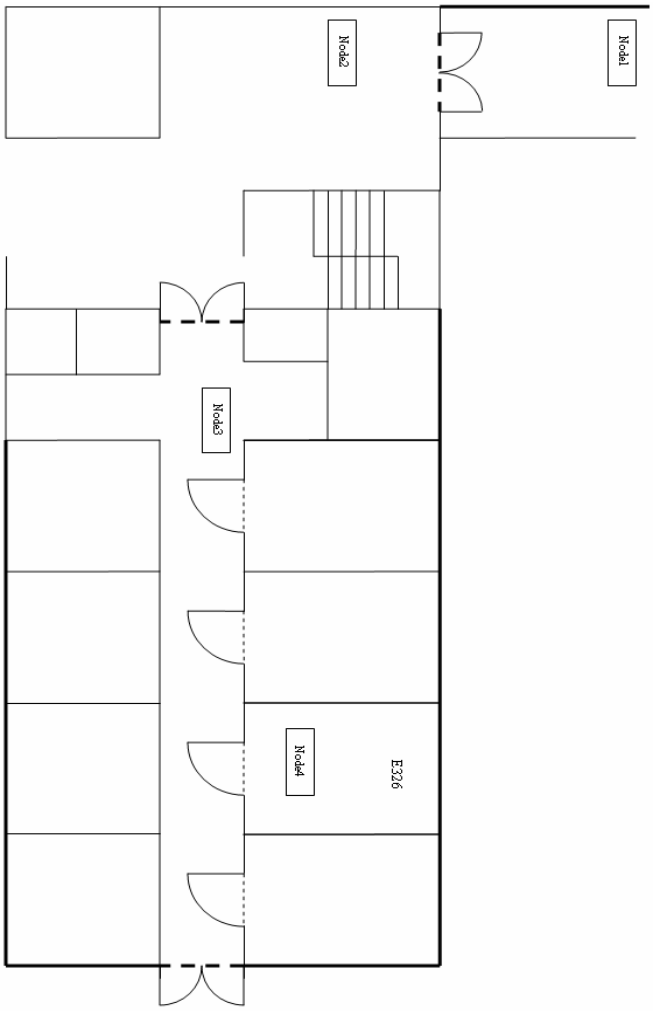


Figure 3.2 Test environment at HUT for Ad Hoc framework on PDA (iPAQ)

Preliminary results from implementing existing MANET IETF routing protocols (AODV) on real nodes showed un-expected behavior that was not found on simulations. There are lots of factors affecting behavior in a real running environment and only the results of these test cases reflect that behavior.

Figure 3.3 shows some of the results from one of the multiple test cases run with 4 iPAQs on the test environment at HUT (Figure 3.2). The results are part of test case 2, where the four iPAQs were located in four different places so that each node was within wireless signal range of only one other iPAQ. The application used for testing was a messaging SIP client where a script was sending repeatedly SIP messages over UDP with variable text content. Figure 3.3 shows that the packet loss is proportional to the packet size. The influence of certain obstacles during the test such as metal doors was creating a lot of link breakages that were reflected with packet loss peaks.

	Test1	Test2	Test3
Test period	3 min	5 min 20 sec	4 min
Number Messages sent	175	316	220
Number of lost packets	50	2	55
Lost packet percentage	28%	12%	25%
Average round trip time	24,5ms	12.7ms	23,5ms
Number of RERR	14	2	10

Legend:

Test1: Test with lots of neighbor link breaks between node 3 and node 4.

Test2: Test with node 3 and node 4 static and node 1 moving freely.

Test3: Test with lots of neighbor link breaks between node 3 and node 4 and LOCAL_REPAIR enabled in order to avoid error messages explosion.

Our results are aligned with findings from other research studies, for example the error messages explosion during a linkage break, gray zones, etc. One of the important findings was that in real environments there are many other factors affecting Ad Hoc networks besides routing algorithms. Some of them come from physical environments and hardware constraints. This means that we have to get some help from lower layers, such as the data link layer and the physical layer in order to predict or accommodate the routing algorithms to certain conditions. It is more and more necessary that other network layers communicate certain changes in order to make Ad Hoc networks scalable.

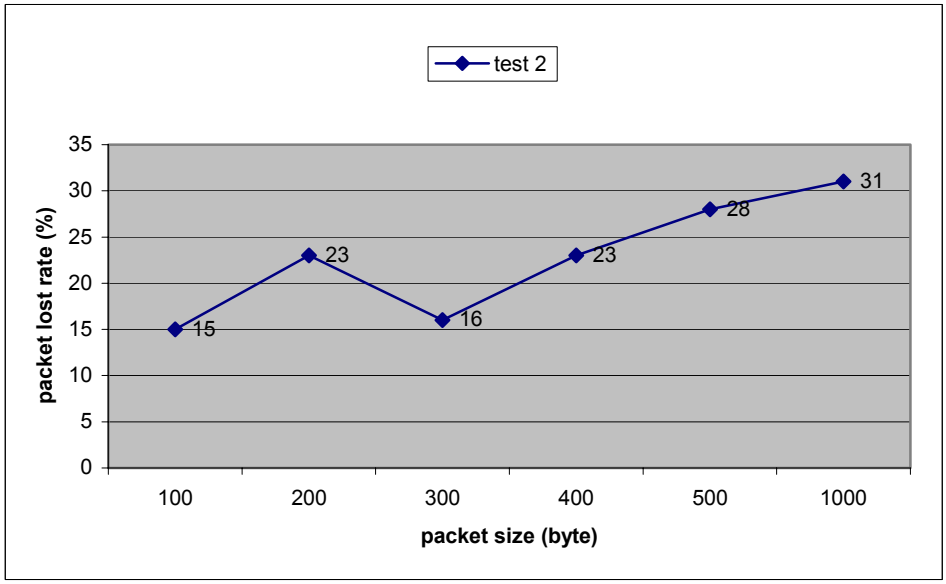


Figure 3.3 packet loss rate for AODV running in the Ad Hoc framework

After having the Ad hoc framework implemented and integrated in real nodes with legacy Ad hoc protocols, the next phases will include new routing algorithms suitable for medium to large networks. This goal requires a set of intermediate steps to verify that we are aiming the right approach. Figure 3.4 shows the different phases on the implementation of the MobileMAN routing architecture towards a suitable routing protocol.

In addition to the implementation of the multiprotocol architecture, a simulative model of the same environment will be developed to support the design of new protocols Most of the actual

simulators do not support the execution of multiple protocols while sharing their routing information as proposed in the "multiprotocol" architecture. Thus, in order to test this proposal the implementation of the "multiprotocol" architecture as new routing model into the simulator is required.

After implementing and integrating the "multiprotocol" architecture based on legacy IETF MANET protocol, the results are analyzed and the output is considered for the design of the novel routing protocols. Novel protocols will be simulated in order to validate the protocol behavior for performance measurements and detecting protocol malfunctioning before its implementation and integration into real nodes. Thus, the simulation is a preliminary step for validating the routing proposal with a huge number of nodes but the real behavior of the protocol will be tested with the prototype using the proposed Ad Hoc framework.

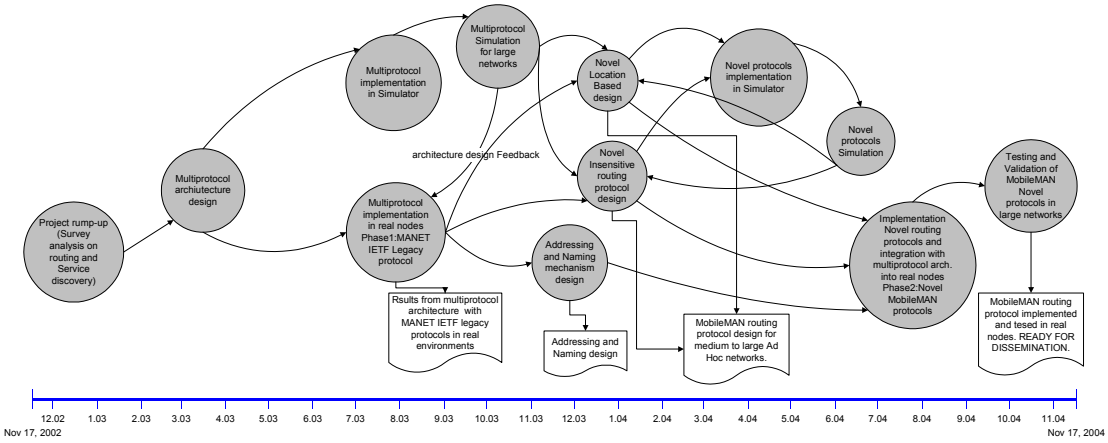


Figure 3.4. Routing protocol design and implementation phases.

3.2.3. Routing in a cross layering architecture

In the current MANET literature, on-demand reactive protocols are considered more efficient than proactive ones [R03]. The claim is that on-demand protocols minimize control overhead and power consumption since routes are only established when required. By contrast, proactive protocols require periodic route updates to keep information current and consistent; in addition, they maintain multiple routes that might never be needed, adding unnecessary routing overheads. We believe that this evaluation is not correct in a cross layering architecture, where protocols overheads cannot be evaluated in isolation but new cross-layer metrics need to be applied. Specifically, as explained before, topology information obtained through the routing protocol can be used at the MAC layer to coordinate the access to the channel among nodes up to a 2-3 hops distance. Furthermore, the knowledge of nodes' location can be used for identifying the closer node that is implementing a service, without requiring the middleware to discover the same location once more. From this perspective, proactive routing protocols that maintain the (at least partial) knowledge of the network topology seem more suitable in the MobileMAN architecture. An important question is therefore which is the cost to proactively maintain the network topology information with respect to an apparently more scalable approach based on reactive routing protocols? A very promising solution to this question has been recently provided by the theoretical analysis presented in [SSR01][SMSR02]. Here, the authors develop an analytical framework to evaluate the protocol scalability taking into consideration, in addition to the proactive and reactive overheads, also the effect introduced by the sub-optimality of routes, accounted for as the additional bandwidth required for using a sub-optimal path. From this perspective, the authors showed that a simple Link State protocol with partial dissemination (information is propagated to the network nodes with a frequency that decreases with the distance) scales better than more complex hierarchical protocols and hence this class of protocols can be an efficient routing

alternative for large-scale ad hoc networks. This is a very important result for the MobileMAN project as it indicates that link-state routing strategies based on limited dissemination of state information not only provide several qualitative advantages when used in a cross-layer architecture, but also provide effective solutions from a quantitative standpoint. Even though this result is apparently contra-intuitive, it can be explained by observing that nodes that are far away do not need to have precise topological information to make a good next hop decision. As pointed out in [BCSW98], the inaccuracy in the topological information is balanced by the *distance effect*: "the greater the distance separating two nodes, the slower they appear to be moving with respect to each other". Hence, the required accuracy of the location information decreases with the distance from the node. By exploiting these results, in the framework of the MobileMAN project we are currently investigating the use of link-state protocols based on a FSR-like approach [PGC00]. The FSR protocol [PGC00][KS71] is an optimization over link-state algorithms using fisheye technique. In essence, FSR propagates link state information to other nodes in the network based on how far away (defined by scopes which are determined by number of hops) the nodes are. The protocol will propagate link state information more frequently to nodes that are in a closer scope, as opposed to ones that are further away. This means that a route will be less accurate the further away the node is, but once the message gets closer to the destination, the accuracy increases. Therefore, each node has its own view of the network topology that uses to forward packets towards the last known direction of the destination node. If a node is far from the destination the inaccuracy of its topology information may be large but it is compensated by two factors: the inaccuracy must be normalized with respect to the distance, and while a packet distance from the destination decreases, the forwarding nodes have a more and more accurate knowledge of the current location of the destination, and hence they will forward the packet along the optimal path.

3.3. Reliable Forwarding

The aim of reliable forwarding is to select source-destination paths taking into consideration several aspects: nodes cooperation, congested links at MAC layer (cross layering with MAC), BER, congested nodes, etc. As it provides a tool for avoiding routes containing misbehaving and/or selfish nodes, it is synergic with cooperation enforcing mechanism. According to our approach a node is responsible not only for forwarding a packet, but it shall forward it on the route that maximizes its success probability.

The mechanisms we designed for MobileMAN reliable forwarding are customized for the MobileMAN *cross layering architecture.* Our mechanism of reliable forwarding is positioned at network layer, and exploits routing information, as well as transport layer packet acknowledgements. Specifically, our reliable forwarding assumes that i) TCP acknowledgments are available also to the other layers in the protocols' stack; ii) a pro-active link-state routing protocol (with partial information dissemination) is used.

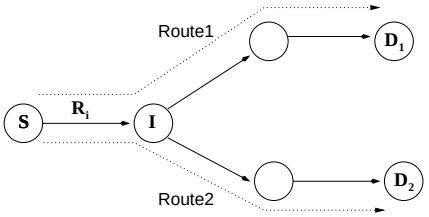
The first assumption is necessary to evaluate the routes reliability. Our system is based on the principle that we can trust only ourselves. Specifically, the proposed mechanism is distributed, but not cooperative, and based on nodes internal knowledge. Every node acts independently, without sharing any information with other nodes, and trusts only information coming from the other communication peer (communication between peers may eventually be encrypted to avoid forged acknowledgements by intermediate malicious nodes) through TCP ACKs.

The primary consequence of a link-state routing protocol with limited dissemination of link-state information is that a node has a precise knowledge of the 2-3 hops network topology around it, a partial knowledge of the remaining network topology, including the possible multiple paths to a destination.

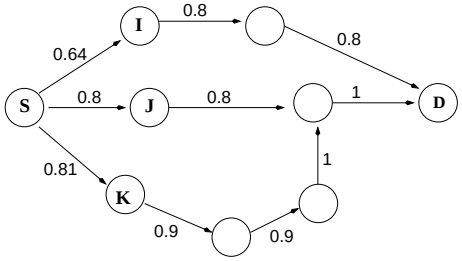
The above features are used for implementing a (multi-path) reliable forwarding mechanism that fully exploits the cross layering principle through a full loop between network (routing,

forwarding), transport and cooperation/performability mechanisms [CGM03]. The reliable forwarding basic idea is to deliver data to a destination by utilizing existing paths according to performance and reliability criteria summarized by a *performability index*. More precisely, the network layer through the routing mechanism has (partial) knowledge of the network topology. This may include the alternative routes for a given destination, and other info on these routes (e.g., number of hops). The Cooperation/Performability function, by using the TCP ACKs, classifies the reliability/performance/cooperation along these routes, and computes the *performability* index for each route. This index classifies the paths taking into account several factors (congestion, links quality, selfish nodes, etc.) that may influence the system performance. The Forwarding function exploits the performability index to select among alternative paths, and to perform a load balancing among the routes. Again, transport ACKs are used by the Cooperation/Performability function to measure the *performability* of the selected routes, and hence to redefine the related metric. Finally, a new *performability* measure is provided to the Forwarding function that will use it for selecting the future paths for packets forwarding. This closes the loop.

More precisely, every node has a dynamically updated performability table containing a value for every outgoing link to a neighbor. Such a value represents a performability index for paths rooted at that neighbor. Every time the node sends a packet on a path, it updates the performability value associated to the neighbor through whom the packet has passed: the updating is positive whenever source node receives an acknowledgement from destination, negative otherwise. The performability value is unique for all paths rooted on that neighbor (see Figure 3.5(a)). If source node observes that the performability index of that sub-tree decreases, then it should immediately reduce the traffic sent through that neighbor, by preferring routes passing through a neighbor with a higher performability index. Figure 3.5(b) shows an example: source node S has three possible routes to send a packet to destination node D. Each route passes through a different neighbor (I, J, K), and each link to a neighbor has a performability index[15]. By comparing such values, source S finds out that path through K is the better one, even if the longer in terms of hops number. Source S may decide to take that path to maximize the success probability of the packet forwarding.



(a) Every time node S sends a packet on Route 1 or Route 2, it updates index R_i that is associated to neighbor I. Thus R_i indicates the performability level of the network subtree rooted at I.



(b) The graph shows for each node, the performability indices associated to its neighbors. Node S has three possible routes to reach destination D and can choose one of them according to the performability index associated to its neighbors (route through K seems the most reliable).

Figure 3.5: Performability index

Hereafter, we briefly present the preliminary version of the reliable forwarding mechanism, see [CGM03] for details.

We model a network as a graph G = (V, L), where V is a set of mobile nodes and L is a set of direct links. Each node i Є V has a unique node identifier (ID). A link (i, j) Є L represents a connection between the two nodes i and j, meaning that j is in the transmitting range of i, and vice

[15] Node S does not have the knowledge of the reliability of all the links, but it has a reliability index for each subtree rooted on its neighbors. This index summarizes the reliability of all the links crossed by the S-D path.

versa. In that case, nodes i, j are said adjacent (or neighbors), and we call N(i), neighbor set of i, the set of nodes adjacent to a given node i:

$$N(i) = \{j \mid j \in V \wedge (i, j) \in L\}$$

Given any node i Є V, for each j Є N(i) we have a probability value R_j that represents the performability level of link (i, j). Probability R_j is dynamically updated every time node i sends a packet on link (i, j) and represents a performability measure for paths rooted at neighbor j: it increases if the sent packet reaches the destination, it decreases otherwise. We suppose to have an end-to-end notification acknowledgement on packet delivery: if node *s* is the source node, node *d* the destination, with an arbitrary number *n* of hops between *s* and *d* (*n > 0*), when destination node *d* receives the packet, it sends back to *s* an ack message. If *s* does not receive any acknowledgement before a specified timeout, then we assume the packet did not reach destination, and some node on the path did not forward it. For each packet sent, we denote with *M* the result of the packet delivery and we estimate a smoothed performability value R_j using a low-pass filter, with the same approach used in the TCP protocol for Round-Trip Time measurement:

$$R_i \leftarrow \alpha \cdot R_i + (1-\alpha)M$$

where α, 0≤ α ≤ 1, is a smoothing factor and represents the percentage of the previous estimate considered on each new estimate. If α =0.9, then ninety percent of the new estimate is from the previous estimate, and 10% is from the new measurement.

M is the result of a packet delivery process from s to d, and it can assume the following values:

$$M = \begin{cases} 0 & \textit{if } s \textit{ does not receive ack from } d \\ 1 & \textit{if } s \quad \textit{receives ack from } d \end{cases}$$

If packet does not reach destination, then the performability on outgoing link of source node decreases by an α factor. If packet reaches destination then nodes are cooperating and performability on outgoing link of source node is smoothed by an α factor and increased by (1 - α).

In the following, we show how performability indices are used to control traffic forwarding.
In case of multiple routes available for packet forwarding to a destination node, source node can choose one of them according to a certain principle. Routing protocols for ad hoc networks usually choose the shorter one, in terms of hops number, or the fresher one in terms of discovering time. Such criteria do not take into account links performability. In [CGM03], we proposed and analyzed two route selection policies dealing with performability values associated to outgoing links, and we investigated their effectiveness.

Policy-1 Source node takes always the most reliable route. In such a case, source node compares performability values for available routes and forwards packets on the link with the greatest value. This policy assures source node of taking always the most reliable route. The main drawback of such a choice is the deviation of all traffic on most reliable links which, in case of high traffic load, can quickly get congested.

Policy-2 This policy relates performability values of available routes to build a probabilistic scheme. Let us suppose we have several possible routes to a destination through different source's neighbor nodes, $i_1, i_2, \ldots, i_n$. Each neighbor has its respective performability value $R_{i_1}, R_{i_2}, \ldots, R_{i_n}$. We associate a probabilistic value to each of such neighbors, $p_{i_j}, 1 \le j \le n$, defined in the following way:

$$p_{i_j} = \frac{R_{i_j}}{\sum_{K=1}^{n} R_{i_k}}$$

(3.1)

Equation (3.1) relates *performability* values so that the resulting probabilistic value reflects the link performability level. Routes are chosen according to the probabilistic value associated to the first node on the path: the greater the probability, the higher the route selection frequency. This probabilistic policy allows nodes to take even less reliable routes: traffic forwarding function is better distributed on all available routes and links congestion becomes rarer.

Simulations have shown that performability-based policies significantly improve network performance (mainly the throughput) in packet forwarding [CGM03]. However, when congestion occurs we need to extend our policies to distinguish low performability due to BER, selfishness, malicious behavior, etc. from packet losses caused by congestion conditions. Work is ongoing to enhance the path selection policies and the performability index definition.

3.4. Transport protocol

TCP is an effective connection-oriented transport control protocol that provides the essential flow control and congestion control required to ensure reliable packet delivery [S94]. TCP was originally designed to work in fixed networks. Because error rate in wired network is quite low, TCP uses packet losses as an indication for network congestion, and deals with this effectively by making corresponding adjustment to its congestion window. Numerous enhancements and optimizations have been proposed over the past few years to improve TCP performance for infrastructure-based WLANs, and cellular networking environments, see e.g., [BB97][BSAK95][BPSK96][BS97]. The issues and solutions for using TCP over mobile networks are surveyed in [HE02].

Infrastructure-based wireless networks are 1-hop wireless networks where a mobile device uses the wireless medium to access the fixed infrastructure (e.g., the access point). Although there are a number of differences between infrastructure and ad hoc networks, many of these proposed solutions can be exploited also in the mobile ad hoc networks, mainly when a MANET needs to be interconnected to the Internet.

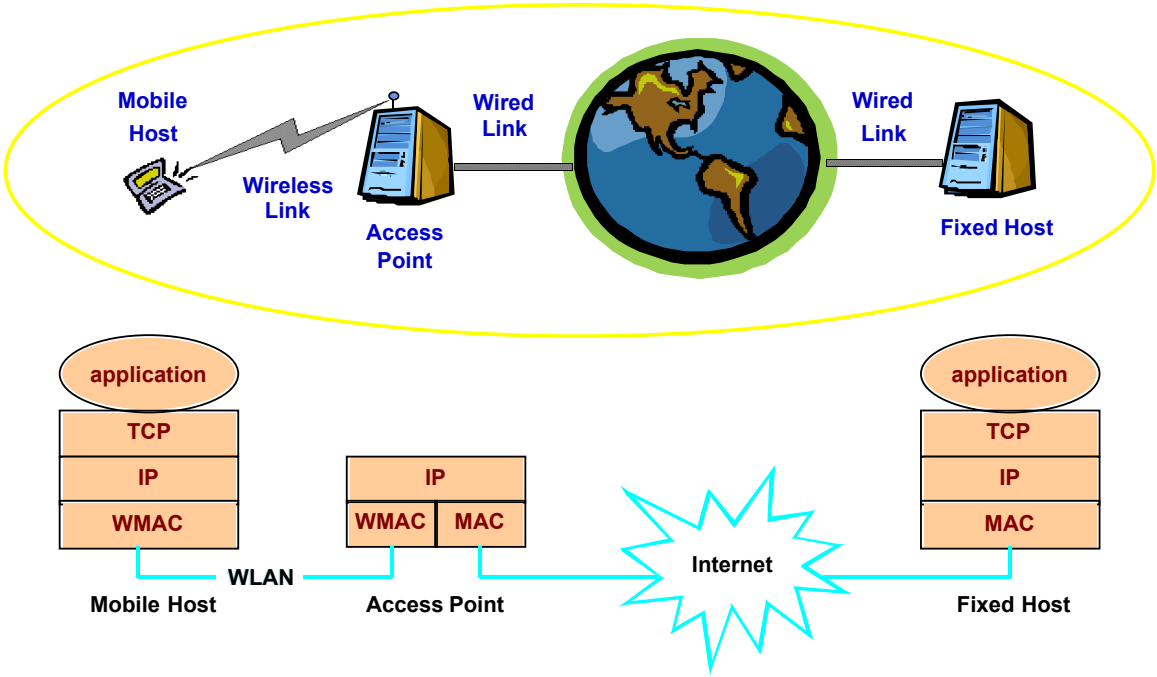


Figure 3.6: Legacy wireless access to Internet

Figure 3.6 shows a typical scenario for 1-hop wireless access to the Internet, and the corresponding architecture and protocols. The communication between a mobile host and a machine connected to the Internet (Fixed Host) is made possible by a third entity (Access Point), which provides Internet connectivity to the mobile host through a wireless link. Although very simple and costless, a legacy TCP-based solution is prone to various drawbacks that heavily impact the system performance. Performance degradation is mainly due to the differences existing between the wired and wireless part of the network, and the impact these differences have on TCP congestion control mechanisms. In detail:

1. The TCP congestion control wrongly interprets losses in the wireless link as congestion signals. Hence, the overall throughput is usually low and the wireless network interface at the mobile host remains idle for most of the time.
2. Congestions in the wired networks limit the throughput in the wireless link as well. The overall effect is the same as in 1.

As these problems are due to the differences existing between the two networking environments a promising direction to overcome them is based on the Indirect-TCP model [BB97]. As shown, in Figure 3.7, the transport connection between the client at the mobile host and the Web server is split into two parts: the first one between the mobile host and the Access Point and the second one between the Access Point and the Web server. At the Access Point an agent (I-TCP daemon) relays data from one connection to another.

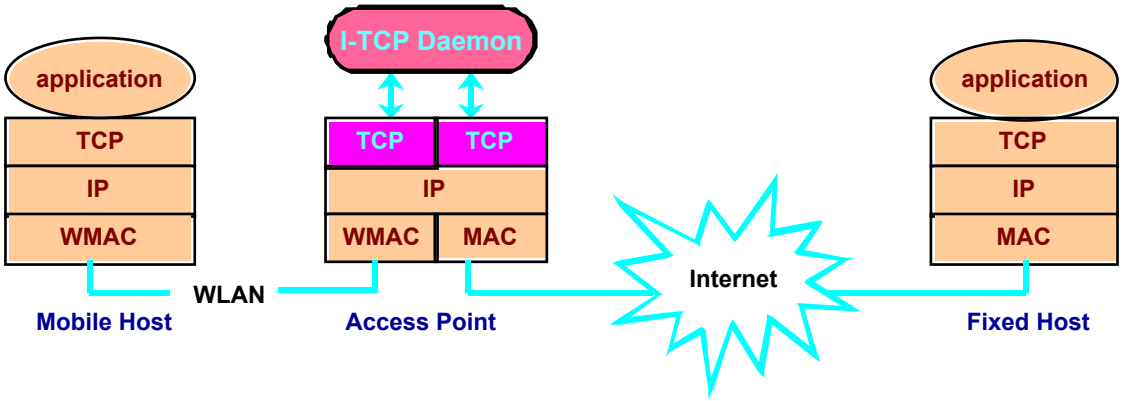


Figure 3.7: The I-TCP model

In the I-TCP model using the legacy TCP protocol on the wireless link is not mandatory. Indeed several works have shown that significant performance advantages can be achieved by using on the wireless part (i.e., on the mobile host and on the part of the access point directly connected with it) a transport protocol tailored to the wireless links characteristics (see, e.g., [ACGP03] and references herein). The mobile multi-hop ad hoc environment brings additional reasons to highly modify the TCP protocol to be used in ad hoc networks [ACGP03]. Hence, a MANET needs to be interconnected to the Internet using the I-TCP model is a promising direction. Specifically, as we discussed in Section 1, we envisage a MANET as a network technology that can be interconnected to the Internet via a proxy. Hence, using the legacy TCP protocol (in the MANET part) is not mandatory, and we decided to explore the performance advantages that can be achieved by using a transport protocol specifically tailored to the MobileMAN architecture. The new protocol needs to be designed to fix the problems emerging when using TCP in MANETs. Specifically, the dynamic topologies, and the interaction of MAC protocol mechanisms (e.g., 802.11 exponential back-off scheme) with TCP mechanisms (congestion control and time-out) lead in a multi-hop environment to new and unexpected phenomena. A survey on TCP research in MANET can be found in [ACG03]. Hereafter, we summarize the main research areas, and the open issues.

IMPACT OF MOBILITY. In a MANET, nodes' mobility may have a severe impact on the performance of the TCP protocol [HV99][HV02][TG99][AA00][DB01]. Mobility may cause route failures, and hence, packet losses and increased delays. The TCP misinterprets these losses as congestion, and invokes the congestion control mechanism, potentially leading to unnecessary transmissions (during routes' reconstruction), and throughput degradation [HV02][CRVP01]. In addition, the stations' mobility may exacerbate the unfairness between competitive TCP sessions [TG99]. The performance of the TCP protocol when running (among others) over DSR and AODV are analyzed in [HV99][HV02] [AA00][DB01]. These results point out the route failure frequency as an important factor in determining TCP throughput in ad hoc networks.

NODES' INTERACTION AT MAC LAYER. Even when stations are static, the performance of an ad hoc network may be quite far from ideal, as the performances are strongly limited by the interaction between neighboring stations. A station activity is limited by the activity of neighboring stations inside the same TX_Range, IF_Range or PCS_Range, and by the interference caused by hidden and exposed stations. For example, in a chain topology stations early in the chain may cause starvation of later stations. Similar considerations apply to other network topologies. In general, the 802.11 MAC protocol appears to be more efficient in case of local traffic patterns, i.e., when the destination is close to the sender [BCLM02].

IMPACT OF TCP CONGESTION WINDOW SIZE. TCP congestion window size may have a significant impact on performance. In [FZX03], the authors show that, for a given network topology and traffic patterns, there exists an optimal value of the TCP congestion window size at which channel utilization is maximized. However, TCP does not operate around this optimal point, but typically with a window that is much larger, leading to decreased throughput (10-30% throughput degradation), and increased packet loss. These losses are due to link-layer drops: a station fails to reach its adjacent station due to the contention/interference of other stations. By increasing the congestion window size, the number of packets in the pipe between the sender and the receiver is increased, and hence the contention at the link-level increases, as well. Small congestion windows (i.e., 1-3 packets) typically provide the best performance [XS01][XS02].

INTERACTION BETWEEN MAC PROTOCOL AND TCP. The interaction of the 802.11 MAC protocol with the TCP protocol mechanisms may lead to unexpected phenomena in a multi-hop environment. For example, in the case of simultaneous TCP flows, severe unfairness problems and - in extreme cases - capture of the channel by few flows may occur. Furthermore, instantaneous TCP throughput may be very unstable also with a single TCP connection. These phenomena can be reduced/exacerbated by using small/large TCP-congestion window. These problems have been revealed in [XS01][XS02]. Recently, similar phenomena have been also observed in other scenarios [KBLG02]. Such phenomena do not appear, or appear with less intensity, when the UDP protocol is used [XG02].

Numerous new mechanisms for TCP optimization have also been proposed with the aim of resolving MANET specific issues, including adaptation of TCP error-detection and recovery strategies to the ad hoc environment. To minimize the impact of mobility and link disconnection on TCP performance, [CRVP01] proposed to introduce explicit signaling (Route Failure and Route Re-establishment notifications) from intermediate nodes to notify the sender TCP of the disruption of the current route, and construction of a new one. In this way, TCP after a link failure does not activates the congestion avoidance mechanisms, but simply freezes its status that will be resumed when a new route is found. In [HV02][HV99] an Explicit Link Failure Notification (ELFN) mechanism is introduced. The ELFN objective is to provide (through ELFN messages) the TCP at the sender-side explicit indications about link and route failures. In this case there is no explicit signaling about route reconstruction. [MSB00] presents a simulation study of ELFN mechanism,

both in static and dynamic scenarios. This study points out limitations of this approach that are intrinsic to TCP properties (e.g., long recovery time after a timeout), and proposes to implement mechanisms below the TCP layer. This is also the approach proposed and implemented in [LS01]. In this work, the standard TCP is unmodified, while new mechanisms are implemented in a new thin layer, ad hoc TCP (ATCP), between TCP and IP. This layer uses ECN messages and ICMP "destination unreachable" packets to distinguish congestion conditions from link failures, and from losses on the wireless links. According to type of event, ATCP takes the appropriate actions. Previous techniques require explicit notification by intermediate nodes to the sender. To avoid this complexity, [WZ02] proposes to infer at the TCP level route changes by observing the out-of-order delivery events that are frequently introduced by a route change.

In [FZX03], the authors focus on static multi-hop networks and provide a solution to fix TCP performance problems caused by MAC-TCP interactions (nodes' interaction at MAC layer plus TCP congestion window size). The basic observation here is that in multi-hop networks the channel utilization is associated to the spatial channel reuse. Spatial reuse defines, given network topology, nodes that may concurrently transmit without interfering with each other. For a given flow and network topology, there exists a contention-window that achieves the best channel reuse, thus providing the maximum throughput. However, legacy TCP operates with a window larger then the optimal one, and hence with a reduced throughput. To address this problem, two link level mechanisms have been proposed [FZX03]: Link RED, and adaptive spacing. Similarly to the RED mechanism implemented in Internet routers, the Link RED tunes the drop probability at the link level by marking/discarding packet according to the average number of retries experienced in the transmission of previous packets. The Link RED thus provides TCP with an early sign of overload at link level. Adaptive spacing is introduced to improve spatial channel reuse, thus reducing the risk of stations' starvation. The idea here is the introduction of extra backoff intervals to mitigate the exposed receiver problems. Adaptive spacing is complementary to Link RED: it is activated only when the average number of retries experienced in previous transmission is below a given threshold.

Hereafter, we present preliminary ideas about a transport protocol, named *Transport Protocol for Ad hoc networks* (*TPA*), which is currently under development in MobileMAN. We started designing this protocol with the aim of i) maximizing the performance of our MANET environment, and ii) guaranteeing an easily integration in the cross-layer architecture. For this reason we not designed it as a modification of the TCP, but as a new protocol. This enabled us to insert in the new protocol mechanisms tailored to the characteristics of the MANET environment. [AP03] provides a preliminary presentation of TPA basic ideas and of the mechanisms required to implement them. An important result is emerging from this first phase of the study: the basic mechanisms of TPA can be implemented by modifying the transport protocol at the sender side only, while TPA is still able to interoperate with legacy TCP receivers. This is an important feature as this means that the new protocol can operate in ad hoc networks were some nodes are using the legacy TCP protocol.

The TPA protocol provides a reliable, connection-oriented type of service (the set up and tear down phases are similar to the TCP protocol). A *Selective ACK* like mechanism is used to manage packets acknowledgments. TPA tries to guarantee a reliable service by reducing as much as possible the number of useless retransmissions. This is extremely important since retransmissions consume energy. To this end, taking into consideration that in a highly dynamic environment the round trip time (RTT) may be inaccurate, packets are not immediately retransmitted on the timer expiration. Indeed packets transmissions are managed in blocks of up to K packets. All the packets in a block are transmitted first before considering the re-transmission of packets for which no ACKs are received. When all packets in a block are transmitted, the re-transmission of un-acknowledged packets starts, and so on.

It is worth noting that the re-transmission policy we introduced in TPA makes it resilient against ACK losses because a single Selective ACK is sufficient to notify the sender about all missed

packets in the current block. In addition, the sender does not suffer from the out-of-order arrivals of packets. This implies that the TPA can operate efficiently also in MANETs using multi-path forwarding, where, on the contrary, the TCP typically performs very poorly [LS01]. Both features (selective ACKs and multi-path support) are important for the Reliable Forwarding policy presented in the previous section.

TPA is designed to manage efficiently route changes and route failures. To efficiently manage route failures an Explicit Link Failure Notification (ELFN) is assumed. In our cross layering architecture, in which we use a link state with partial dissemination, the ELFN signal can be part of the routing protocol.

When a route change occurs, TPA utilizes an effective way to tune the TCP retransmission timer to the new route. This is achieved by increasing (in the estimate of the round trip time) the weight of the new RTT samples computed just after a route change.

TPA flow control is similar to that used by TCP, while it includes a completely re-designed congestion control mechanism. The TPA congestion control mechanism is window-based as in the TCP protocol. However, by exploiting results shown in [FZX03], in the TPA the maximum congestion window size is very small (3-4 TPA packets[16]), and the TPA congestion control algorithm is very simple. When the TPA is not in the congested state, the congestion window is set to the maximum value. On the other hand, during the congested state, the congestion window is reduced to 1 to allow congestion to disappear. A congested state is assumed after the expiration of *th_cong* consecutive timeouts without receiving an ELFN signal. The end of the congestion period is assumed upon the arrival of *th_ack* consecutive ACKs. Both *th_cong* and *th_ack* are protocol parameters to be tuned according to simulative and experimental results).

We have developed a TPA simulation model in the QualNet environment. Simulations are currently ongoing to tune of the protocol parameters and evaluate the protocol performance.

The retransmission policy we used in TPA may increase the packets delay as the retransmission of a packet can start only after transmitting all the packets in its block. However, this does not degrade the system throughput as the sender is continuously transmitting packets if the congestion window is open. For environments where delay is an important quality of service parameter, we have planned to extend the TPA protocol with a FEC-based mechanism based on erasure codes [R97] that encode the bytes to be transmitted in n blocks. The correct delivery of k out of n ($k<n$) blocks is sufficient to guarantee a reliable service. In our case, according to the cross-layer principle, the redundancy level to be used for encoding the stream of bytes (i.e., the ratio between n and k) will be defined by TPA taking into consideration the path(s) reliability index as estimated by the reliable forwarding mechanism.

[16] Simulations are ongoing to exactly estimate this parameter value.

3.5. References

[AA00] A. Ahuja et al., "Performance of TCP over different routing protocols in mobile ad-hoc networks," Proceedings of IEEE Vehicular Technology Conference (VTC 2000), Tokyo, Japan, May 2000.

[ACG03] G. Anastasi, M. Conti, E. Gregori, "IEEE 802.11 Ad Hoc Networks: Protocols, Performance and Open Issues", *Ad hoc networking*, S. Basagni, M. Conti, S. Giordano, I. Stojmenovic (Editors), IEEE Press and John Wiley and Sons, Inc., New York, 2003.

[ACGP03] G. Anastasi, M. Conti, E. Gregori, A. Passarella, "Balancing energy saving and QoS in the mobile internet: an application-independent approach", Proceedings of the 36th Hawaii International Conference on System Sciences, (HICSS-36), 2003, pp. 305 –314.

[AP03] G. Anastasi, A. Passarella "Towards a Novel Transport Protocol for Ad Hoc Networks" Proc. PWC2003, 23-25 September 2003, LNCS 2775, pp. 795-800.

[APE02] APE: Ad hoc Protocol Evaluation testbed. Department of Computer Systems at Uppsala, Sweden. http://apetestbed.sourceforge.net/

[BB97] A. V. Bakre and B. R. Badrinath, "Implementation and performance evaluation of indirect TCP," IEEE Trans. Computers, vol. 46, March 1997.

[BCLM02] J. Li, C. Blake, D. De Couto, H. Lee, R. Morris, "Capacity of Wireless Ad Hoc Wireless Networks", Proceedings of The Seventh ACM International Conference on Mobile computing and networking (MOBICOM 2001), July 16-21, 2001, Rome, Italy. pp. 61-69.

[BCSW98] S. Basagni, I. Chlamtac, V. Syrotiuk, and B. Woodward, "A Distance Routing Effect Algorithm for Mobility (DREAM)," in Proceedings of The Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM '98), October 25-30, 1998, Dallas, Texas, USA.

[BMJ00] Josh Broch, David A. Maltz, David B. Johnson, "Quantitative Lessons From a Full-Scale Multi-Hop Wireless Ad Hoc Network Testbed", Proceedings of the IEEE Wireless Communications and Network Conference 2000 (WCNC 2000).

[BCSW98] S. Basagni, I. Chlamtac, V. Syrotiuk, and B. Woodward, "A Distance Routing Effect Algorithm for Mobility (DREAM)," in Proceedings of The Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM '98), October 25-30, 1998, Dallas, Texas, USA.

[BOT01] B. Bellur, R. G. Ogier, F. L. Templin, "Topology Broadcast Based on Reverse-Path Forwarding (TBRPF)", IETF Internet Draft, draft-ietf-manet-tbrpf-01.txt, March 2001.

[BPSK96] H. Balakrishnan, V. Padmanabhan, S. Seshan, and R. Katz, "A comparison of mechanisms for improving TCP performance over wireless links," Proc. ACM SIGCOMM, Stanford, CA, August 1996.

[BSAK95] H. Balakrishnan, S. Seshan, E. Amir, R. Katz, "Improving TCP/IP Performance over Wireless Networks", in Proceedings of the First Annual International Conference on Mobile Computing and Networking (MOBICOM '95), November 13-15, 1995, Berkeley, CA.

[BS97] K. Brown and S. Singh. M-TCP: TCP for Mobile cellular Networks, in Proceedings of the ACM SIGCOMM Computer Communication Review, 1997, pp. 19-43.

[CCL03] I. Chlamtac, M. Conti, and J. Liu, "Mobile Ad Hoc Networking: Imperatives and Challenges", *Ad Hoc Networks*, No. 1(1) 2003.

[CGM03] M. Conti, E. Gregori, G. Maselli. Towards Reliable Forwarding for Ad Hoc Networks. Proc. Personal Wireless Communications Conference 2003 (PWC 2003) Sept. 2003, LNCS 2775, pp. 780-794.

[CGT03] M. Conti, E. Gregori, G., "Design and analysis of a context-aware location service for ad hoc networks" Proc. First International Working Conference on Performance Modelling and Evaluation of Hetrogeneous Networks (HET-NETs '03), 21-23 July 2003, Ilkley, West Yorkshire, U.K.

[CKN01] J. Costa-Requena, R. Kantola, N. Beijar ," Replication of Routing Tables for Mobility Management in Ad Hoc Networks", ACM Wireless Networks (WINET), 2003.

[CRVP01] K. Chandran, S. Raghunathan, S. Venkatesan, R. Prakash, "A Feedback Based Scheme for Improving TCP Performance in Ad Hoc Wireless Networks", *IEEE Personal Communication Magazine*, Special Issue on Ad Hoc Networks, Vol. 8, N. 1, pp. 34-39, February 2001.

[DB01] T.D. Dyer, R.V. Boppana "A Comparison of TCP Performance over Three Routing Protocols for Mobile Ad Hoc Networks", Proc. ACM Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc), October 2001.

[ER03] Elizabeth Belding-Royer, "Routing approaches in Mobile Ad Hoc Networks", in *Mobile Ad Hoc Networking*, S. Basagni, M. Conti, S. Giordano, I. Stojmenovic (Editors), IEEE Press and John Wiley and Sons, Inc., New York, 2003.

[FZX03] Zhenghua Fu, Petros Zerfos, Kaixin Xu, Haiyun Luo, Songwu Lu, Lixia Zhang, Mario Gerla, "The Impact of Multihop Wireless Channel on TCP Throughput and Loss", Proc. Infocom 2003, San Francisco, April 2003.

[GH99] S. Giordano, M. Hamdi, "Mobility Management: The Virtual Home Region", Technical Report No. SSC/1999/037, EPFL, October 1999, http://www.terminodes.org

[HE02] H. Elaarag, "Improving TCP Performance over Mobile Networks", ACM Computing Surveys, Vol. 34, No. 3, September 2002, pp. 357–374.

[HV02] Gavin Holland, Nitin H. Vaidya "Analysis of TCP Performance over Mobile Ad Hoc Networks", ACM/Kluwer Journal of Wireless Networks 8(2-3), (2002) pp. 275-288.

[HV99] G. Holland, N. H. Vaidya, "Analysis of TCP performance over mobile ad hoc networks," Proceedings of The Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM '99), August 15-19, 1999, Seattle, Washington, USA. pp. 219-230.

[iPAQ] HP handheld devices (http://welcome.hp.com/country/us/eng/prodserv/handheld.html)

[IS99] Ivan Stojmenovic, "Home Agent Based Location Update and Destination Search Schemes in ad hoc wireless networks", Technical Report TR99-10, Computer Science, SITE, University of Ottawa, Canada, September 1999.

[JM96] D. B. Johnson, D. A. Maltz, "Dynamic Source Routing in Ad-Hoc Wireless Networks", Mobile Computing, T. Imielinski and H. Korth (eds.), Kluwer Academic Publishers, pp. 153--181, 1996.

[JMQ98] P. Jacquet, P. Muhlethaler, A. Qayyum, "Optimized Link State Routing Protocol", *Internet Draft,* draft-ietf-manet-olsr-00.txt, November 1998.

[KBLG02] K. Xu, S. Bae, S. Lee, M. Gerla, "TCP Behavior across Multihop Wireless Networks and the Wired Networks", Proceedings of the *ACM Workshop on Mobile Multimedia (WoWMoM 2002),* Atlanta (GA), September 28, 2002, pp. 41-48.

[KS71] L. Kleinrock, K. Stevens, "Fisheye: A Lenslike Computer Display Transformation", Technical Report, UCLA, Computer Science Department, 1971.

[LC00] Y. Bing Lin and I. Chlamtac, *Wireless and Mobile Network Architectures*, John Wiley & Sons, October 2000.

[LJD99] Jinyang Li, John Jannotti, Douglas S. J. De Couto, David R. Karger, Robert Morris, "A Scalable Location Service for Geographic Ad Hoc Routing" in Proceedings of The sixth ACM International Conference on Mobile Computing and Networking (MOBICOM 2000), August 6-11, 2000, Boston, MA, USA.

[LINUX] Familiar homepage, http://www.handhelds.org, March 2003.

[LLNN01] H. Lundgren, D. Lundberg, J. Nielsen, E. Nordström, C. Tschudin, "A Large-scale Testbed for Reproducible Ad hoc Protocol Evaluations", Proc. WCNC 2002.

[LNT02] H. Lundgren, E. Nordstron, C. Tschudin, "Coping with Communication Gray Zones in IEEE 802.11 based Ad Hoc Networks", Proceedings of the *ACM Workshop on Mobile Multimedia (WoWMoM 2002)*, Atlanta (GA), September 28, 2002, pp. 49-55.

[LNT03] H. Lundgren, E. Nordstron, C. Tschudin, "The Gray Zone Problem in IEEE 802.11b based Ad hoc Networks, ACM SIGMOBILE Mobile Computing and Communications Review 2002.

[LS01] J. Liu, S. Singh, "ATCP: TCP for mobile ad hoc networks", IEEE Journal on Selected Areas in Communications, 19(7):1300-1315, July 2001.

[MAN02] MANET Meeting Report at 55th IETF Meeting in Altanta, Georgia USA http://www.ietf.org/proceedings/02nov/177.htm

[MBS00] J. Monks, P. Sinha, V. Bharghavan, "Limitations of TCP-ELFN for Ad hoc Networks", Proc MoMuc 2000, Tokyo, Japan, October 2000.

[MMCG01] D. Meddour,B. Mathieu, Y. Carlinet, Y. Gourhant, "Requirements and enabling achitecture for Ad-Hoc networks application Scenarios", Workshop on Mobile Ad Hoc Networking and Computing (MADNET 2003), March 2003, Sophia-Antipolis, France.

[NTCS99] Sze-Yao Ni, Yu-Chee Tseng, Yuh-Shyan Chen, Jang-Ping Sheu, "The broadcast storm problem in a mobile ad hoc network", Proceedings of The Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM '99), August 15-19, 1999, Seattle, Washington, USA. pp. 151-162.

[P00] C.E. Perkins, "Ad Hoc Networking", Addison-Wesley, Reading, MA, 2000

[PGC00] G. Pei, M. Gerla, and T.W. Chen, "Fisheye State Routing in Mobile Ad Hoc Networks", Proceedings of the 2000 ICDCS Workshops, Taipei, Taiwan, Apr. 2000.

[PGH00] G. Pei, M. Gerla, X. Hong, "LANMAR: Landmark Routing for Large Scale Wireless Ad Hoc Networks with Group Mobility", *Proceedings of IEEE/ACM MobiHOC 200*0, pp. 11-18, Boston, MA, August 2000.

[PH01] Pai-Hsiang Hsiao, "Geographical region summary service for geographical routing", Mobile Computing and Communications Review, Volume 5, Number 4, October 2001.

[PK] Petteri Kuosmanen, "Classification of Ad Hoc Routing Protocols", Finnish Defence Forces, Naval Academy, Finland, http://keskus.hut.fi/opetus/s38030/k02/Papers/12-Petteri.pdf

[PR99] C. E. Perkins, E. M. Royer, "Ad-hoc On-Demand Distance Vector Routing," Proceedings of 2nd IEEE Workshop on Mobile Computing Systems and Applications, February 1999.

[R97] L. Rizzo, "Effective erasure codes for reliable computer communication protocols", ACM SIGCOMM Computer Communication Review, Volume 27 , Issue 2 (April 1997), Pages: 24 – 36.

[RT99] E.M. Belding-Royer, C.-K. Toh, "A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks", IEEE Personal Communications Magazine, April 1999, pp. 46-55.

[S94] W.R. Stevens, *TCP/IP Illustrated, Vol. 1, The Protocol,* Addison-Wesley, Reading, Massachusetts, 1994.

[SMSR02] Cesar A. Santivanez, Bruce McDonald, Ioannis Stavrakakis, Ram Ramanathan, "On the Scalability of Ad Hoc Routing Protocols", Proceedings of INFOCOM2002, New York, June 23-27 2002.

[SSR01] Cesar A. Santivanez, Ioannis Stavrakakis, Ram Ramanathan, "Making LinkState Routing Scale for Ad Hoc Networks", Proc. MobiHoc 2001, Long Beach, October, 2001.

[SW03] I. Stojmenovic, J. Wu, "Broadcasting and Activity-Scheduling in Ad Hoc Networks", in *Mobile Ad Hoc Networking*, S. Basagni, M. Conti, S. Giordano, I. Stojmenovic (Editors), IEEE Press and John Wiley and Sons, Inc., New York, 2003.

[T03] Giovanni Turi, "Locating Nodes in Metropolitan Ad Hoc Networks", Proc. PWC2003, 23-25 September 2003. LNCS 2775 pp. 813-818.

[TG99] K. Tang, M. Gerla, "Fair Sharing of MAC under TCP in Wireless Ad Hoc Networks", Proceedings of IEEE MMT'99, Venice (I), October 1999.

[XG02] K. Xu, M. Gerla, "TCP over an IEEE 802.11 Ad Hoc Network: Unfairness Problems and Solutions", *UCLA Computer Science Deptartment Technical Report* – 020019, May 2002.

[XS01] S. Xu, T. Saadawi, "Does the IEEE 802.11 MAC protocol Work Well in Multihop Wireless Ad Hoc Networks?", *IEEE Communication Magazine*, Volume 39, N. 6, June 2001, pp. 130-137.

[XS02] S. Xu, T. Saadawi, "Revealing the Problems with 802.11 MAC Protocol in Multi-hop Wireless Networks", *Computer Networks*, Volume 38, N. 4, March 2002.

[WZ02] Feng Wang, Yongguang Zhang, "Improving TCP Performance over Mobile Ad-Hoc Networks with Out-of-Order Detection and Response", Proceedings of the third ACM international symposium on Mobile ad hoc networking & computing (MobiHoc 2002), Lausanne, Switzerland, 2002.

4. SECURITY AND CO-OPERATION MODEL AND MECHANISMS

Security in MANET is an essential component for basic network functions like packet forwarding and routing: network operation can be easily jeopardized if countermeasures are not embedded into basic network functions at the early stages of their design. Unlike networks using dedicated nodes to support basic functions like packet forwarding, routing, and network management, in ad hoc networks those functions are carried out by all available nodes. This very difference is at the core of the security problems that are specific to ad hoc networks. As opposed to dedicated nodes of a classical network, the nodes of an ad hoc network cannot be trusted for the correct execution of critical network functions. These security problems call on the other hand for different solutions based on the organizational links between the nodes of a MANET:

- in managed environments, the nodes are controlled by an organization (or a structured set of organizations) and an a priori trust relationship between the nodes can be derived from the existing trust relationship of the organization;
- in open environments whereby nodes and their owners aren't linked by any organizational relationship, network security mechanisms cannot rely on any existing trust relationship among the nodes.

In managed environments, entity authentication can be sufficient to verify the trust level of each node in the organization and correct execution of critical network functions is assured based on the organizational trust. Such a priori trust can only exist in a few special scenarios like military networks and corporate networks, where a common, trusted authority manage the network, and requires tamper-proof hardware for the implementation of critical functions. Entity authentication in a large network on the other hand raises key management requirements.

In managed environments without tamper-proof hardware and strong authentication infrastructure, or in open environments where a common authority that regulates the network does not exist, any node of an ad hoc network can endanger the reliability of basic functions like routing. The correct operation of the network requires not only the correct execution of critical network functions by each participating node but it also requires that each node performs a fair share of the functions. The latter requirement seems to be a strong limitation for wireless mobile nodes whereby power saving is a major concern. The threats considered in the MANET scenario are thus not limited to maliciousness and a new type of misbehavior called selfishness should also be taken into account to prevent nodes that simply do not cooperate.

With lack of a priori trust, classical network security mechanisms based on authentication and access control cannot cope with selfishness and cooperative security schemes seem to offer the only reasonable solution. In a cooperative security scheme, node misbehavior can be detected through the collaboration between a number of nodes assuming that a majority of nodes do not misbehave.

4.1. Secure Routing

Unlike traditional networks whereby routing functions are performed by dedicated nodes or routers, in MANET, routing functions are carried out by all available nodes. Likewise, common routing security mechanisms consist of node and message authentication referring to an a priori trust model in which legitimate routers are believed to perform correct operations. Authentication of a node or its messages does not guarantee the correct execution of routing functions in open environments with lack of a priori trust like MANET.

Security exposures of ad hoc routing protocols are due to two different types of attacks: active attacks through which the misbehaving node has to bear some energy costs in order to perform some harmful operation and passive attacks that mainly consist of lack of cooperation with the purpose of energy saving. Nodes that perform active attacks with the aim of damaging other nodes

by causing network outage are considered to be malicious while nodes that perform passive attacks with the aim of saving battery life for their own communications are considered to be selfish.

Malicious nodes can disrupt the correct functioning of a routing protocol by modifying routing information, by fabricating false routing information and by impersonating other nodes. Recent research studies [PHJ01] brought up also a new type of attack that goes under the name of wormhole attack. On the other side, selfish nodes can severely degrade network performances and eventually partition the network [MME02] by simply not participating to the network operation.

In the existing ad hoc routing protocols nodes are trusted in that they do not maliciously tamper with the content of protocol messages transferred among nodes. Malicious nodes can easily perpetrate integrity attacks by simply altering protocol fields in order to subvert traffic, deny communication to legitimate nodes (denial of service) and compromise the integrity of routing computations in general. As a result the attacker can cause network traffic to be dropped, redirected to a different destination or to take a longer route to the destination increasing communication delays. A special case of integrity attacks is spoofing whereby a malicious node impersonates a legitimate node due to the lack of authentication in the current ad hoc routing protocols. The main result of spoofing attacks is the misrepresentation of the network topology that possibly causes network loops or partitioning. Lack of integrity and authentication in routing protocols can further be exploited through "fabrication" referring to the generation of bogus routing messages. Fabrication attacks cannot be detected without strong authentication means and can cause severe problems ranging from denial of service to route subversion.

A more subtle type of active attack is the creation of a tunnel (or wormhole) in the network between two colluding malicious nodes linked through a private connection by-passing the network. This exploit allows a node to short-circuit the normal flow of routing messages creating a virtual vertex cut in the network that is controlled by the two colluding attackers.

Another exposure of current ad hoc routing protocols is due node selfishness that results in lack of cooperation among ad hoc nodes. A selfish node that wants to save battery life for its own communication can endanger the correct network operation by simply not participating in the routing protocol or by not forwarding packets as in the so called black hole attack. Current ad hoc routing protocols do not address the selfishness problem.

4.1.1. State of the art

Current efforts towards the design of secure routing protocols are mainly oriented to reactive (on-demand) routing protocols such as DSR [JM96] or AODV [P00], where a node attempts to discover a route to some destination only when it has a packet to send to that destination. On-demand routing protocols have been demonstrated to perform better with significantly lower overheads than proactive routing protocols in many scenarios since they are able to react quickly to topology changes while keeping routing overhead low in periods or areas of the network in which changes are less frequent. It is possible to find, however, interesting security solutions for proactive routing protocols which are worthwhile to mention.

Current secure routing protocols proposed in the literature take into account *active attacks* performed by malicious nodes that aim at intentionally tampering with the execution of routing protocols whereas *passive attacks* and the selfishness problem are not addressed. Furthermore the prerequisite for all the available solutions is a *managed* environment characterized by some security infrastructure established prior to the secure routing protocol execution. The most significant proposals for secure routing in ad hoc networks are outlined in the sequel of this section.

Secure Routing Protocol

The Secure Routing Protocol (SRP) [PH02] is designed as an extension compatible with a variety of existing *reactive* routing protocols. SRP combats attacks that disrupt the route discovery process

and guarantees the acquisition of correct topological information: SRP allows the initiator of a route discovery to detect and discard bogus replies. SRP relies on the availability of a *security association* (SA) between the source node (S) and the destination node (T). The SA could be established using a hybrid key distribution based on the public keys of the communicating parties. S and T can exchange a secret symmetric key ($K_{S,T}$) using the public keys of one another to establish a secure channel. S and T can then further proceed to mutual authentication of one another and the authentication of routing messages.

SRP copes with non-colluding *malicious* nodes that are able to modify (corrupt), replay and fabricate routing packets. In case of the Dynamic Source Routing (DSR) protocol [JM96], SRP requires including a 6-word header containing unique identifiers that tag the discovery process and a message authentication code (MAC) computed using a keyed hash algorithm. In order to initiate a route request (RREQ) the source node has to generate the MAC of the entire IP header, the basic protocol RREQ packet and the shared key $K_{S,T}$.

The intermediate nodes that relay the RREQ towards the destination measure the frequencies of queries received from their neighbors in order to regulate the query propagation process: each node maintains a priority ranking that is inversely proportional to the query rate. A node that maliciously pollutes network traffic with unsolicited RREQ will be served last (or ignored) because of its low priority ranking.

Upon reception of a RREQ, the destination node verifies the *integrity* and *authenticity* of the RREQ by calculating the keyed hash of the request fields and comparing them with the MAC contained in the SRP header. If the RREQ is valid, the destination initiates a route replay (RREP) using the SRP header the same way the source did when initiating the request. The source node discards replays that do not match with pending query identifiers and checks the integrity using the MAC generated by the destination.

The basic version of SRP suffers from the route cache poisoning attack: routing information gathered by nodes that operate in promiscuous mode in order to improve the efficiency of the DSR protocol could be invalid, because of potential fabrication by malicious nodes. The authors propose two alternative designs of SRP that use an Intermediate Node Reply Token (INRT). INRT allows intermediate nodes that belong to the same group that share a common key (K_G) to validate RREQ and provide valid RREP messages.

SRP suffers also from the lack of a validation mechanism for route maintenance messages: route error packets are not verified. However, in order to minimize the effects of fabricated error messages, SRP source-routes error packets along the prefix of the route reported as broken: the source node can thus verify that each route error feedback refers to the actual route and that it was originated at the a node that is part of the route. A malicious node can harm only routes it actually belongs to.

Assuming that the neighbor discovery mechanism maintains information on the binding of the medium access control and the IP addresses of nodes, SRP is proven to be essentially immune to IP spoofing [PH02].

SRP is, however, not immune to the wormhole attack: two colluding malicious nodes can misroute the routing packets on a private network connection and alter the perception of the network topology by legitimate nodes.

ARIADNE

Hu, Perrig and Johnson present an *on-demand* secure ad hoc routing protocol based on DSR that withstands node compromise and relies only on highly efficient *symmetric* cryptography. ARIADNE guarantees that the target node of a route discovery process can authenticate the initiator, that the initiator can authenticate each intermediate node on the path to the destination present in the RREP message and that no intermediate node can remove a previous node in the node list in the RREQ or RREP messages.

As for the SRP protocol, ARIADNE needs some mechanism to bootstrap authentic keys required by the protocol. In particular, each node needs a shared secret key ($K_{S,D}$, is the shared key between a source S and a destination D) with each node it communicates with at a higher layer, an authentic TESLA [PCST01, PCTS00] key for each node in the network and an authentic "Route Discovery chain" element for each node for which this node will forward RREQ messages.

ARIADNE provides point-to-point *authentication* of a routing message using a message authentication code (MAC) and a shared key between the two parties. However, for authentication of a broadcast packet such as RREQ, ARIADNE uses the TESLA broadcast authentication protocol. ARIADNE copes with attacks performed by *malicious* nodes that modify and fabricate routing information, with attacks using impersonation and, in an advanced version, with the wormhole attack. Selfish nodes are not taken into account.

In ARIADNE, the basic RREQ mechanism is enhanced by eight additional fields used to provide authentication and integrity to the routing protocol as follows:

<ROUTE REQUEST, initiator, target, id, time interval, hash chain, node list, MAC list>

The initiator and target are set to the address of the initiator and target nodes, respectively. As in DSR, the initiator sets the id to an identifier that it has not recently used in initiating a Route Discovery. The time interval is the TESLA time interval at the pessimistic expected arrival time of the request at the target, accounting for clock skew. The initiator of the request then initializes the hash chain to $MAC_{KS,D}$ (initiator, target, id, time interval) and the node list and MAC list to empty lists.

When any node *A* receives a RREQ for which it is not the target, the node checks its local table of <initiator, id> values from recent requests it has received, to determine if it has already seen a request from this same Route Discovery. If it has, the node discards the packet, as in DSR. The node also checks whether the time interval in the request is valid: that time interval must not be too far in the future, and the key corresponding to it must not have been disclosed yet. If the time interval is not valid, the node discards the packet. Otherwise, the node modifies the request by appending its own address (*A*) to the node list in the request, replacing the hash chain field with H [A, *hash chain*], and appending a MAC of the entire REQUEST to the MAC list. The node uses the TESLA key K_{Ai} to compute the MAC, where *i* is the index for the time interval specified in the request. Finally, the node rebroadcasts the modified RREQ, as in DSR.

When the target node receives the RREQ, it checks the validity of the request by determining that the keys from the time interval specified have not been disclosed yet, and that the hash chain field is equal to:

$H [\eta_n , H [\eta_{n-1} , H [\ldots , H [\eta_1 , MAC_{KSD}$ (initiator, target, id, time interval)] \ldots]]]

where η_i is the node address at position i of the node list in the request, and where n is the number of nodes in the node list. If the target node determines that the request is valid, it returns a RREP to the initiator, containing eight fields:

<ROUTE REPLY, target, initiator, time interval, node list, MAC list, target MAC, key list>

The target, initiator, time interval, node list, and MAC list fields are set to the corresponding values from the RREQ, the target MAC is set to a MAC computed on the preceding fields in the reply with the key KDS, and the key list is initialized to the empty list. The RREP is then returned to the initiator of the request along the source route obtained by reversing the sequence of hops in the node list of the request.

A node forwarding a RREP waits until it is able to disclose its key from the time interval specified, then it appends its key from that time interval to the key list field in the reply and forwards the packet according to the source route indicated in the packet. Waiting delays the return of the RREP but does not consume extra computational power.

When the initiator receives a RREP, it verifies that each key in the key list is valid, that the target MAC is valid, and that each MAC in the MAC list is valid. If all of these tests succeed, the node accepts the RREP; otherwise, it discards it.

In order to prevent the injection of invalid route errors into the network fabricated by any node other than the one on the sending end of the link specified in the error message, each node that encounters a broken link adds TESLA authentication information to the route error message, such that all nodes on the return path can authenticate the error. However, TESLA authentication is delayed, so all the nodes on the return path buffer the error but do not consider it until it is authenticated. Later, the node that encountered the broken link discloses the key and sends it over the return path, which enables nodes on that path to authenticate the buffered error messages.

ARIADNE is protected also from a flood of RREQ packets that could lead to the cache poisoning attack. Benign nodes can filter out forged or excessive RREQ packets using *Route Discovery chains*, a mechanism for authenticating route discovery, allowing each node to rate-limit discoveries initiated by any other node. The authors present two different approaches that can be found in [HPJ02].

ARIADNE is immune to the wormhole attack only in its advanced version: using an extension called TIK (TESLA with Instant Key disclosure) that requires tight clock synchronization between the nodes, it is possible to detect anomalies caused by a wormhole based on timing discrepancies.

ARAN

The ARAN [DLRS02]secure routing protocol proposed by Dahill, Levine, Royer and Shields is conceived as an on-demand routing protocol that detects and protects against malicious actions carried out by third parties and peers in the ad hoc environment. ARAN introduces *authentication*, message *integrity* and *non-repudiation* as part of a minimal security policy for the ad hoc environment and consists of a preliminary certification process, a mandatory end-to-end authentication stage and an optional second stage that provides secure shortest paths.

ARAN requires the use of a trusted certificate server (T): before entering in the ad hoc network, each node has to request a certificate signed by T. The certificate contains the IP address of the node, its public key, a timestamp of when the certificate was created and a time at which the certificate expires along with the signature by T. All nodes are supposed to maintain fresh certificates with the trusted server and must know T's public key.

The goal of the first stage of the ARAN protocol is for the source to verify that the intended destination was reached. In this stage, the source trusts the destination to choose the return path. A source node, *A*, initiates the route discovery process to reach the destination *X* by broadcasting to its neighbors a route discovery packet called RDP:

$$[\mathbf{RDP; IP_X ; cert_A ; N_A ; t}]\mathbf{K_{A-}}$$

The RDP includes a packet type identifier ("RDP"), the IP address of the destination (IP_X), *A*'s certificate ($cert_A$), a nonce N_A , and the current time t, all signed with A's private key. Each time A performs route discovery, it monotonically increases the nonce.

Each node records the neighbor from which it received the message. It then forwards the message to each of its neighbors, signing the contents of the message. This signature prevents spoofing attacks that may alter the route or form loops. Let *A*'s neighbor be *B*. It will broadcast the following message:

[[RDP; IP_X ; $cert_A$; N_A ; t]K_{A-}]K_{B-} ; $cert_B$

Nodes do not forward messages for which they have already seen the (N_A ; IP_A) tuple. The IP address of A is contained in the certificate, and the monotonically increasing nonce facilitates easy storage of recently-received nonces.

Upon receiving the broadcast, *B*'s neighbor *C* validates the signature with the given certificate. *C* then rebroadcasts the RDP to its neighbors, first removing B's signature:

[[RDP; IP_X ; $cert_A$; N_A ; t]K_{A-}]K_{C-} ; $cert_C$

Eventually, the message is received by the destination, *X*, who replies to the first RDP that it receives for a source and a given nonce. There is no guarantee that the first RDP received traveled along the shortest path from the source. The destination unicasts a Reply (REP) packet back along the reverse path to the source. Let the first node that receives the RDP sent by X be node *D*. *X* will send to *D* the following message:

[REP; IP_A ; $cert_X$; N_A ; t]K_{X-}

The REP includes a packet type identifier ("REP"), the IP address of A, the certificate belonging to *X*, the nonce and associated timestamp sent by *A*. Nodes that receive the REP forward the packet back to the predecessor from which they received the original RDP. All REPs are signed by the sender. Let *D*'s next hop to the source be node *C*. *D* will send to *C* the following message:

[[REP; IP_A ; $cert_X$; N_A ; t]K_{X-}]K_{D-} ; $cert_D$

C validates *D*'s signature, removes the signature, and then signs the contents of the message before unicasting the following RDP message to *B*:

[[REP; IP_A ; $cert_X$; N_A ; t]K_{X-}]K_{C-} ; $cert_C$

A node checks the signature of the previous hop as the REP is returned to the source. This avoids attacks where malicious nodes instantiate routes by impersonation and re-play of X's message. When the source receives the REP, it verifies that the correct nonce was returned by the destination as well as the destination's signature. Only the destination can answer an RDP packet. Other nodes that already have paths to the destination cannot reply for the destination. While other protocols allow this networking optimization, ARAN removes several possible exploits and cuts down on the reply traffic received by the source by disabling this option.

The second stage of the ARAN protocol guarantees in a secure way that the path received by a source initiating a route discovery process is the shortest. Similarly to the first stage of the protocol, the source broadcasts a *Shortest Path Confirmation* (SPC) message to its neighbors: the SPC message is different from the RDP message only in two additional fields that provide the destination *X* certificate and the encryption of the entire message with *X*'s public key (which is a costly operation). The onion-like signing of messages combined with the encryption of the data prevents nodes in the middle from changing the path length because doing so would break the integrity of SPC the packet.

Also the route maintenance phase of the ARAN protocol is secured by digitally signing the route error packets. However it is extremely difficult to detect when error messages are *fabricated* for

links that are truly active and not broken. Nevertheless, because messages are signed, malicious nodes cannot generate error messages for other nodes. The non-repudiation provided by the signed error message allows a node to be verified as the source of each error message that it sends.

As with any secure system based on cryptographic certificates, the key revocation issue has to be addressed in order to make sure that expired or revoked certificates do not allow the holder to access the network. In ARAN, when a certificate needs to be revoked, the trusted certificate server T sends a broadcast message to the ad hoc group that announces the revocation. Any node receiving this message re-broadcast it to its neighbors. Revocation notices need to be stored until the revoked certificate would have expired normally. Any neighbor of the node with the revoked certificate needs to reform routing as necessary to avoid transmission through the now un-trusted node. This method is not failsafe. In some cases, the un-trusted node that is having its certificate revoked may be the sole connection between two parts of the ad hoc network. In this case, the un-trusted node may not forward the notice of revocation for its certificate, resulting in a partition of the network, as nodes that have received the revocation notice will no longer forward messages through the un-trusted node, while all other nodes depend on it to reach the rest of the network. This only lasts as long as the un-trusted node's certificate would have otherwise been valid, or until the un-trusted node is no longer the sole connection between the two partitions. At the time that the revoked certificate should have expired, the un-trusted node is unable to renew the certificate, and routing across that node ceases. Additionally, to detect this situation and to hasten the propagation of revocation notices, when a node meets a new neighbor, it can exchange a summary of its revocation notices with that neighbor; if these summaries do not match, the actual signed notices can be forwarded and re-broadcasted to restart propagation of the notice.

The ARAN protocol protects against exploits using *modification*, *fabrication* and *impersonation* but the use of asymmetric cryptography makes it a very costly protocol to use in terms of CPU and energy usage. Furthermore, ARAN is not immune to the *wormhole* attack.

SEAD

Hu, Perrig and Johnson [HJP02] present a *proactive* secure routing protocol based on the Destination-Sequenced Distance Vector protocol (DSDV). In a proactive (or periodic) routing protocol nodes periodically exchange routing information with other nodes in attempt to have each node always know a current route to all destinations [PB94]. Specifically, SEAD is inspired by the DSDV-SQ version of the DSDV protocol. The DSDV-SQ version of the DSDV protocol has been shown to outperform other DSDV versions in previous ad hoc networks simulations [BMJHJ98, JLHMD99].

SEAD deals with attackers that *modify* routing information broadcasted during the update phase of the DSDV-SQ protocol: in particular, routing can be disrupted if the attacker modifies the sequence number and the metric field of a routing table update message. *Replay attacks* are also taken into account.

In order to secure the DSDV-SQ routing protocol, SEAD makes use of efficient *one-way hash chains* rather than relaying on expensive asymmetric cryptography operations. However, like the other secure protocols presented in this chapter, SEAD assumes some mechanism for a node to distribute an authentic element of the hash chain that can be used to authenticate all the other elements of the chain. As a traditional approach, the authors suggest to ensure the key distribution relaying on a trusted entity that signs public key certificates for each node; each node can then use its public key to sign a hash chain element and distribute it.

The basic idea of SEAD is to authenticate the sequence number and metric of a routing table update message using hash chains elements. In addition, the receiver of SEAD routing information also authenticates the sender, ensuring that the routing information originates form the correct node.

To create a one-way hash chain, a node chooses a random initial value $x \in \{0,1\}^{\rho}$, where ρ is the length in bits of the output of the hash function, and computes the list of values $h_0,h_1,h_2,h_3,...,h_n$, where $h_0=x$, and $h_i = H(h_{i-1})$ for $0< i \leq n$, for some n. As an example, given an authenticated h_i value, a node can authenticate h_{i-3} by computing $H(H(H(h_{i-3})))$ and verifying that the resulting value equals *hi*.

Each node uses a specific authentic (i.e. signed) element from its hash chain in each routing update that it sends about itself (metric 0). Based on this initial element, the one-way hash chain provides authentication for the lower bound on the metric in other routing updates for that node. The use of a hash value corresponding to the sequence number and metric in a routing update entry prevents any node from advertising a route to some destination claiming a greater sequence number than that destination's own current sequence number. Likewise, a node can not advertise a route better than those for which it has received an advertisement, since the metric in an existing route cannot be decreased due to the on-way nature of the hash chain.

When a node receives a routing update, it checks the authenticity of the information for each entry in the update using the destination address, the sequence number and the metric of the received entry, together with the latest prior *authentic* hash value received from that destination's hash chain. Hashing the received elements the correct number of times (according to the prior authentic hash value) assures the authenticity of the received information if the calculated hash value and the authentic hash value match.

The source of each routing update message in SEAD must also be authenticated, since otherwise, an attacker may be able to create routing loops through the *impersonation* attack. The authors propose two different approaches to provide node authentication: the first is based on a broadcast authentication mechanism such as TESLA, the second is based on the use of Message Authentication Codes, assuming a shared secret key between each couple of nodes in the network.

SEAD does not cope with *wormhole* attacks though the authors propose, as in the ARIADNE protocol, to use the TIK protocol to detect the threat.

Notes on the wormhole attack

The wormhole attack is a severe threat against ad hoc routing protocols that is particularly challenging to detect and prevent. In a wormhole attack a malicious node can record packets (or bits) at one location in the network and tunnel them to another location through a private network shared with a colluding malicious node. Most existing ad hoc routing protocols, without some mechanism to defend them against the wormhole attack, would be unable to find consistent routes to any destination, severely disrupting communication.

A dangerous threat can be perpetrated if a wormhole attacker tunnels all packets through the wormhole honestly and reliably since no harm seems to be done: the attacker actually seems to provide a useful service in connecting the network more efficiently. However, when an attacker forwards only routing control messages and not data packets, communication may be severely damaged. As an example, when used against an on demand routing protocol such as DSR, a powerful application of the wormhole attack can be mounted by tunneling each RREQ message directly to the destination target node of the request. This attack prevents routes more than two hops long from being discovered because RREP messages would arrive to the source faster than any other replies or, worse, RREQ messages arriving from other nodes next to the destination than the attacker would be discarded since already seen.

Hu, Perrig and Johnson propose an approach to detect a wormhole based on *packet leashes* [PHJ01]. The key intuition is that by authenticating either an extremely precise timestamp or location information combined with a loose timestamp, a receiver can determine if the packet has traversed a distance that is unrealistic for the specific network technology used.

Temporal leashes rely on extremely precise time synchronization and extremely precise timestamps in each packet. The travel time of a packet can be approximated as the difference between the receive time and the timestamp. Given the precise time synchronization required by temporal leashes, the authors propose efficient broadcast authenticators based on symmetric primitives. In particular they extend the TESLA broadcast authentication protocol to allow the disclosure of the authentication key within the packet that is authenticated.

Geographical leashes are based on location information and loosely synchronized clocks. If the clocks of the sender and the receiver are synchronized within a certain threshold and the velocity of any node is bounded, the receiver can compute an upper bound on the distance between the sender and itself and use it to detect anomalies in the traffic flow. In certain circumstances however, bounding the distance between the sender and the receiver cannot prevent wormhole attacks: when obstacles prevent communication between two nodes that would otherwise be in transmission range, a distance-based scheme would still allow wormholes between the sender and the receiver. To overcome this problem, in a variation of the geographical leashes the receiver verifies that every possible location of the sender can reach every possible location of the receiver based on a radio propagation model implemented in every node.

In some special cases, wormholes can also be detected through techniques that don't require precise time synchronization nor location information. As an example, it would be sufficient to modify the routing protocol used to discover the path to a destination so that it could handle multiple routes: a verification mechanism would then detect anomalies when comparing the metric (e.g. number of hops) associated to each route. Any node advertising a path to a destination with a metric considerably lower than all the others could raise the suspect of a wormhole.

Furthermore, if the wormhole attack is performed only on routing information while dropping data packets, other mechanisms can be used to detect this misbehavior. When a node doesn't correctly participate to the network operation by not executing a particular function (e.g. packet forwarding) a collaborative monitoring technique can detect and gradually isolate misbehaving nodes. Lack of cooperation and security mechanism used to enforce node cooperation to the network operation is the subject of the next section.

4.1.2. Secure routing in MobileMan

In this area we are investigating a new approach that does not rely on any fixed infrastructure nor requires a managed network setup. Our research is still to be considered work in progress: we plan to provide a self-organizing security infrastructure strongly linked to the presence of a cooperation enforcement mechanism based on reputation.

Basic security services provided by our infrastructure are entity authentication and routing message integrity. The main issue that has still to be solved is the secure distribution of self-generated keying material, a problem that goes under the name of *contributory key agreement* and for which some solutions (based, however, on the presence of a central authority) are available in the literature [STW00, CT03, LDM03, BEGA02].

4.2. Co-operation Mechanisms

Selfishness is a new type of misbehavior that is inherent to ad hoc networks and cooperation enforcement is the countermeasure against selfishness. A selfish node does not directly intend to damage other nodes with active attacks (mainly because performing active attacks can be very expensive in terms of energy consumption) but it simply does not contribute in the network operation, saving battery life for its own communications. Selfishness can cause serious damage in terms of global network throughput and delay as shown by a simulation study on the impact of selfish behavior on the DSR routing protocol [MME02]. The node selfishness problem has only recently been addressed by the research community, and still very few cooperation enforcement mechanisms are proposed to combat such misbehavior. Current cooperation enforcement proposals

for MANET fall in two categories: currency-based solutions whereby some form of digital cash is used as an incentive for cooperation and monitoring solutions based on the principle that misbehaving nodes will be detected through the shared observations of a majority of legitimate nodes. The most significant proposals in each category are outlined in the sequel of this section.

4.2.1. State of the art

Nuglets

In [BH01], Buttyan and Hubaux present two important issues targeted specifically at the ad hoc networking environment: first, end-users must be given some incentive to contribute in the network operation (especially to relay packets belonging to other nodes); second, end-users must be discouraged from overloading the network. The solution consists of a virtual currency call Nuglet used in every transaction. Two different models are described: the Packet Purse Model and the Packet Trade Model. In the Packet Purse Model each packet is loaded with nuglets by the source and each forwarding host takes out nuglets for its forwarding service. The advantage of this approach is that it discourages users from flooding the network but the drawback is that the source needs to know exactly how many nuglets it has to include in the packet it sends. In the Packet Trade Model each packet is traded for nuglets by the intermediate nodes: each intermediate node buys the packet from the previous node on the path. Thus, the destination has to pay for the packet. The direct advantage of this approach is that the source does not need to know how many nuglets need to be loaded into the packet. On the other hand, since the packet generation is not charged, malicious flooding of the network cannot be prevented. There are some further issues that have to be solved: concerning the Packet Purse Model, the intermediate nodes are able to take out more nuglets than they are supposed to; concerning the Packet Trade Model, the intermediate nodes are able to deny the forwarding service after taking out nuglets from a packet.

CONFIDANT

Buchegger and Le Boudec proposed a technique called CONFIDANT (Cooperation Of Nodes, Fairness In Dynamic Ad-hoc NeTworks) [BLB01, BLB02] aiming at detecting malicious nodes by means of combined monitoring and reporting and establishing routes by avoiding misbehaving nodes. CONFIDANT is designed as an extension to a routing protocol such as DSR. CONFIDANT components in each node include a network monitor, reputation records for first-hand and trusted second-hand observations about routing and forwarding behavior of other nodes, trust records to control trust given to received warnings, and a path manager to adapt the behavior of the local node according to reputation and to take action against malicious nodes. The term reputation is used to evaluate routing and forwarding behavior according to the network protocol, whereas the term trust is used to evaluate participation in the CONFIDANT meta-protocol.

The dynamic behavior of CONFIDANT is as follows. Nodes monitor their neighbors and change the reputation accordingly. If they have a reason to believe that a node misbehaves, they can take action in terms of their own routing and forwarding and they can decide to inform other nodes by sending an ALARM message. When a node receives such an ALARM either directly or by promiscuously listening to the network, it evaluates how trustworthy the ALARM is based on the source of the ALARM and the accumulated ALARM messages about the node in question. It can then decide whether to take action against the misbehaved node in the form of excluding routes containing the misbehaved node, re-ranking paths in the path cache, reciprocating by non-cooperation, and forwarding an ALARM about the node.

The first version of CONFIDANT was, despite the filtering of ALARM messages in the trust manager, vulnerable to concerted efforts of spreading wrong accusations. In a recent enhancement of the protocol, this problem has been addressed by the use of Bayesian statistics for classification and the exclusion of liars.

Simulations with nodes that do not participate in the forwarding function have shown that CONFIDANT can cope well, even if half of the network population acts maliciously. Further simulations concerning the effect of second-hand information and slander have shown that slander can effectively be prevented while still retaining a significant detection speed-up over using merely first-hand information.

The limitations of CONFIDANT lie in the assumptions for detection-based reputation systems. Events have to be observable and classifiable for detection, and reputation can only be meaningful if the identity of each node is persistent, otherwise it is vulnerable to spoofing attacks.

Token-based cooperation enforcement

In [YML02] Yang, Meng, Lu suggest a mechanism whereby each node of the ad hoc network is required to hold a token in order to participate in the network operations. Tokens are granted to a node collaboratively by its neighbors based on the monitoring of the node's contribution to packet forwarding and routing operations. Upon expiration of the token, each node renews its token through a token renewal exchange with its neighbors: the duration of a token's validity is based on the duration of the node's correct behavior as monitored by the neighbors granting/renewing the token. This mechanism typically allows a well-behaved node to accumulate credit and to renew its token less frequently as time evolves.

The token-based cooperation enforcement mechanism includes four interacting components: *neighbor verification* through which the local node verifies whether neighboring nodes are legitimate, *neighbor monitoring* that allows the local node to monitor the behavior of each node in the network and to detect attacks from malicious nodes, *intrusion reaction* that assures the generation of network alerts and the isolation of attackers, and *security enhanced routing protocol* that consists of the ad hoc routing protocol including security extensions.

A valid token is constructed using a group signature whereby a mechanism based on polynomial secret sharing [S79] assures that at least k neighbors agree to issue or renew the token. The key setup complexity of polynomial secret sharing and the requirement for at least k nodes to sign each token both are incompatible with high mobility and call for a rather large and dense ad hoc network. Furthermore the duration of a token's validity increases proportionally with the duration of the node's correct behavior as monitored by its neighbors; this feature again calls for low mobility. The token-based cooperation enforcement mechanism is thus suitable for ad hoc networks where node mobility is low. Spoofing attacks through which a node can request more than one token claiming different identity, are not taken into account by the proposal even if the authors suggest that MAC addresses can be sufficient for node authentication purposes.

4.2.2. Co-operation in MobileMan

The cooperation enforcement mechanism proposed for the MobileMan architecture is the CORE mechanism. CORE [MMC02] is a collaborative monitoring mechanism based on reputation that strongly binds network utilization and the correct participation to basic networking function like routing and packet forwarding.

CORE has been implemented for the QualNet network simulator and simulations are being prepared to measure its efficacy. We plan also to develop a Linux user-space daemon version of CORE that can possibly be used by different routing protocols and that stores reputation information in a local storage accessible from different layers of the MobileMan stack in order to help inter-layer optimization.

4.3. Authentication and Key Management

Authentication of peer entities involved in ad hoc routing and the integrity verification of routing exchanges are the two essential building blocks of secure routing. Both entity authentication and

message integrity call on the other hand for a key management mechanism to provide parties involved in authentication and integrity verification with proper keying material. Key management approaches suggested by current secure routing proposals fall in two categories:

- manual configuration of symmetric (secret) keys: the pair-wise secret keys can serve as key encryption keys in a point-to-point key exchange protocol to establish session keys used for authentication and message integrity between communicating nodes. If some dedicated infrastructure including a key server can be afforded, automatic distribution of session keys with a key distribution protocol like Kerberos can also be envisioned.
- public-key based scheme: each node possesses a pair of public and private keys based on an asymmetric algorithm like RSA. Based on this keypair each node can perform authentication and message integrity operations or further exchange pair-wise symmetric keys used for efficient authentication and encryption operations.

Secure routing proposals like SRP assume manual configuration of secure associations based on shared secret keys. Most of other proposals such as Ariadne rely on a public-key based scheme whereby a well known trusted third party (TTP) issues public key certificates used for authentication. The requirement for such a public-key infrastructure does not necessarily imply a managed ad hoc network environment and an open environment can be targeted as well. Indeed, it is not necessary for the mobile nodes that form the ad hoc network to be managed by the public-key certification authority. However, the bootstrap phase requires an external infrastructure, which has to be available also during the lifetime of the ad hoc network to provide revocation services for certificates that have expired or that have been explicitly revoked.

Two interesting proposals presented in the next section tackle the complexity of public-key infrastructures in the ad hoc network environment through self-organization: public-key management based on the concept of web of trust akin to Pretty Good Privacy (PGP) and a public-key certification mechanism based on polynomial secret sharing.

4.3.1. State of the art

Self-Organized Public-Key Management based on PGP

Capkun, Buttyan and Hubaux propose a fully self-organized public key management system that can be used to support security of ad hoc network routing protocols [CBH02]. The suggested approach is similar to PGP [Z95] in the sense that users issue certificates for each other based on their personal acquaintances. However, in the proposed system, certificates are stored and distributed by the users themselves, unlike in PGP, where this task is performed by on-line servers (called certificate directories). In the proposed self-organizing public-key management system, each user maintains a *local certificate repository*. When two users want to verify the public keys of each other, they merge their local certificate repositories and try to find appropriate certificate chains within the merged repository.

The success of this approach very much depends on the construction of the local certificate repositories and on the characteristics of the certificate graphs. The vertices of a certificate graph represent public-keys of the users and the edges represent public-key certificates issued by the users. The authors investigate several repository construction algorithms and study their performance. The proposed algorithms take into account the characteristics of the certificate graphs in a sense that the choice of the certificates that are stored by each mobile node depends on the connectivity of the node and its neighbors in the certificate graph.

More precisely, each node stores in its local repository several directed and mutually disjoint paths of certificates. Each path begins at the node itself, and the certificates are added to the path such that a new certificate is chosen among the certificates connected to the last node on the path (initially the node that stores the certificates), such that the new certificate leads to the node that has the highest number of certificates connected to it (i.e., the highest vertex degree). The authors

call this algorithm the *Maximum Degree Algorithm*, as the local repository construction criterion is the degree of the vertices in a certificate graph.

In a more sophisticated extension called the *Shortcut Hunter Algorithm*, certificates are stored into the local repositories based on the number of the shortcut certificates connected to the users. The shortcut certificate is a certificate that, when removed from the graph makes the shortest path between two users previously connected by this certificate strictly larger than two.

When verifying a certificate chain, the node must trust the issuer of the certificates in the chain for correctly checking that the public key in the certificate indeed belongs to the node identification (ID) named in the certificate. When certificates are issued by the mobile nodes of an ad hoc network instead of trusted authorities, this assumption becomes unrealistic. In addition, there may be malicious nodes who issue false certificates. In order to alleviate these problems, the authors propose the use of authentication metrics [RS99]: it is not enough to verify a node ID key binding via a single chain of certificates. The authentication metric is a function that accepts two keys (the verifier and the verified node) and a certificate graph and returns a numeric value corresponding to the degree of authenticity of the key that has to be verified: one example of authentication metric is the number of disjoint chains of certificates between two nodes in a certificate graph.

The authors emphasize that before being able to perform key authentication, each node must first build its local certificate repository, which is a complex operation. However this initialization phase must be performed rarely and once the certificate repositories have been built, then any node can perform key authentication using only local information and the information provided by the targeted node. It should also be noted that local repositories become obsolete if a large number of certificate are revoked, as then the certificate chains are no longer valid; the same comment applies in the case when the certificate graph changes significantly. Furthermore, PGP-like schemes are more suitable for small communities because that the authenticity of a key can be assured with a higher degree of trustiness. The authors propose the use of authentication metrics to alleviate this problem: this approach however provides only probabilistic guarantees and is dependent on the characteristics of the certificate graph on which it operates. The authors also carried out a simulation study showing that for the certificate graphs that are likely to emerge in self-organized systems, the proposed approach yields good performances both in terms of the size of the local repository stored in each node and scalability.

Authentication based on polynomial secret sharing

In [LL03] Luo and Lu present an authentication service whereby the public-key certificate of each node is cooperatively generated by a set of neighbors based on the behavior of the node as monitored by the neighbors. Using a group signature mechanism based on polynomial secret sharing, the secret digital signature key used to generate public-key certificates is distributed among several nodes. Certification services like issuing, renewal and revocation of certificates thus are distributed among the nodes: a single node holds just a share of the complete certificate signature key. The authors propose a *localized trust model* to characterize the localized nature of security concerns in large ad hoc wireless networks. When applying such trust model, an entity is trusted if any k trusted entities claim so: these k trusted entities are typically the neighboring nodes of the entity. A locally trusted entity is globally accepted and a locally distrusted entity is regarded untrustworthy all over the network.

In the suggested security architecture, each node carries a certificate signed by the shared certificate signing key SK, while the corresponding public key PK is assumed to be well-known by all the nodes of the network, so that certificates are globally verifiable. Nodes without valid certificates will be isolated, that is, their packets will not be forwarded by the network. Essentially, any node without a valid certificate is considered a potential intruder. When a mobile node moves to a new location, it exchanges certificates with its new neighbors and goes through mutual authentication process to build trust relationships. Neighboring nodes with such trust relationship help each other to forward and route packets. They also monitor each other to detect possible

attacks and break-ins. Specific monitoring algorithms and mechanisms are left to each individual node's choice. When a node requests a signed certificate from a coalition of *k* nodes, each of the latter checks its records about the requesting node. If the requestor is recorded as a legitimate node, a partial certificate is computed by applying the local node's share of SK and returned to the requestor. Upon collecting *k* partial certificates, the requesting node combines them to generate the complete certificate of its public-key as if issued by a centralized certification authority.

The multiple signature scheme used to build the certificate is based on a k-threshold polynomial secret sharing mechanism. This technique requires a bootstrapping phase where a "dealer" has to privately send each node its share of the secret signature key SK. The authors propose a scalable initialization mechanism called "self-initialization" whereby the dealer only has to initialize the very first *k* nodes, regardless of the global network span. The initialized nodes collaboratively initialize other nodes: repeating this procedure, the network progressively self-initializes itself. The same mechanism is applied when new nodes join the network.

Certificate revocation is also handled by the proposed architecture and an original approach to handle roaming adversaries is presented in order to prevent a misbehaving node that moves to a new location from getting a valid certificate. Roaming nodes are defeated with the flooding of "accusation" messages that travel in the network and inform distant nodes about the behavior of a suspect node.

The main drawback of the proposed architecture is the requirement for a trusted dealer that initializes the very first *k* nodes of a coalition to the choice of the system-wide parameter *k*. To cope with the first problem, the authors propose to use a distributed RSA key pair generation [S79] for the very first *k* nodes. The other major limitation of the scheme is the strong assumption that every node of the network has at least *k* trusted neighbors. Moreover, the authors assume that any new node that joins the system already has an initial certificate issued by an offline authority or by a coalition of *k* neighbors.

4.4. MANET and Data Link Layer Security

Various security mechanisms have been proposed as part of 802.11 [IEEE802.11] and Bluetooth [BT01] specifications. While the robustness of these mechanisms has often been argued [SLR01], the main question is the relevance of security mechanisms implemented in the data link layer with respect to the requirements of MANET. This question deserves careful analysis in the light of requirements raised by the two different environments in which these mechanisms can potentially be deployed:

1. wireless extension of a wired infrastructure as the original target of 802.11 and Bluetooth security mechanisms,
2. wireless ad hoc networks with no infrastructure.

In case of 1 the main requirement for data link layer security mechanisms is the need to cope with the lack of physical security on the wireless segments of the communication infrastructure. Data link layer security is then perfectly justified as a means of building a "wired equivalent" security as stated by the objectives of Wired Equivalent Privacy (WEP) of 802.11. Data link layer mechanisms like the ones provided by 802.11 and Bluetooth basically serve for access control and privacy enhancements to cope with the vulnerabilities of radio communication links. However, data link layer security performed at each hop cannot meet the end-to-end security requirements of applications neither on wireless links protected by 802.11 or Bluetooth nor on physically protected wired links.

In case of wireless ad hoc networks as defined in 2 there are two possible scenarios:

- managed environments whereby the nodes of the ad hoc network are controlled by an organization and can thus be trusted based on authentication,
- open environments with no a priori organization among network nodes.

The managed environment raises requirements similar to the ones of 1. Data link layer security is justified in this case by the need to establish a trusted infrastructure based on logical security means. If the integrity of higher layer functions implemented by the nodes of a managed environment can be assured (i.e. using tamper-proof hardware) then data link layer security can even meet the security requirements raised by higher layers including the routing protocol and the applications.

Open environments on the other hand offer no trust among the nodes and across communication layers. In this case trust in higher layers like routing or application protocols cannot be based on data link layer security mechanisms. The only relevant use of the latter appears to be ad hoc routing security proposals whereby the data link layer security can provide node-to-node authentication and data integrity as required by the routing layer. Moreover the main impediment to the deployment of existing data link layer security solutions (802.11 and Bluetooth) would be the lack of support for automated key management which is mandatory in open environments whereby manual key installation is not suitable.

4.5. References

[BEGA02] R. Bobba, L. Eschenauer, V. Gligor, W. Arbaugh, Bootstrapping Security Associations for Routing in Mobile Ad hoc Networks

[BH01] L. Buttyan, J.-P. Hubaux, Nuglets: a virtual currency to stimulate cooperation in self-organized ad hoc networks, Technical Report DSC/2001/001, Swiss Federal Institute of Technology -- Lausanne, 2001.

[BLB01] S. Buchegger, J.-Y. Le Boudec, Nodes Bearing Grudges: Towards Routing Security, Fairness, and Robustness in Mobile Ad Hoc Networks, in proceedings of the 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing.

[BLB02] S. Buchegger, J.-Y. Le Boudec, Performance Analysis of the CONFIDANT Protocol, in proceedings of MobiHoc 2002.

[BMJHJ98] J. Broch, D. A. Maltz, D. B. Johnson, Y-C Hu, J. G. Jetcheva, A performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols, in proceedings of MOBICOM 1998.

[BT01] "Specification of the Bluetooth System", Bluetooth Special Interest Group, Version 1.1, February 22, 2001, http://www.bluetooth.com/pdf/Bluetooth_11_Specifications_Book.pdf

[CBH02] S. Capkun, L. Buttyan and J-P Hubaux, Self-Organized Public-Key Management for Mobile Ad Hoc Networks, in ACM International Workshop on Wireless Security, WiSe 2002.

[CT03] W. Chen, D. Towsley, Modeling (k,n) Threshold Schemes in Ad hoc Netoworks, Course Project CMPSCI 691R – Performance Evaluation, University of Massachusetts, Amherst

[DLRS02] B. Dahill, B. N. Levine, E. Royer, C. Shields, ARAN: A secure Routing Protocol for Ad Hoc Networks, UMass Tech Report 02-32, 2002.

[HJP02] Y-C Hu, D. B. Johnson, A. Perrig, SEAD: Secure Efficient Distance Vector Routing for Mobile Wireless Ad Hoc Networks, in the Fourth IEEE Workshop on Mobile Computing Systems and Applications.

[HPJ02] Y-C Hu, A. Perrig, D. B. Johnson, Ariadne : A secure On-Demand Routing Protocol for Ad Hoc Networks, in proceedings of MOBICOM 2002.

[IEEE802.11] IEEE 802.11b-1999 Supplement to 802.11-1999,Wireless LAN MAC and PHY specifications: Higher speed Physical Layer (PHY) extension in the 2.4 GHz band

[JLHMD99] P. Johansson, T. Larsson, N. Hedman, B. Mielczarek, M. Degermark, Scenario-based Performance Analysis of Routing Protocols for Mobile Ad Hoc Networks, in proceedings of MOBICOM 1999.

[JM96] D. B. Johnson, D. A. Maltz, Dynamic Source Routing in Ad Hoc Wireless Networks, Mobile Computing, edited by Tomasz Imielinski and Hank Korth, Chapter 5, pages 153-181, Kluwer Academic Publishers, 1996.

[KZLLZ01] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang, "Providing robust and ubiquitous security support for manet", In Proc. IEEE ICNP, 2001

[LDM03] B. Lehane, L. Doyle, D. O'Mahony, Shared RSA Key Generation in A Mobile Ad hoc Network

[LL03] H. Luo, S. Lu, Ubiquitous and Robust Authenticaion Services for Ad Hoc Wireless Networks, UCLA-CSD-TR-200030.

[MGLB00] S. Marti, T. Giuli, K. Lai, and M. Baker, Mitigating routing misbehavior in mobile ad hoc networks, in proceedings of MOBICOM 2000.

[MMC02] P. Michiardi, R. Molva, Core: A COllaborative REputation mechanism to enforce node cooperation in Mobile Ad Hoc Networks, IFIP - Communication and Multimedia Security Conference 2002.

[MME02] P. Michiardi, R. Molva, Simulation-based Analysis of Security Exposures in Mobile Ad Hoc Networks, in proceedings of European Wireless Conference, 2002.

[MMR02] P. Michiardi, R. Molva, Game Theoretic Analysis of Security in Mobile Ad Hoc Networks, Institut Eurecom Research Report RR-02-070 - April 2002.

[P00] Charles Perkins, Ad hoc On Demand Distance Vector (AODV) Routing, Internet draft, draft-ietf-manet-aodv-00.txt.

[PB94] C. E. Perkins, P. Bhagwat, Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers, in proceedings of SIGCOMM 1994.

[PCST01] A. Perrig, R. Canetti, D. Song, J.D. Tygar, Efficient and secure source authentication for multicast, in proceedings of NDSS 2001

[PCTS00] A. Perrig, R. Canetti, J.D. Tygar, D. Song, Efficient authentication and signing of multicast streams over lossy channels, in IEEE Symposium on Security and Privacy, 2000.

[PH02] P. Papadimitratos, Z. Haas, Secure Routing for Mobile Ad Hoc Networks, in proceedings of CNDS 2002.

[PHJ01] A. Perrig, Y-C Hu, D. B. Johnson, Wormhole Protection in Wireless Ad Hoc Networks, Technical Report TR01-384, Dep. Of Computer Science, Rice University.

[RS99] M. Reiter, S. Stybblebine, Authentication metric analysis and design, ACM Transactions on Information and System Security, 1999.

[S79] A. Shamir, How to share a secret, Communications of ACM 1979.

[S96] Bruce Schneier , “Applied Cryptography”, John Wiley & Sons, 1996

[SLR01] Stubblefield, Loannidis, and Rubin, “Using the Fluhrer, Mantin, and Shamir Attack to Break WEP”, AT&T Labs Technical Report 2001

[STW00] M. Steiner, G. Tsudik, M. Waidner, Key Agreement in Dynamic Peer Groups, IEEE Transactions on Parallel and Distributed Systems, Vol. 11, No. 8, August 2000

[YML02] Yang, X. Meng, S. Lu, Self-Organized Network-Layer Security in Mobile Ad Hoc Networks.

[Z95] P. Zimmermann, The Official PGP User’s Guide. MIT Press, 1995.

5. MIDDLEWARE

The middleware layer operates between the networking layers and the distributed applications (i.e., it mainly implements layers 5-7 of the OSI model), with the aim to build on top of raw network services, higher level mechanisms that easy the development and deployment of applications. Mobile ad hoc systems currently developed adopt the approach of not having a middleware, but rather rely on each application to handle all the services it needs. This constitutes a major complexity/inefficiency in the development of MANET applications.
Research on middleware for mobile ad hoc networks is still in its infancy. Ad hoc networking and self-organization have not yet received the attention they deserve. Existing middleware mainly focus on mobile/nomadic environments, where a fixed infrastructure contains the relevant information. For an overview on middleware for mobile and pervasive systems, see [MCE02][ADG02][CCR02].

Recently, in research circles, some middleware proposals for mobile ad hoc environments appeared in [MC02][H03][MPR01][MCE02a]. Their emphasis is on supporting transient data sharing [MPR01] between nodes in communication range, data replication for disconnected operations [MCE02], or both [H03]. To achieve this, classical middleware technologies have been adopted. These include tuple space, mobile agents, and reactive programming through the usage of events' publishing/subscribing [MCE02][ADG02]. While these technologies provide service abstractions that highly simplify the application development, their efficiency in ad hoc environments is still an open issue. Specifically, among others, solutions must be devised to implement and manage in an efficient way agents' synchronization, shared memory, and to support group communications in an ad hoc network.

Among middleware services, *Service discovery and location* play a relevant role in ad hoc environments. Upon joining a self-organizing network, mobile nodes should be able to explore the environment to learn and locate the available services. Due to the scarce resources of a MANET the service discovery and location should be designed to act in a "context aware" manner [MCE02]. Context information, such as node's current position (both geographical and logical in terms of network topology), neighborhood, available resources and constraints must be used to select the most appropriate service providers. A novel notion of "nearness" based on communication proximity (e.g., to measure the existence a stable communication path between the terminal and the service provider, rather then physical proximity) would be useful to estimate the amount of resources needed to access a service [W01].

Current solutions on service discovery assume (more or less) a fairly stable network since they are based on centralized service registries. Several architectures for service distribution in ad hoc networks have been proposed in literature. Many of them focus on the mechanisms for describing and matching services, as well as the user interfaces for discovery applications (i.e. [HDV01] and [TH01]). In contrast, service discovery models in Ad Hoc networks require the existence of service description ontology, and focus in issues such as the efficiency of the discovery mechanism, the reliability of the underlying communication mechanism, and the service lifetime management. This requires having a service distribution based on a set of core nodes, which are referred as the service backbone. The backbone constitutes the infrastructure to provide service discovery information, thus relieving non-backbone nodes from participation in this activity. Ad hoc networks require a higher degree of freedom and a fully decentralized architecture that is provided by a service backbone, which will be fully distributed or organized in clusters.

An approach to QoS-Aware resource discovery in ad hoc network has been presented in [LSZ03]. The proposed approach implements, in an ad hoc environment, the rendezvous discovery approach commonly used by middleware for mobile/nomadic networks, e.g., the Java Intelligent Network Infrastructure (Jini). Rendezvous servers (brokers) store the service-publish requests coming from service providers, and deliver service information to requesting clients. In an ad hoc network, brokers must be dynamically identified. Specifically, in [LSZ03] the brokers (directory agents)

election happens through the usage of clusters formation techniques. To reduce the communication overheads, most of the discovery messages are only exchanged among these directory agents. Hash indexing is applied to distributed agents for reducing the query latency. Specifically, a hash function applied to the service attributes returns the list of directory agents. QoS guarantees are achieved through a continuous monitoring.

5.1. P2P information delivery

Ad hoc communities make use of fully decentralized mechanisms for organizing resources. Much of the MobileMAN project work addresses these mechanisms in the context of networks, and especially ad-hoc wireless regional networks. This part of the project focuses in on the organization of end systems, and in this part of deliverable 5 we survey existing work on peer to peer systems, and contrast them with classical distributed computing architectures such as CORBA, Web Services and Grid Services.

In the diagram below, we envisage there to be (at least) two layers of self-organization, constituting the network and the content and computational resource sharing layer.

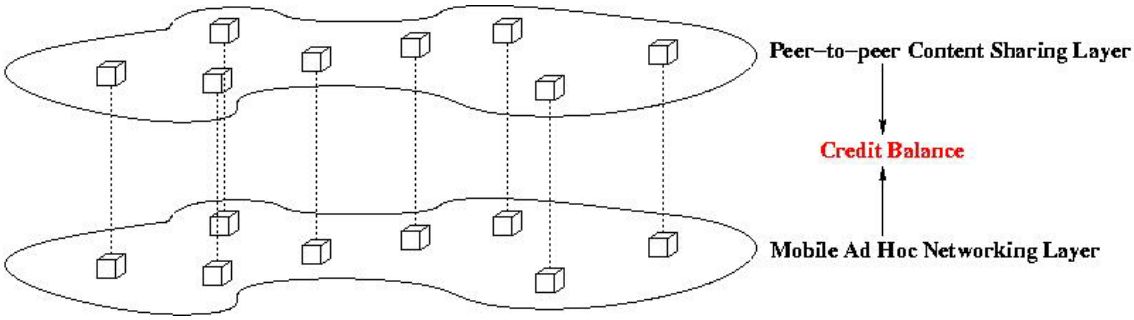


Figure 5.1: Illustration of P2P service layer and Ad Hoc Wireless Network Layer

We will be extending our resource sharing economic model to encompass these two layers. In fact we intend to compare two approaches which we call “bundled” and “unbundled”. In the former, we expect the credit system to combine payment for multiple resources: in previous work, we addressed already the combination of payment for battery and spectrum; we will extend this to include payment for content and storage. In contrast, we will also simulate and analyze the unbundled model, where two separate systems for decentralized economies of resource sharing operate fully independently: the network will operate as already described, and the content sharing p2p layer will operate in the same way, but without direct interaction. Thus there will be two independent payment systems. The objective is to see if the independent systems scenario ends up organizing itself in a similar manner (i.e. self optimizes) to the system of bundled services. In fact we would expect it to be better as the way that prices are bundled does not operate in a free-market way since the relationship between price for network transmission (spectrum), battery, and storage and content is fixed at design time, whereas in the un-bundled scenario, we expect the relative value of content and networking to find its own level.

In the rest of this document, we survey the existing P2P systems, and contrast them with more classical (less decentralized) distributed systems architectures which make assumptions about infrastructure.

Peer-to-peer (P2P) systems are Internet applications that harness the resources of a large number of autonomous participants. In many cases, these peers form self-organizing networks that are layered on top of conventional Internet protocols and have no centralized structure. Inspired by the successes of early P2P systems such as Napster, Gnutella, and SETI@home, a large and active research community continues to explore the principles, technologies, and applications of such applications.

P2P and classical distributed computing are both concerned with enabling resource sharing within distributed communities. However, different base assumptions have led to distinct requirements and technical directions [FI03] (see Figure 5.2). P2P systems have focused on resource sharing in environments characterized by potentially millions of users, most with homogenous desktop systems and low-bandwidth, intermittent connections to the Internet. As such, the emphasis has been on global fault-tolerance and massive scalability. Classical distributed systems have arisen from collaboration between generally smaller, better-connected groups of users with more diverse resources to share.

Despite these differences, the long-term evolution of classical distributed computing and P2P seem likely to converge at least in some regards, as distributed systems expand in scale and incorporate more transient services and resources, and as P2P researchers consider a broader class of applications [FI03]. Our goal in this document is to take a step towards this reconciliation of approaches, first by introducing the reader to key P2P concepts and technologies, and second by pointing out areas in which P2P results appear particularly likely to find application in the classical context. We discuss, for example, how P2P results may apply to challenges faced by the Open Grid Services Architecture [FK02], in such areas as resource discovery, scalable load balancing, and highly available storage and data distribution systems.

The rest of this section is arranged as follows. After reviewing the history of P2P computing, we examine P2P middleware and three broad application areas: storage, computation, and searching. We investigate the relationship between P2P and classical distributed computing, and conclude by looking at possible future developments that build on P2P and classical approaches. The evolution of CORBA and Web Services is now happening in the context of Open Grid Services Architecture and we mention this as it seems likely that sharing of CPU resources (as well as data) will often be carried out through OGSA.

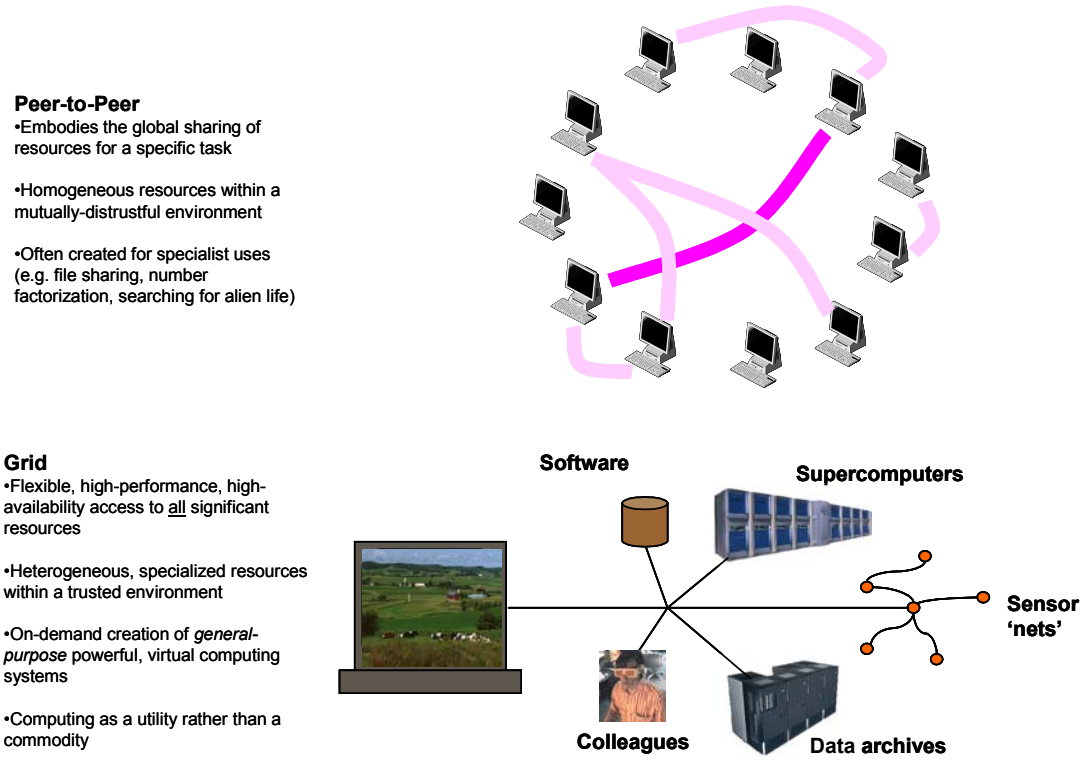


Figure 5.2: Comparing P2P and Classical Distributed Computing at a high level

5.1.1. A Brief History

P2P networking has divided research circles. The traditional distributed computing community views these young technologies as "upstarts with little regard for, or memory of, the past"; evidence supports this view in some cases. Others welcome an opportunity to revisit past results and to gain practical experience with large-scale distributed algorithms. An early use of the term "Peer-to-peer computing" is in IBM's Systems Network Architecture documents on LU6.2 Transactions, over 25 years ago. The term, which we shall use interchangeably with *P2P*, came to the fore publicly with the rise and fall of Napster [N]. Although prior systems do exist in this evolutionary phase of distributed computing (e.g., Eternity [A97]), we limit our survey to the period from "Napster 'til now" (i.e., 1998-2003).

Beyond the Client-Server Model

P2P systems can be contrasted with asymmetric *client-server* systems, in which a *server*—usually a more powerful and better-connected machine—runs for long periods of time and delivers storage and computational resources to some number of *clients*. Thus the server emerges as a performance and reliability bottleneck. To mitigate these problems, server sites may use such as replication, load balancing, and request routing, so that one conceptual server is made up of many distinct machines. A natural evolution of this thinking is to include the clients' resources in the system, an approach that becomes increasingly attractive as the performance gap between desktop and server machines narrows and broadband networks dramatically improve client connectivity.

Thus, P2P systems evolve from client-server systems by removing the asymmetry in roles: A client is also a server that allows access to its resources. Clients, now really *peers*, contribute their own resources in exchange for the use of the service. Work (be it message passing, computation, storage, or searching) is partitioned between all peers, so that a peer consumes its own resources on behalf of others (acting as a server), while asking other peers to do the same for its own benefit (acting as a client). As in the real world, this cooperative model may break down if peers are not provided with incentives to participate. We look into trust, reputation, and economically grounded approaches later.

It is sometimes claimed that P2P systems have *no* distinguished node and thus are highly fault tolerant and have good performance and scaling properties. While there is some truth to this claim, many P2P systems do have distinguished nodes, and many have performance limitations. In fact, the fault-tolerance claims are hardly borne out in early P2P systems: availability figures in Napster, Gnutella [RFI00], and Freenet [CSWH01] do not compare favorably with even the humblest Web sites. Second- and later-generation systems, however, may indeed provide the claimed functionality and performance gains. We see promising results in Pastry [RD01a], Chord [SMKKB01], and CAN [RFHKS00]; and even more recent work building applications and services over these systems shows great potential gains.

One can also compare and contrast classical client-server and modern P2P systems on another axis, namely statefulness. Despite successes with stateless servers, many Web servers use cookies, script-driven repositories, and Web services to maintain state over various transactions with a client. In a P2P system, since a peer rarely knows directly which node is able to fulfill its request, each peer keeps track of a soft-state set of neighbors (in some sense) in order to pass requests, messages, or state around the network. While soft state is also a long-recognized technique in distributed computing [50, 84, 85], classical services themselves are often inherently stateful during their (explicitly managed) lifetimes.

Yet another viewpoint from which one can dissect these systems is the use of *intermediaries*. The Web (and client-server file systems such as NFS and AFS) uses caches to reduce average latency and networking load, but these caches are typically arranged statically. P2P systems partition work dynamically among cooperative peers to achieve locality oriented load balancing. Content distribution systems such as PAST [RD01b] and Pasta [MPH02] use demand-driven strategies to

distribute data to peers close in the network to that demand. Similarly, classic systems are starting to explore the dynamic provisioning of distributed computation close to data sources, in proportion to the time and parallelization demands [TBSL01, KF].

The classical distributed systems community would claim that many of these ideas were present in early work on fault tolerant systems in the 1970s. For example the Xerox Network System's name service, *Grapevine* [MSN84], included many traits mentioned here. Other systems that can be construed as P2P systems include Net News (NNTP is certainly not client-server) and the Web's Inter-cache protocol, ICP. The Domain Name System also includes zone transfers and other mechanisms that are not part of its normal client-server resolver behavior.

Deploying Internet Services by Overlaying

New Internet network level services such as IP QoS in the form of integrated services and differentiated services, as well as novel service models such as multicast and mobility have proved notoriously hard to build and deploy in their native forms. Thus, network researchers, frustrated in their attempts to deliver new network services within the context of traditional telecommunications or Internet networks, have built experimental infrastructures by constructing *overlay* systems: developing new infrastructures using services of and layering on the existing infrastructure, rather than by complementing or replacing it.

An overlay may be as simple as a collection of static IP-in-IP tunnels or as complex as a full dynamic VPN (virtual private network). Some such systems are in use in the active networks research community. Clearly, overlaying is a relative term: The nature of the overlay and the underlay depends on the infrastructure being developed. Grid and Web service based distributed systems use overlaying to provide unified (virtualized) interfaces to all aspects of service management, with the aim of integrating underlying native platforms and protocols. In contrast, P2P systems have focused on the use of overlaying to provide an abstraction for *addressing* between peers spread throughout the Internet.

IP was originally an overlay service, implemented above other layered communications system: the PSTN, ARPANET and X.25 circuit switched networks. Indeed, this overlay model keeps re-emerging as network operators deploy faster switched infrastructures such as Frame Relay, ATM and WDM and PONS (Pure Optical Networked Systems) core networks.

In the Resilient Overlay Network system [ABKM01], sites collaborate to find a longer "IP level" path that has better properties (such as throughput or loss) at the application level, by dynamically routing via a set of dynamically created tunnels. Similar approaches have used for multicast [ST-TR] [CRSZ01, CRZ00], multimedia conferencing systems [JGJKO00], streaming media [DBF-TR], anycast [ZAFB00], and server selection [HLL99-a].

Napster

The Napster [N] *file-sharing system* allowed users to search for and download music files held on other Napster users' hard drives. When the application is started, metadata for a user's shared songs is transferred into a global directory. When other users search for a song using keywords in this metadata, the directory returns a list of clients sharing songs matching the query. The end machines (peers, in this sense) cooperate to transfer the song directly between themselves. Each takes a client or server role depending on whether they are downloading or uploading a song (see Figure 5.3).

Opinions differ as to whether Napster is truly P2P because its directory is stored on central servers. However, by distributing the bandwidth and storage requirements, the system ameliorated its initial perceived scalability and performance bottlenecks. Further, the real utility of the network—the diversity of music that was available—was certainly a property of its constituent peers.

Technically, the program suffered from a simple interface, and the poor reliability and bandwidth of other clients' connections often hampered users' attempts to retrieve songs. However, because it dramatically simplified the task of obtaining music on the Internet, Napster became immensely popular, having at its peak approximately 1,6 million simultaneous users [WN01]. Over time, Napster's centralized directory became both a severe bottleneck and a single point of failure for legal, economic, and political attacks; and Napster was eventually shut down by court order for helping users infringe copyright.

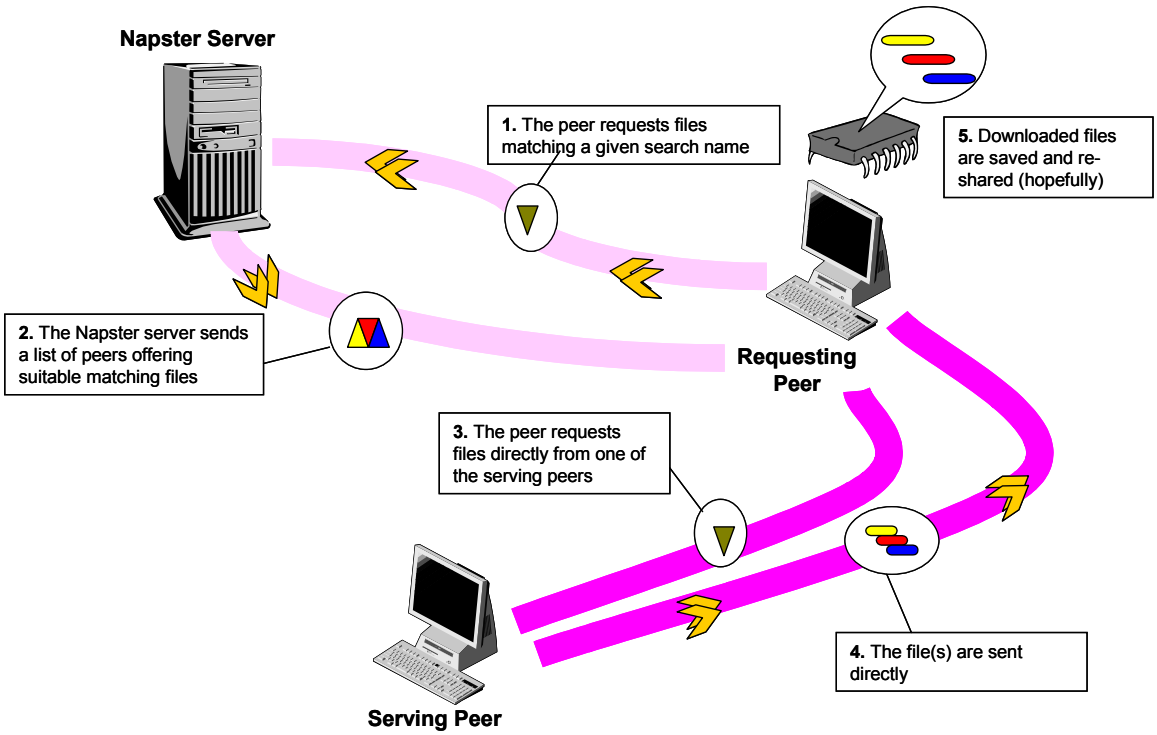


Figure 5.3 : Napster, an example of a centralized P2P system

In Napster's Wake

Napster's success was attributable to online music sharing being a "killer application." Moreover, it demonstrated the potential in harnessing client resources to satisfy their need for a service. With the demise of Napster, there arose a desire within the music-sharing community for a fully decentralized service that would not be susceptible to a similar legal attack. The projects that rose to the challenge stimulated important technical developments in distributed object location and routing, distributed searching, and content dissemination.

The Second Generation: Full Decentralization

Gnutella [RFI00] is a distributed search protocol adopted by several file-sharing applications that dispensed with the centralized directory and instead broadcast search queries between a peer's neighbors. Despite measures to limit and restrict queries, several studies and user experience found that the volume of query and control traffic caused excessive network load, limiting the size of the network, the chance of satisfying a given query, and the amount of a client's bandwidth left for actual file transfers.

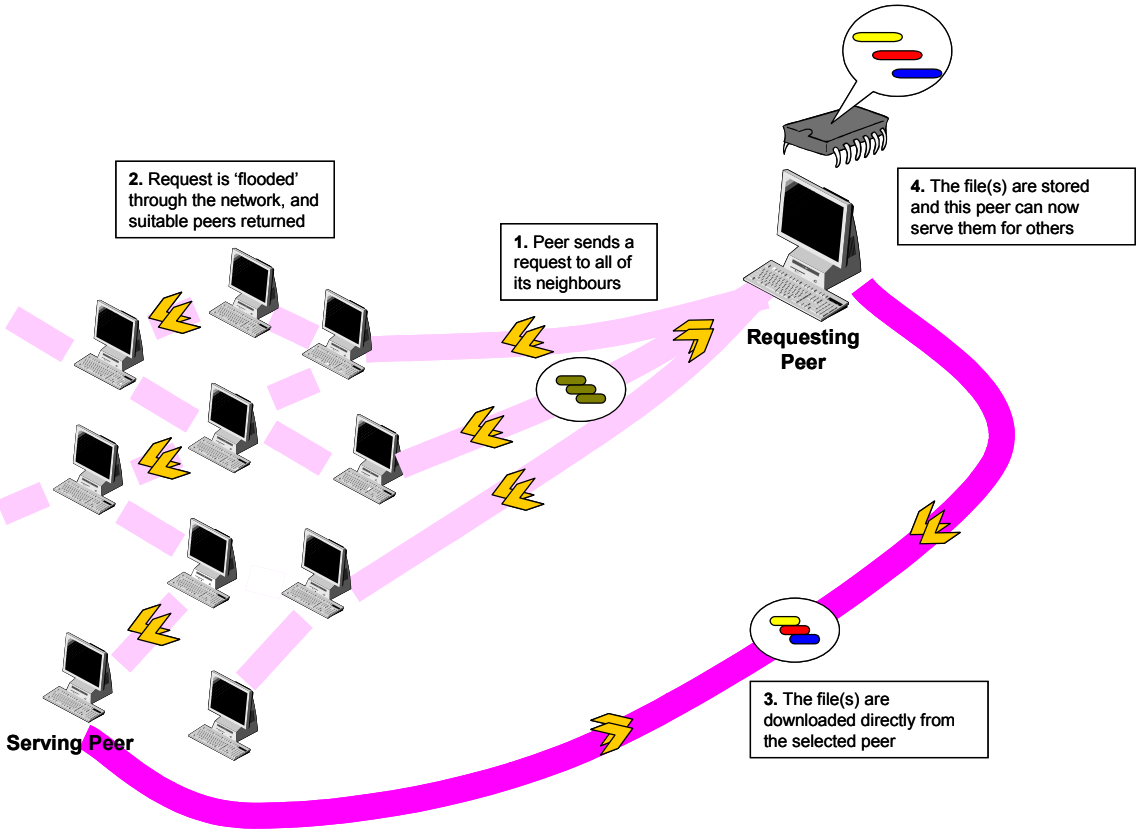


Figure 5.4: Gnutella, an example of a fully decentralized P2P system

Other systems for locating content, including Freenet [CSWH01], added mechanisms to route requests to a node where the content is likely to be, in a best-effort partial partitioning of the networks' content. Systems for file sharing such as Kazaa [LRW03] as well as recent Gnutella evolutions [SRU] added structure to P2P file-sharing networks by dynamically electing nodes to become *super-peers*, caching and serving common queries or content. These schemes take advantage of the observed Zipf-like distribution of object popularity and mitigate the difficulties of passing queries through hosts on high latency, low bandwidth dialup connections.

The Third Generation: Efficient Routing Substrates

Although the range of applications for P2P techniques remained limited by the end of 2001, a common requirement had emerged. In order for each peer to make a useful contribution to the global service, a reliable way of partitioning workload and addressing the node responsible was needed. Further, the emphasis on scalability, and the corresponding observation that in global-scale system peers will be joining, failing, and leaving continually, required that these functions be performed with knowledge of only a fraction of the global state on each peer, maintained with only a low communication overhead in the underlying network.

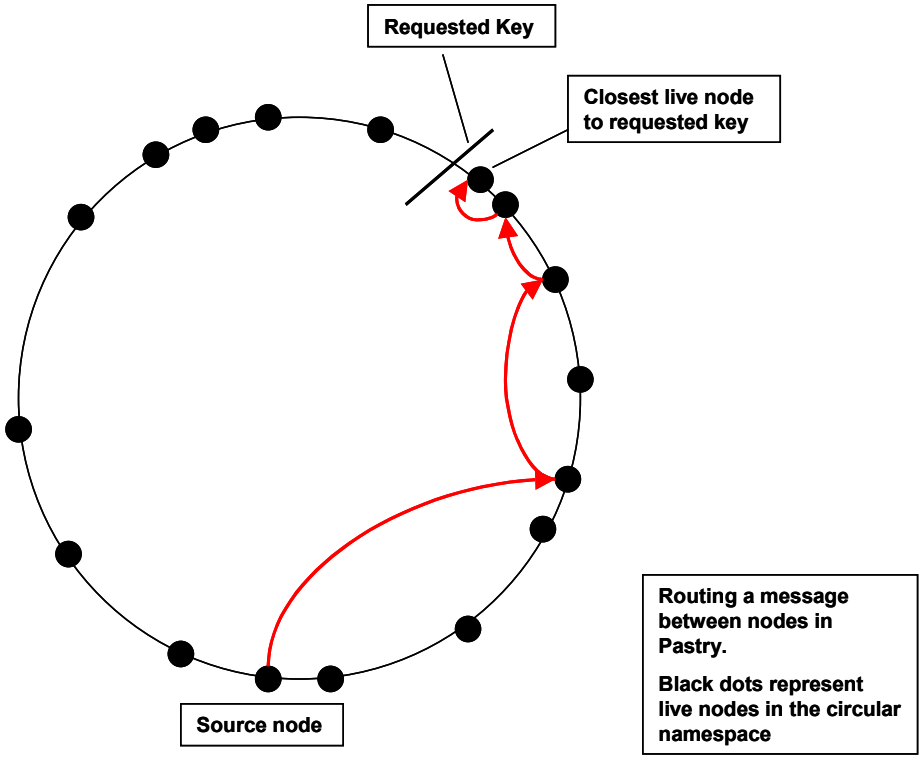


Figure 5.5: Routing a message between nodes in Pastry, a distributed hash table

This architectural separation inspired a generation of *P2P routing substrates* that provided a distributed message passing, object or key location service. The most popular approach adopts a distributed hash table (DHT), in which nodes are assigned a unique pseudo-random identifier that determines their position in a key space (see Figure 5.5). Messages are routed to points within the same key space and are delivered eventually to the closest node. According to the way in which applications use this service, a message destined for a given key represents a request to provide a given service with respect to that key. As requests' keys must be mapped on to the key space pseudo-randomly (usually using some secure hash function such as SHA-1 [FIPS180]), DHTs offer effective partitioning of the work between peers. Different variants of this basic approach differ as to the structure of information on nodes and the way messages (or sometimes requests for routing information) are passed between peers

The presence of DHT substrates offering routing services, node management, and a simple interface led to a rise in the number and variety of P2P applications. Systems for event dissemination and overlay multicast, file archive, file systems, and replacements for DNS have emerged.

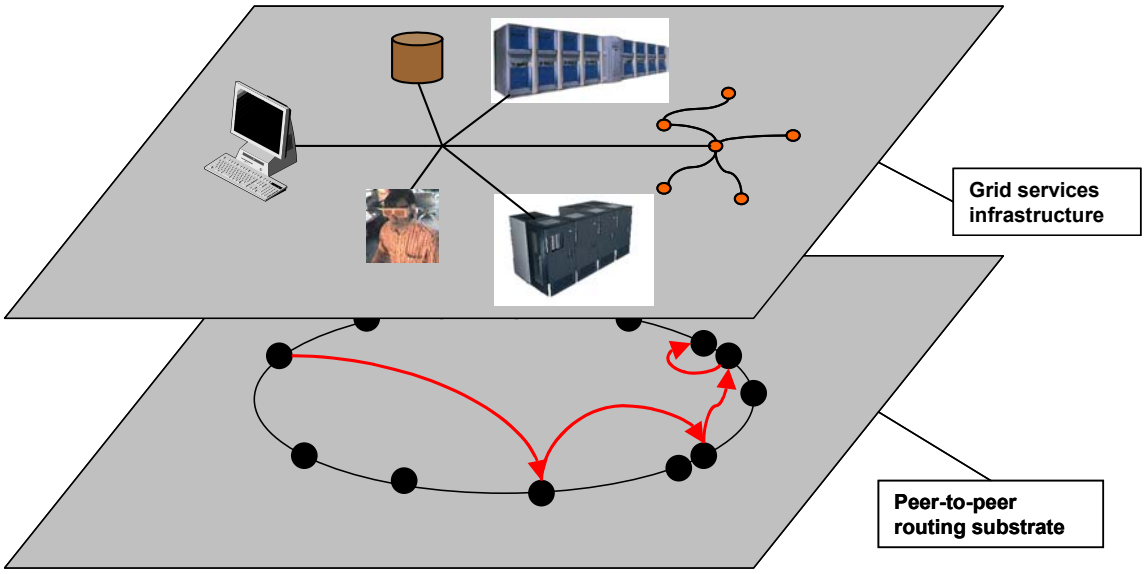


Figure 5.6: ***A P2P computer?*** *One seeks to combine the varied resources, services and power of Grid computing with the global-scale, resilient, and self-organizing properties of large P2P systems. A P2P substrate provides lower-level services on which to build a globally distributed Grid services infrastructure. Issues such as trust, which classical distributed computing assumes but are lacking in P2P systems, need to be managed between the layers.*

Future Directions for P2P Systems

P2P systems are still an active area of research, and progress is steady. We outline below technical issues facing the research community, before describing them and their application to traditional or classical distributed systems architectures in more detail in Section 5.1.2.

While DHTs introduced an essential split between P2P middleware and applications, they have limitations that are providing an impetus for more flexible schemes. Further, each proposal for a new routing substrate contains convincing evaluation results from large-scale simulations, but no Internet deployment has tested their properties under real-world conditions—with respect to failure and latency, in particular. Such analysis will play an important part in directing research.

The scale of P2P systems means that participants are typically individuals and organizations without an out-of-band trust relationship. This key characteristic is not currently shared by classical distributed computing but takes on increasing significance as such architectures scale up. This property generates interesting work in the area of trust, reputation systems, economic incentives, and detection of malicious participants.

Indeed, many of the lessons learned from studies of second-generation deployments concern human behavior. Most P2P applications rely on a cooperative model of node interaction. Participants join the network and use the service: in return, they are expected to contribute their own resources, although doing so yields no direct benefit to them. Even the mutual benefits of cooperation will not stop people defecting [AH01], and thus incentives through economic [W02] and trust [MT02] models form an important part of ongoing research.

Much progress has been made in the security and censor-resistant aspects of some applications [RW, B-TR], including an important general result in the impossibility of preventing *pseudo-spoofing* [D02] (owning more than one virtual identity) without a trusted link to a real world identity.

As P2P computing matures, we will see a diversification in its applications. As classical systems scale up and P2P techniques begin to capture shared use of more specialized resources, and as users are able to specify location, performance, availability, and consistency requirements more finely, we may see a convergence of techniques in the two areas. We describe this view further in Section 5.1.2.

5.1.2. Applications

We partition P2P projects into routing substrates and the main classes of applications that run on them: systems for storage, computation, and searching.

Routing Substrates

Routing substrates are P2P middleware that facilitates communication between and management of a network's constituent nodes. We categorize these substrates as *unstructured* or *structured*, the essential difference being whether the neighbors that each peer maintains are organized in such a way as to allow deterministic location of a piece of content or a node.

Unstructured Routing

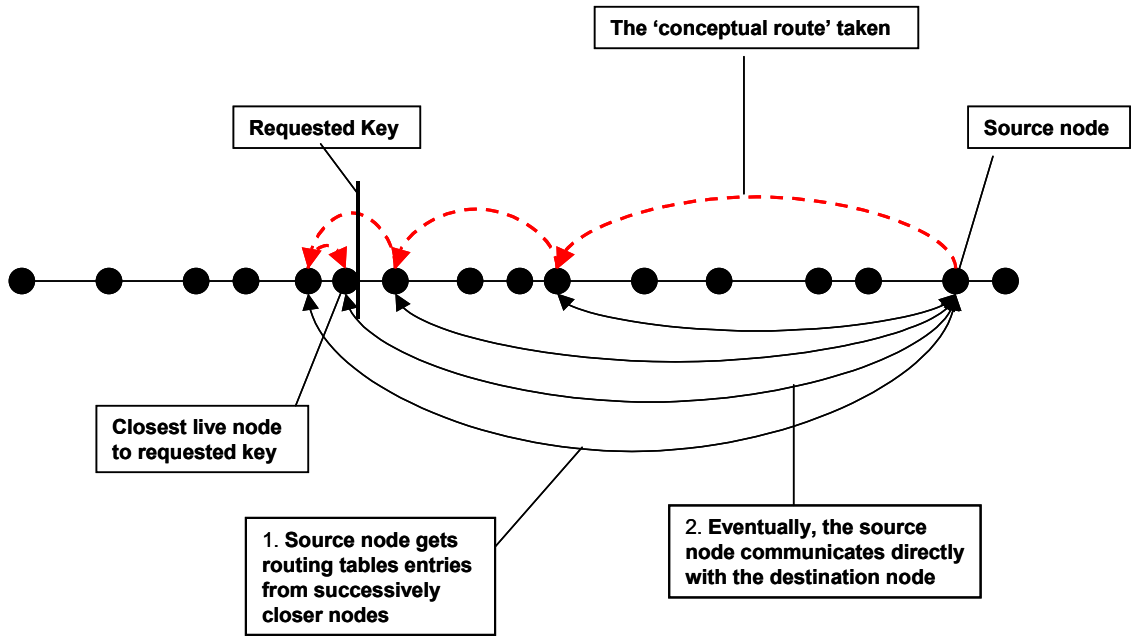


Figure 5.7: Routing a message between nodes in Kademlia, a distributed hash table (DHT). ***The key space is acyclic, and the source node locates the node closest to the requested key by successively learning about and querying nodes closer to it. The dashed line represents the route that Pastry would have taken.***

When joining a P2P network, a new node needs knowledge of at least one peer already present in the network from which to obtain its initial routing table entries. Nodes in unstructured systems tend to maintain these same neighbors, replacing their entries only if the node in question has failed. Hence, the topology of the network grows in an arbitrary, unstructured manner; it becomes difficult to bound the maximum path length and guarantee even connectivity between groups of nodes. This situation impacts performance and reliability: Unintentionally, some nodes may become bottlenecks.

So far, such systems have not allowed efficient searching (either for keys or for more complicated metadata queries). Gnutella [RFI00] uses a flooding-based search in which a query is broadcast to each of its neighbors, which in turn pass it on to each of their neighbors; each peer tracks the queries that it has seen to prevent routing loops. Unfortunately, the buildup of traffic from each query is exponential—to such an extent that unless the search breadth and depth are low the system will not scale. More efficient schemes have borrowed from conventional data structures, including iterative deepening techniques [YG02] (incrementally considering the nodes at a given number of hops from the requester) and random walks [LCCLS02].

A concept of *direction* seems essential, however, in pruning the potential search space and in routing efficiently. In Freenet [CSWH01], a publishing network where peers cooperatively participate in caching and retrieving documents, each node maintains a data store that locally caches the data and key associated with a document, and also the node from which it was originally obtained. Entries for evicted documents are maintained, but without the attached data. On receiving a request for a key where no cache entry exists, a node finds the entry for the document with key *numerically closest* to that sought, and forwards the request to the node from which it was obtained. In this way, Freenet nodes over time may come to specialize in portions of the keyspace, and other nodes' knowledge of this gives searches direction. Because this scheme relies on uniform local knowledge of the keyspace, however, it suffers poor worst-case lookup performance [H01] and cannot guarantee success.

DHTs and Structured Routing

Structured routing substrates (approximately synonymous with DHTs at present) organize their peers so that any node can be reached in a bounded number of hops, typically logarithmic in the size of the network. Although subtly different, all of the main schemes operate similarly. Pastry maintains per node routing tables organized by the length of the entries' shared address prefix. Tapestry [ZKJ01] nodes each maintain an inverted index organized, again, by prefix. Kademlia [MMK02] routes according to distance between IDs using the XOR metric. In Chord [SMKKB01], each peer is arranged in a circular ID space and maintains links with its immediate predecessor and successor, and a number of "chords" to other nodes whose distances are arranged in an exponential fashion. CAN [RFHKS00] uses several hashes to map into a multidimensional ID space; queries are passed along the axes of this space.

The class of DHTs derived from Karger's work on consistent hashing [KLLLLP97] and Plaxton's distributed data structure [PRR97] are mathematically described in [ADS02], which gives upper and lower bounds on the query time in DHTs. The results are similar to the empirical results from CAN and Chord. As an alternative underlying technique, Distributed Tries [FV02] use a trie, and so the same `lookup(key)->value` as DHTs, but they may offer lower average-case message overhead. Each node holds a trie: By using a backtracking search, they query nodes that are known to contain other parts of the trie, which in turn may return the object or more up-to-date or detailed parts of the trie. In the worst case this scheme degenerates to broadcast search.

An inherent difficulty with DHTs relates to the uniform partitioning of work. Since data (be it content, blocks to store, or multicast topics) is associated with pseudo-random keys, a user cannot control which peer is responsible for a particular data item. Locality of reference is broken: an essential property if P2P computing is to offer the performance seen by conventional client-server models. Significantly, though, SkipNet [HJSTW03] offers a hybrid architecture based on skip lists that can route and store within explicit administrative domains.

Content Distribution and Storage Systems

P2P techniques first found their niche in *file-sharing systems*. We distinguish such applications from *distributed file systems* (e.g., NFS [SGKWL85]). The former allow users to obtain specific well-defined content, usually based on a metadata search; the latter expose local file system

hierarchies to remote users, may be writable, and may implement access control or consistency semantics. We also describe *distributed archival storage systems*, in which insertion and retrieval operations are coarse-grained (i.e., documents at a time) and storage is durable, long-term, and often focused on censor-resistance or anonymity. Additionally, we consider storage requirements in distributed processing applications.

File Sharing

Recall that Napster, while partly centralized, applied P2P techniques to file sharing by distributing the high-bandwidth requirement of transfers: files were passed directly between peers. However, in this set up the performance and reliability of file retrieval is dependent on the peer, with which a user is transacting, preventing the system implementing any quality-of-service guarantees. Frequently, transfers may be aborted when the sender cancels or disconnects or when the network partitions. The rate at which transfers proceed depends on the relative positions of the endpoints, their latency and their bandwidths. *Swarm distribution* in systems such as Kazaa [KAZAA] improves load balancing and reduces a transfer's dependence on individual nodes by naming the file by the secure hash of its contents. If a transfer aborts, another peer sharing the same file may be identified and the transfer resumed. Further, subdividing the file into portions and naming these allows different parts of the file to be transferred from multiple sources at once, improving performance for well-connected machines. By allocating small portions to dial-up peers and larger portions to others, each node contributes according to its ability.

Recent P2P applications based on DHTs offer content streaming and effective content dissemination by replicating data in proportion to demand for it, close in the network to that demand. These applications include CFS [DKKMS01] and Pasta [MPH02], file systems, and SplitStream [CDKNRS03] for streaming media.

The difficulty of performing arbitrary metadata searches and obtaining deterministic results in a fully decentralized environment is limiting file sharing. Many systems are restricted to specific areas (in particular media and software distribution) where the search-space for users is well defined; typically, users "discover" content by out-of-band means. Certainly, adoption of a consistent metadata framework such as Dublin-Core [DCMI] is important for progress.

Archival Storage Systems

The Eternity service [A97] proposed a design for a system that offers censor-resistant, durable storage to users for a specific time period by replicating and scattering data across servers in a distributed system. While Eternity is a design and does not specify any of the mechanisms by which P2P applications are now characterized, its ambitions were reflected in many early P2P systems.

Free Haven [DFM00], a document storage system, implements a variation on Eternity. Its primary aim is to offer anonymity to publishers of documents and to provide plausible deniability to server owners. Documents are stored on a P2P network of mutually-distrustful servers, called a *servnet*: queries and publication requests are broadcast to the whole group. Much early work on the nature of peer behavior is present in Free Haven's design. It makes pairs of servers mutually accountable for files that they store using a buddy system, and uses a reputation system to report complaints from "buddies": Servers over time develop trust for other servers according to their reputation. An economic system of trading reputation capital for resources on other servers provides an incentive to participate and minimizes the damage caused by individual malicious entities.

PAST [RD01b] is an experimental archival utility built over the Pastry DHT. Storage and retrieval operations are performed at the granularity of whole files. No human-readable naming scheme is supported; rather, a fileID associated with the insertion must be passed by other means. Inserted files are immutable until withdrawn by their owner.

Global-Scale File Systems

Network file systems were one of the first great successes of client-server distributed systems. Sun RPC and NFS were ubiquitous from the mid-1980s in research, and education labs and many small organizations use Samba to share storage resources.

Recent distributed file system designs aim more ambitiously to present a unified view of storage resources of any Internet-connected system, while offering secure reliable storage, better-defined, application-variable concurrency guarantees, and efficient content-distribution. The current cost of management and organization of storage tends to exceed the cost of the physical media. This situation has led to the adoption of P2P techniques for managing large, dynamic sets of unreliable hosts, replacing to brittle, location-dependent mutual client-server systems (such as NFS) and high-maintenance organization-centric client-server systems, such as AFS.

The Cooperative File System (CFS) [DKKMS01] is implemented over Chord. Files are split into fixed-size blocks, which are identified by their secure hash, then distributed to nodes. Storage can be guaranteed for a set time period enabled by per node storage limits based on IP addresses. Users arrange files hierarchically in a "file system," which forms a per-publisher, decentralized namespace. CFS offers coarse-grained file mutability; but since each publisher manages its own file system, collaborative file manipulation is not possible. No cache consistency or concurrent update control scheme is proposed.

Pasta [MPH02], a prototype P2P file system operating over Pastry [RD01a], offers a persistent, mutable shared storage and content distribution service to a potentially large collection of mutually distrustful users. It integrates closely with local file systems through a loopback NFS. Users store data as a series of immutable blocks, referenced through mutable index blocks that store fragments of decentralized namespace. By replicating and maintaining blocks across peers, Pasta offers persistent and reliable storage. Widespread localized caching in proportion to demand provides an efficient content distribution system by preventing hot spots and migrating data to the source of requests. Files are split into blocks in terms of their contents, to exploit commonality between sections of files; this arrangement allows Pasta to store different versions of a file efficiently and to modify them on a copy-on-write basis. A scheme of storage quotas, enforced in a distributed fashion, regulates consumption. Ongoing work on Pasta focuses on a scheme whereby privately owned data may be collaboratively modified by untrusted users. Through namespace overlays, users may appear to share and organize each other's stored data; modifications to files or namespace are seen as in a versioning file system, and third parties form a view by choosing which modifications to trust.

OceanStore [KBCE00] is an infrastructure for providing persistent data storage in a global-scale ubiquitous computing environment, although its current prototype, Pond [REGWZ03] shares more with the above systems. It uses a variant of Plaxton's distributed hierarchical data structure [PRR97] to locate stored data efficiently. The system caches data widely for performance and availability and performs distributed cache consistency management. For each file, a primary tier of replica-carrying nodes use a Byzantine agreement protocol [CL99] to commit modifications. A conflict resolution scheme resolves concurrent updates as in Bayou [TTPDSH95].

Data Access in Distributed Computational

Requirements for data access and movement in distributed computation, where computation is performed at a remote site, may motivate applications that combine techniques from both content distribution and P2P file systems. Several P2P file system designs, including OceanStore [KBCE00] and Pasta [MPH02], might be suitable for the task—and also offer fault-tolerant, highly available long-term storage. They provide schemes to support location-independence of data on a global-scale, allowing data to be gathered from a variety of sites and sensors or from dynamically established caches, while being named in a unified way. Although many systems offer only limited

concurrency semantics, this is all that most file access patterns require. Such systems also use conventional file system interfaces, necessary to minimize rewriting of applications for a distributed (but not decentralized) setting.

Distributed computational applications require flexible caching of input files and output files for rerunning computational tasks with different parameters or for comparing results. Systems such as CFS and Pasta incorporate caching schemes suitable for this purpose. The size of some datasets, however, may necessitate files being stored across different nearby sites, then streamed; moreover, streaming of diagnostic data back to the client site is essential for tracking progress. Data movement issues such as these may benefit from swarm distribution and other techniques developed for P2P file-sharing applications.

Distributed Computation

Several hundred million personal computers and workstations are now on the public Internet with hundreds of MIPS going unused every second. Attempts to exploit this vast processing resource, however, have been limited for several reasons. The granularity of computations required in many applications is small, individual nodes are unreliable, and external code and data distribution is hampered by relatively poor latency and bandwidth. The computations that make sense are those that can be broken into many relatively small pieces that require long individual processing times but perform little communication.

Numerous P2P computing systems have been developed and have seen considerable success for computations that are able to use large numbers of processors that may vary in job throughput but are homogeneous in that they offer no specialized functionality. “Philanthropic computing” projects continue to thrive in such domains as large-scale signal processing (e.g., SETI@Home [SETI]), genome searching [UD], Astrogrid, and code cracking. Since many peers in such systems are home users with slow and intermittent connections, such projects have typically involved highly parallelizable tasks. Recent consideration of internode proximity in P2P systems may, however, lead to the use of clustering, allowing more tightly coupled computations of the sort commonly run on classic distributed systems. These technologies have also been applied in corporate settings, where higher connectivity allows for more tightly coupled applications.

Distributed Searching

In addition to key-based searching and filename metadata searching, researchers have tried to offer *generalized metadata search functionality* over DHTs. Systems having this functionality operate within the context of resource discovery and may suggest directions for similar service discovery mechanisms in traditional distributed computing. Examples include PlanetP [CN02], pSearch [TXM02]/Sedar [MTX02] and Multi-Dimensional Pastry [SH03]. These systems use the vector space model to represent documents, which maps complex searches to similarity searches in a vector space. Searches are carried out in pSearch by using a CAN [RFHKS00] network to route requests, and in PlanetP by summarizing using Bloom filters [B70] and “gossiping” using the “name dropper” algorithm [HLL99-b]. Multi-Dimensional Pastry (an extension to Pastry: see Figure 5.6) represents each query dimension as a Pastry ring, and uses Bloom filters to summarize each dimension at various levels. These filters allow a query to combine *ranges* of each dimension by union and intersection, instead of being a simple hypersphere search.

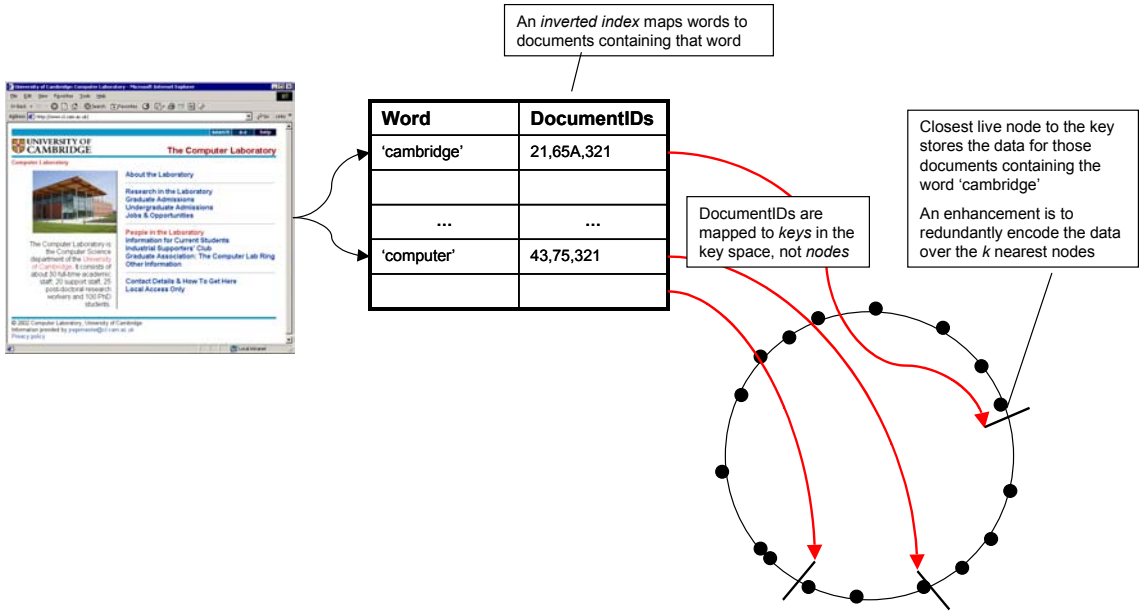


Figure 5.8: Distributing an inverted index over a distributed hash table, such as Pastry's circular key space. An inverted index maps keywords to documents containing those words. A web page containing certain keywords is found by intersecting (using bloom filters passed between peers) the sets of possible documentIDs

Developing P2P Software

The symmetric nature of P2P software makes such software systems harder to write than client-server systems. One must pay great attention to synchronization effects: one cannot separate concerns as in client-server systems. (However, the separation of concerns that can often be achieved between routing substrate and application can reduce development complexity.) In addition, the P2P programmer must cope with the types of erroneous requests that only server (rather than client) application programmers have to deal with. Thus, P2P programming is currently an expert systems-programmer task.

Nevertheless, the widespread use of Java in some research projects, combined with the observation that there are enough common components in P2P development activities, has motivated development of a few toolkits. One example is the JXTA toolkit [SUN02] from Sun. The SEDA toolkit [WCB01] provides a framework that supports event-driven programming in Java. SEDA has been used in OceanStore [KBCE00] to build event-driven systems. And of course OGSA [OGSA], as realized for example in the Globus Toolkit, provides many relevant primitives.

5.1.3. Properties and Issues

We now explore various properties of P2P computing, some aspects of which are present in current systems, while some still the subject of ongoing work. In each case, we discuss the relevance to the evolution of classical distributed systems computing.

Harnessing Resources

The P2P fault model of an unreliable infrastructure and mutually distrustful participants leads P2P systems to treat resources as *homogeneous* and peers as *individually dispensable*. Therein lie many of the strengths and weaknesses of the approach. DHT designs embody these assumptions. Any

node is equally likely to be responsible for one particular key, so it is assumed to be equally suitable to carry out a task related to it. Nodes carry similar numbers of keys, so it is assumed that their resources for storing or managing these keys are also equal.

Peers, however, are unlikely to have similar resources—either in quantity or in quality. Nor are peer resources likely to have similar reliability characteristics. Systems that recognize these facts benefit in terms of performance and availability. For example, super-peering in file-sharing systems takes advantage of well-connected nodes to implement distributed caching and indexing schemes.

Unfortunately, properties of a DHT routing substrate can hinder an application's ability to recognize heterogeneity of resources. In CFS, for example, each node offers a globally fixed amount of disk space as storage for blocks inserted by other nodes. Nodes with substantially more spare capacity can run separate *virtual nodes* on the same physical machine, each offering the same fixed unit. CFS employs routing table "snooping" to avoid increased lookup path lengths (because virtual nodes effectively increase the size of the network), but this approach weakens assumptions about independent failure of nodes.

Traditional distributed systems, on the other hand, tend to comprise fewer, more varied, more specialized resources; each resource's properties are described and published, and individual work units are matched to a provider based on their own description.

User Connectivity

The nature of a peer's network connection is an essential consideration when designing P2P systems for practical use in established user communities. The problem is twofold: In many contexts, mean connection quality is low but has also has high variance, due to differences between dial-up, broadband, and connections from academic or corporate networks. Thus, the scope for generic internode communication is severely limited, and applications must consider the heterogeneity of their peers' connections.

Another source of difficulty results from peers that cannot accept incoming connections, either because their ISP uses NAT and as such have no externally-recognized IP address, or because they are behind an separately administered firewall. Additionally, most broadband connections have unequal provisioning of upstream and downstream bandwidth, and many impose "caps" on permanent connections. These factors all complicate attempts to understand routing behavior in real deployments.

Existing distributed systems tend to comprise participants connected by well-administered, reliable academic networks. However, as generic services begin to incorporate more diverse peers, these issues may become more important.

5.2. Collaboration and Trust

In client-server and P2P systems, participants constitute a virtual organization defined by common interests. However, the nature of these common interests and associated trust relationships can span a broad spectrum, with traditional systems tending to feature stronger "intra-VO" bonds than in P2P systems. The following features tend to be more common in P2P systems than in classical systems:

- *Nonexistent trust relationships.* The owner of one peer in a P2P system frequently has no real-world knowledge of the owner of other pees.
- *Poor administration*. Peers in P2P systems are more likely to be *run by individuals than corporations.* This property has implications for uptime and reliability, leading to the

observed power law uptime distributions for Gnutella and the relatively small number of hosts that are reliably available.

- *Operated with no prior agreement between peers.* Consider a typical traditional computing task, such as processing a large amount of (possibly confidential) data. Before submitting the task to a compute service, the submitter (consumer) often agrees on some terms and conditions with the service provider.
- *Composed of nodes that act in their own interests.* In P2P systems, one must assume a mutually distrustful environment and must assume that without incentive to participate, a node will use the service without returning anything to it. In fact, such free-riding is the norm rather than the exception in Gnutella.
- In particular, while many P2P services rely on a cooperative model of interaction among nodes, they provide little incentive for nodes to collaborate.

One approach to addressing this situation is to use economic models to realign nodes' incentives to participate in P2P systems. Such models assume variable demand for services. For example, Geels and Kubiatowicz [GK02] argue that replica management in global-scale storage systems should be organized as an economy. Nodes trade off the cost of storing data blocks with the value they are able to charge for access to them. In this context, a variable demand for blocks is essential. However, whereas variable-demand properties may hold for human-valued commodities such as information stored or shared in a P2P system, such demands may not hold for routing table entries. Since DHTs typically determine the allocation of items to nodes pseudo-randomly, requests for keys will also be distributed evenly. Hence, no particular value can be conferred on any particular destination.

The lack of a scalable, low-overhead, fully decentralized digital cash system has hampered adoption of economic models. Mojo Nation [W02], a distributed file storage and trading system, used a currency "mojo" to discourage free-riding and to obtain load balancing at servers by congestion charging; but it relied on a centralized, trusted third party to discourage double spending.

Acqusti et al. [ADS03] have developed an incentive model for a distributed application that offers anonymity to its participants. They take a game-theoretic approach to analyzing node behavior and attacks in various system models. Trust is considered only as a means to ameliorate pseudo-spoofing attacks, rather than as a means to provide incentives to peers.

Aberer et al. [ADM01] have devised a system for managing trust in a P2P system using a complaint-based metric. Nodes can make "complaints" regarding interactions they have had with other nodes. A threshold technique checks whether a node is untrustworthy, based on the difference between its recommendations and the average view. This system presents a rather brittle and almost binary view of trust, however, which is often difficult to reason about explicitly. Taking a different thrust, the NICE system [LB03] aims to identify rather than enforce the existence of cooperative peers. It claims to "efficiently locate the generous minority [of cooperating users], and form a clique of users all of whom offer local services to the community." The system takes a novel approach: Rather than using economics to model trust, it proposes using trust to model expected service prices.

Many approaches to enforcing or encouraging collaboration have been proposed based on rather arbitrary measures. We believe that, instead, a collaborative service should have two properties: *avoidance*, whereby dishonest nodes are "routed around" by those using the service (which is often the desired case for a hedonistic node); and *exclusion*, whereby dishonest nodes are unable to use the service because others refuse to carry out work for them (e.g., not forwarding packets). We have developed a trust and security architecture [MT02] for a routing and node location service that uses a *trust protocol,* which describes how honest nodes perform, and a distributed, explicit trust model that allows reasoning about trust in the network. This system is resistant to a number of attacks, including collusion.

Scalability

P2P systems often add resources as they add customers. Thus, they should scale (at least at the computing and storage resource level, if not networking) linearly, or better, with the number of peers. Of course, many networks are designed assuming client-server traffic, and so it is possible that performance scaling properties may not be achieved transparently. Indeed, some claim that the "small world" models of human behavior and interaction, combined with the preferential way in which people link to known locations, lead to power law structures in connectivity.

Scalability is not a trivial consideration. While, for example, hops in a DHT may vary as log(n) with the number n of nodes in the network, acceptable access latency is constant and bounded by the user. Thus, P2P designs must make good use of proximity information to balance load between nodes and to maintain state efficiently as nodes join and fail. For file system applications, this means exploiting caching, predictive prefetching and the apparent Zipf-like power law distribution in content popularity.

Forming a structured topology is important in most P2P systems to provide bounds on latency and performance of internode communication. Approaches that structure and partition the keyspace tend to allow deterministic node location and better load balancing.

Proximity

Latency is an important consideration when routing in P2P systems. Poor proximity approximation in DHTs can result in a message traveling around the globe many times to reach its final destination. Several distributed applications aim to automate the gathering of relevant proximity information, and Zhang et al. [ZPS-TR00] assure us that point-to-point latency is constant enough over time to allow such systems to provide good approximations. Current systems include IDMaps [FJJJRSZ01], GNP [NZ02], Lighthouse [PCW02], King [GSGK02], and geographical position estimates [PS01].

Issues of "stretch" (distance traveled relative to the underlying network) become increasingly important in file sharing or storage systems in which large quantities of data must be transferred between peers. Furthermore, to minimize the load on the network and increase the rate at which data may be obtained, data must be stored near the place it is accessed. A system simply storing blocks across a DHT is at odds with this approach. Since nodes and keys tend to have pseudo-random identifiers, blocks will be assigned to a node regardless of that node's position. However, when replicating data across k neighbors, which are likely to diverse in location, a DHT that takes into account locality (and so is likely to route through the *nearest* of these k nodes) provides some means of obtaining content from nearby peers.

Load Balancing

Pseudo-random assignment of nodes and keys tends to lead to a imbalance in the allocation of keys to nodes,. With the maximum load at any node being log*(n)* times the mean load in an n node network. Of course, this analysis assumes that any resources associated with keys are homogenous, making the same requirements of their destination node.

In PAST, an archival storage system, whole files are associated with a key and inserted into an underlying DHT. Because the size distribution of files is heavily skewed, the above imbalance is exacerbated, and a complicated scheme of storage management is required in which replicas are diverted to nodes with more storage space. The net effect is an increase in the average hop count (and so latency) required to retrieve files.

Load balancing in an environment of heterogeneous resources and competing job requirements is difficult and requires a trade-off between the best allocation of job to resource and the rate at which job and resource properties are distributed.

Availability

P2P networks experience a high rate of nodes joining and leaving, both because of their scale and because of the nature of their user communities. Hence, individual peers cannot be relied upon to maintain any essential state on their own.

For purposes of redundancy, most DHTs replicate state at the *k* nodes with identifiers numerically closest to the associated key. This replication invariant is maintained by local node cooperation, despite nodes joining or failing. If surrounding nodes maintain their routing tables correctly, this offers automatic fail-over: If the nearest node to a key fails, requests are automatically rerouted to the next closest node, which will also maintain state for that node.

At the application level, erasure coding schemes have been shown to offer the same availability for an order of magnitude lower overhead compared with deployed replication schemes. Data is encoded into *n* unique fragments; the coding ratio determines the proportion *m*/*n* of unique fragments that are required to recover the original data. Since each fragment is unique, however, a simple local maintenance scheme does not suffice to maintain a data item's availability as different nodes fail.

The nature of network failures in the Internet is an important consideration for any practical system. Networks tend to fail along administrative boundaries, close to users, because of individual router or link failures. Conversely, individual peers are assumed to fail randomly (although other patterns also occur). While most DHTs perform suitably under the latter failure model, a network failure of the former type tends to render most of the keyspace inaccessible (most likely, all of it, unless locality is taken into account when choosing routing table entries).

SkipNet [HJSTW03] is a routing substrate based on a data structure similar to a *skip list* and offers DHT-like data distribution, load balancing and routing at various administrative levels. Keys specify using a reverse-DNS notation the domain of peers over which they may be placed allowing items to be distributed solely across nodes in the same organization. Furthermore, a request for a key specifying a specific organizational domain will never be routed through nodes outside the same domain; thus, access to that data is maintained even if a network failure separates that organization from the rest of the Internet.

Little attention has been paid to the effect of network partitions on systems in which partially or wholly independent fragments of systems are formed, update their own state, and then later rejoin. Quorum systems (e.g., [CL99]) have been used to enforce state consistency between peers updating replicated data [KBCE00], but the overhead of these schemes is prohibitive for use across a whole network. Instead, techniques from the literature on reconciliation of divergent file replicas may inform the semantics of P2P systems under network partition. One exception is Ivy [MMGC02], a P2P file system where participants each maintain a log of the updates they make to the global state. Log entries have per-log sequence numbers and are tagged with version vectors, detailing the latest log entry seen for each other log in the system. Later, as each update carries sufficient information to determine the exact view of the global state at that time, conflict resolution can be performed to combine these states.

Anonymity and Censorship-Resistance

Some P2P systems offer privacy by masking user identities. Some go further and mask content so that peers exchanging data do not know who delivered or stores which content. True anonymity is typically two layer, requiring some form of anonymous IP routing system (e.g., onion routing) and application layer mechanisms to protect privacy. Eternity and subsequent P2P file systems withstand censorship by several means.

Partition: A file is split into component parts to ensure that no single site carries the whole file, and a denial of service attack has to run over multiple sites. Later systems made clever use of

techniques such as Rabin fingerprinting or other techniques for finding common elements of objects. These techniques can also exploit overlapping content between multiple files to reduce storage costs.

Replication: Blocks of a file are replicated over multiple sites to provide higher availability. This strategy can be combined with locality information to reduce latency and increase file sharing throughput, at some cost in terms of consistency in update.

Encryption: File blocks are encrypted to ensure not only that disclosure is unlikely but also that a node can deny knowledge of the actual content it carries. Again, P2P exploits mutual benefit: The argument is that "I might not approve of this content, but I approve of the ability to hide my content in a set of peers, so I will live with what I do not know here." This is often termed "plausible deniability" and is used by service providers to align themselves with the "common carrier"' defense against legal liability for content, as telephony and postal providers can do.

"Anonymization": The identities of request sources and sinks are masked, thus protecting *users* from the potential *censor* or unsavory *agency*. Location information must be masked as well as identifiers, as otherwise, a traffic analysis may effectively reveal identities, so some form of onion routing is also usually required.

5.2.1. Future Directions

We examine several areas in which additional research questions can be identified, based on problems and shortcomings in current P2P systems. In describing these areas, we try to keep history in mind and to consider how some problems relate to those past, present, and possibly future in distributed computing.

Sharing Computation

Two systems epitomize the sum use of P2P technology thus far: the Napster file-sharing and the SETI@home coarse-grained distributed computing system. P2P systems have been successful in supporting the former, but the latter represents the tip of the iceberg in distributed computing. Figure 5.9 shows an apparent continuum between P2P systems and traditional distributed computing.

A critical difference between sharing files (and file transfers) and sharing computation is that the former are static, and hence are partitioned easily: Transferring one file is independent of another. Computation is notoriously hard to distribute, yet there are some interesting cases that are well understood; these are situations with a low communication overhead, compared with the computation required.

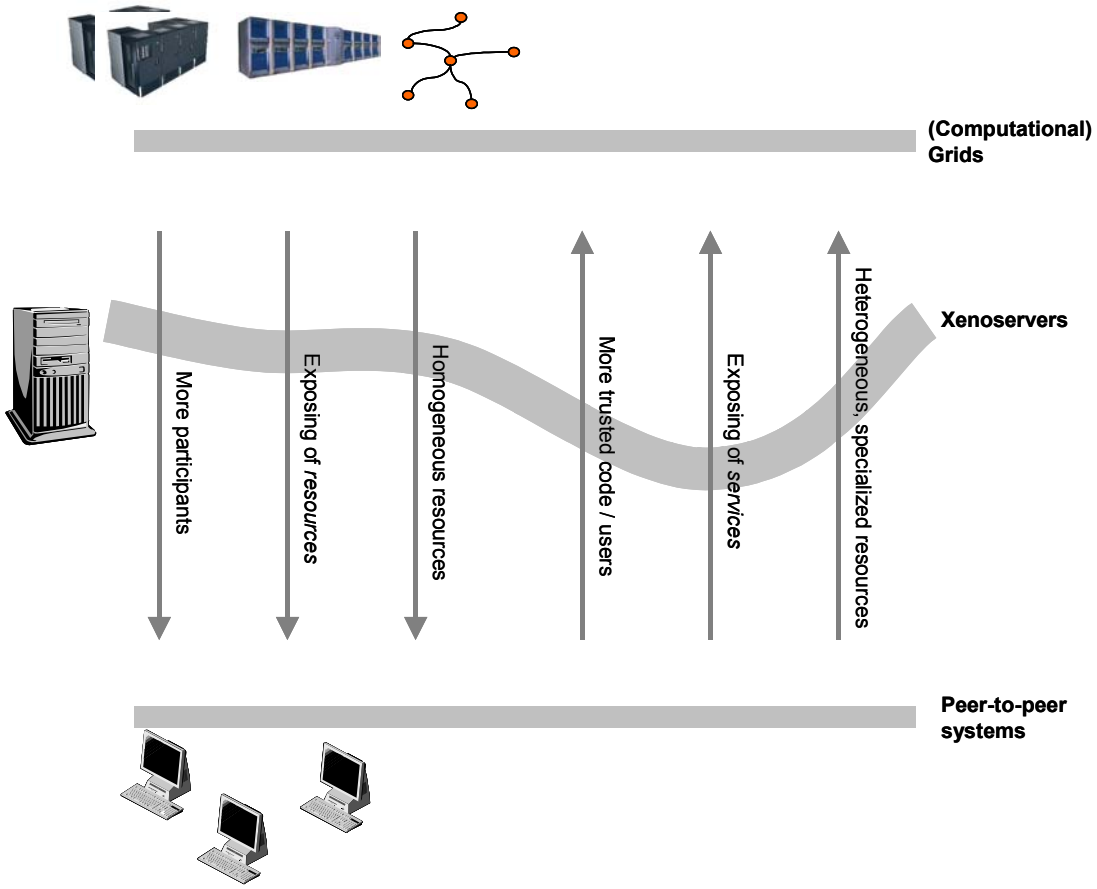


Figure 5.9 : Apparent continuum between traditional distribution and P2P computing styles

Work is required in two areas in order to broaden the range of computational tasks that can be treated with a massive distributed P2P system. First, the infrastructure needs to handle tightly-coupled distributed computation better, by exploiting self-organizing properties, staging and timing, and using innovative data transfer schemes to minimize communication overhead. Second, algorithms should be *designed* to exploit P2P properties, by using, for example, asynchronous schemes to reduce dependency on low latency communication links.

Auditing and Accounting

SETI@home relies on users simply volunteering their CPU resources. Introducing an economic model whereby resources are bought and sold adds a new complication: accounting for their use.

The motivations for file and CPU resource sharing are not the same: It is not clear how to apply the mutual benefit arguments that work for file sharing to cycle sharing. The network connections of file sharer typically have some degree of separation in capacity provisioning that allows them to upload and download independently; to some extent, cooperation does not disadvantage them. The bursty nature of interactive use, however, means the same is not true for any third-party sharing of a local CPU. Fine-grained accounting for resource sharing is required to limit, prioritize, and arbitrate between sharers. But how does one measure and enforce relative resource usage between different services?

The Xenoserver project takes one approach. Its goal is to deploy machines running a *hypervisor* that performs accounting of commodities such as CPU cycles and network bandwidth. Principals may use these machines to deploy high level services, paying server operators for each service's use of low-level resources. In turn, other principals may be charged for accessing the high-level services. Xenoserver might be used a platform for both P2P and less decentralized services, but their accounting scheme has direct application in all distributed computation systems.

P2P systems make accounting difficult in general, because of the coarse nature of jobs, the mutual distrust assumed between participants (the lack of an out-of-band trust relationship), and the possible short-term network presence of pseudonymous participants. Accounting in a non decentralized environment is easier: one possible approach and the associated issues are investigated by GSAX [BHHLM01] (Grid Services Accounting eXtensions), an extension to the OGSA standard. Accounting, and ultimately charging for, services at the application level suit the Grid computing paradigm because OGSA embraces a notion of *virtualization*, exposing resources as services.

Local Solutions to Achieve a Global Optimum?

Recent P2P systems select preferred neighbors to fill routing table entries by *proximity*. While it has been shown (e.g., in Pastry) that making peering decisions locally can improve global routing properties (i.e., by minimizing "stretch," routing distance relative to IP), such an approach is error prone. More important, it leads to a succession of local optimizations, rather than a solution *optimized* for global performance. In the future, results from location services may be obtained and cached locally to improve proximity estimates and inform routing decisions.

We have a long way to go before we can provide performance guarantees within large-scale distributed systems, whether via adaptive techniques that responds to congestion (e.g., via explicit congestion notification or a pricing mechanism) or via an engineered solution based on some type of signaling. In particular, high-level composite interactions in a P2P system often rely on the coordination of many peers, so it is difficult to base solutions purely on local decisions. However, as measurement projects produce more results to characterize peers' behavior, we may be able to obtain good solutions through localized traffic engineering, admission control and scheduling algorithms.

The problem is related to providing incentives to participants. If we apply a market economy model, in which each peer is free to set its own prices for resources and a stable global equilibrium is reached based on supply and demand, will an "invisible hand" mechanism globally optimize resource supply and utilization? How do we ensure fairness? Lessons from economics and distributed algorithmic mechanism design will play an increasingly large part in the design of such systems.

Locality versus Anonymity

P2P networks with large sets of participants offer an opportunity for obfuscating the activities of individual nodes. Indeed, many early projects capitalized on this. Yet practical P2P systems struggle with the apparently inherent contradiction between offering anonymous sharing of resources and the localization of service offers. A number of factors are reducing the anonymity characteristic of P2P systems; at the very least, their immunity to traffic analysis is being lost as such techniques become more sophisticated.

Increasingly, anonymity-preserving features may be implemented as an overlay on top of P2P applications or middleware. Crowds [RR98] and "onion routing" already take this approach to the Web and e-mail, respectively. Of course, the extent of the trade-off with performance lies in different users' requirements, and the degree to which particular applications need to exploit locality to obtain this.

From Overlay to Infrastructure

Many successful overlay systems migrate over time into the infrastructure itself, often to maximize the efficiency of the protocols but also to handle application-level security concerns. Indeed, the US National Academy of Science [C-TR02] recommended looking at overlay systems as a general approach to building research infrastructures. As the nature of P2P networks and infrastructures become well understood, the techniques may migrate into the infrastructure, just as the handling logic for IP traffic has migrated from overlay networks into native services such as routers.

Many P2P systems are complex, however, and devising a minimal service enhancement in the lower levels that would support their algorithms is an interesting challenge. The IETF FORCES working group has been working on remote IP router control (separating out packet forwarding and packet routing). Yet more is needed if general P2P intermediary functions are to be performed within time insignificant relative to packet transmission time. Furthermore, complex filtering and processing of content keys will be needed for maintaining a global-scale distributed hash table at the infrastructure level—not to mention the many hashes and signatures used in many current schemes. Such hopes show that we do not really understand what P2P actually means.

P2P Systems and Ad Hoc Wireless Network Duality

A defining characteristic of P2P systems is their ability to provide efficient, reliable, and resilient routing between their constituent nodes by forming structured ad hoc topologies. In this respect, we can draw useful parallels with ad hoc wireless networking.

Resilient P2P mechanisms for content distribution services have been proposed, but the effect of these systems on global network utilization is not well understood. Studies show that high topology maintenance and message routing overheads prohibit the use of such systems on wireless ad hoc networks, which suffer stringent power requirements and highly transient network connectivity.

An application of P2P techniques to mobile wireless ad hoc networks would involve making peers "load aware," by distributing load information and implementing a distributed congestion control scheme. To handle mobility, a topographical metric might be used for nodes to specify their own location. An *anycast* routing scheme, allowing a request to specify a set of satisfactory destinations, would be one approach to reducing message-passing overhead.

Next-generation P2P systems will probably use actual location and scope information to influence routing functions so that content is initially placed and requests are routed to copies that have proximity on a number of quality of service axes—often including delay, throughput, and packet loss, but perhaps also battery considerations for wireless users. Thus, the distribution of replicas in a content delivery or storage infrastructure will evolve to meet the user demand distribution, optimizing use of the scarce wireless resources to better match user concerns.

5.2.2. Conclusion

P2P computing has had a dramatic effect on mainstream computing, even blurring the distinctions between computer science, engineering, and politics. An unfortunate side effect is that due consideration often has not been given to the classic research in distributed systems. We hope that this section has shed light on the nature of P2P computing, the reasons for its successes (and failures) to date and the interesting relationships that exist between P2P and traditional distributed computing styles.

We are entering the age of massively distributed, global-scale computing and storage systems in which computing will change from a *commodity* to a *utility*. We hope that this discussion can stimulate both the P2P and traditional communities to identify how ideas from P2P and classical distributed computing can best be synthesized to influence and ultimately realize this vision of utility computing.

5.3. Analysis of existing middleware

In this section we analyze the major middleware architectures that have been until now proposed for mobile and/or distributed systems. Two of these systems, namely Lime and Xmiddle, are data centric middleware proposed for network composed by mobile nodes. These two systems make no assumptions about the existence of infrastructures. The third platform is an open-source project managed by Sun Microsystems Inc. called JXTA. This project is intended as a standardisation of protocols useful to build distributed peer-to-peer applications and services. It does not focus on new peer-to-peer polices, but only on the mechanisms to build them. This makes the JXTA project closer to the industry world (interest on standardization) than to the research world (focus on new polices). The last three reported proposals, namely CAN, Chord and Pastry, are platforms addressing distributed content organization. They use Distributed Hash Table (DHT) policies to organize content (whatever it is) over a distributed system. All of them have been designed and developed to work on networks with infrastructure.

Hereafter, we discuss the pros and cons offered by these systems to MobileMAN. For completeness, in Appendixes of this deliverable we include an overview of these middleware architectures.

Lime. Lime provides application developers with a powerful programming abstraction, represented by the transient tuple-space. This is appealing for ad hoc environments, but the emphasis of the work is on the semantic of the transient tuple-space and the primitives working on it, and not on the polices used to realize it efficiently on an ad hoc network. To give an example, suppose to have a Lime agent that performs an out (write) operation of a tuple on the transient tuple-space. The current Lime semantic leaves the tuple on the Agent local memory, unless a valid location (host address) is specified to let the tuple move deterministically. In both cases, if the destination agent quits the "game" the tuple will stop to exist in the shared space. A more persistent behaviour would lead the system closer to the (very desirable) concept of shared memory: outplaced tuples last on the system independently from the agents' lifetime.

Xmiddle. Xmiddle proposes another approach to data sharing. The system realizes the functionalities of a distributed version controller system and for this reason it supports a limited range of applications. It is designed for network without infrastructure but does not cope with multi-hop paths between peers sharing data. We see no advantages in adopting this system as a starting research point.

JXTA. JXTA is an industry-related project, where standardization of the mechanisms to be used in peer-to-peer service development is the main aim. The result is that the project does not focus in building new policies. The JXTA architecture is complete, but its size makes it hard to handle. Our suggestion would be not to use it for research purposes, but refer to JXTA to implement the policies developed during this project, in order to have them standardized in an open widely accepted community. This final step would be of great help to introduce the MobileMAN research results in the industry world.

Platforms for distributed content organization. The three analyzed systems, all provide a general distributed lookup mechanism. This service is of primary importance for P2P systems: it solves the problem of efficiently locating data items. Besides, the three systems report comparable lookup performances from the simulations, as they all perform lookups with logarithmic costs. However, they differ in the overlay schemes built to support distributed lookups:

> *CAN* realizes a *d*-dimensional space abstraction, where the participating nodes control non-overlapping zones of the space. Data items, as well as data queries, are mapped on space

points using a hash function, so that they fall under the control of one of the nodes. Zones are randomly chosen from the participating nodes. Routing is easy, as each node maintains information only about its neighbours in the virtual space, and forward CAN packets choosing the neighbour that owns a zones closer to the packet destination.

Chord works with a ring of addresses: in practice it can be seen as a mono-dimensional space where node addresses, data items and data description are mapped. In the Chord ring, each node is responsible for a range of consecutive keys: in a ring with three nodes *k*, *h*, *t*, with $k < h < t$, the nodes will own respectively the key ranges *[k, h)*, *[h, t)* and *[t, k)*, considering arithmetic modulo the greatest possible identifier. In Chord nodes do not choose which range of keys to own. The key ranges are specified by the node identifiers. Routing in Chord requires each node maintaining information about a logarithmic number of nodes, called *fingers*, chosen to span the whole ring. Packets are routed selecting at each step the finger that is supposed to own a range including the packet's key.

Pastry builds a ring similar to Chord, a mono-dimensional space modulo 2**128 – 1 where both node addresses and keys identifying data and data query are mapped. The address space is not divided in ranges as in Chord: a key *k* is stored on the node with the address closer to *k* (it could be smaller or greater of *k*). Routing works with address prefixes: at each step a packet is forwarded to a node which has a longer prefix in common with the packet key respect to the present node.

Among these distributed hashing mechanisms, Pastry offers the simpler mechanism and the clearest understanding of how messages are exchanged and routing tables are built. Besides, it is open source and comes with a series of services already built on top of it:

- SCRIBE, a group communication/event notification system
- PAST, an archival storage
- SQUIRREL, a co-operative web cache
- SplitStream, a high-bandwidth content distribution platform
- POST, a co-operative messaging system
- Scrivener, a system for resources fair sharing

In the framework of the MobileMAN project we are going to explore in details the issues related to porting Pastry over the proposed architecture, stressing the exploitation of the cross-layered architecture (strong interaction with the network status). In particular, we could see if it is possible to heavily simplify the Pastry overlay creation and maintenance (routing tables) by exploiting the Network Status information sharing capabilities. On top of the ported system, other fundamental services could be realized like distributed service discovery, distributed instant messaging, a more persistent Lime-like tuple-space concept and so on. The resulting architecture could be finally proposed to the JXTA project for standardization, in order to meet industrial targets.

5.4. References

[A97] ANDERSON, R. J. The Eternity Service, June 1997. http://www.cl.cam.ac.uk/users/rja14/eternity/ eternity.html.

[ABKM01] ANDERSON, D., BALAKRISHNAN, H., KAASHOEK, F., AND MORRIS, R. The Case for Resilient Overlay Networks. Proc. *of the 8th Annual Workshop on Hot Topics in Operating Systems (HotOS-VIII*), May 2001.

[ADG02] E. Anceaume, A.K. Datta, M. Gradinariu, G. Simon "Publish/subscribe scheme for mobile networks", Proc. ACM Workshop On Principles Of Mobile Computing 2002, pp. 74 – 81

[ADM01] Aberer, K., and Despotovic, Z. Managing trust in a P2P information system. In ACM Conference on Information and Knowledge Management, 2001.

[ADS02] ASPNES, J., DIAMADI, Z., AND SHAH, G. Fault-tolerant routing in P2P systems. In *Twenty-First ACM Symposium on Principles of Distributed Computing* (Monterey, USA, July 2002), pp. 223-232.

[ADS03] Acquisti, A., Dingledine, R., and Syverson, P. On the Economics of Anonymity. http://freehaven.net/doc/fc03/econymics.pdf

[AH01] ADAR, E., AND HUBERMAN, B. Free riding on Gnutella. Tech. rep., 2001.

[AX02] ANDRZEJAK, A., AND XU, Z. Scalable, efficient range queries for grid information services. Tech. Rep. HPL-2002-215, Hewlett-Packard Laboratories, Palo Alto, 2002.

[B70] BLOOM, B. H. Space/time tradeoffs in hash coding with allowable errors. *Comm. of the ACM 13*, 7 (July 1970), 422.

[BHHLM01] Beardsmore, A., Hartley, K., Hawkins, S., Laws, S., Magowan, J., Twigg, A. GSAX Grid Service Accounting Extensions http://www.gridforum.org/meetings/ggf6/ggf6_wg_papers/ggf-rus-gsax-01.doc

[B-TR] BELLOVIN, S. Security aspects of Napster and Gnutella. Tech. rep.

[CCR02] Special issue "Special feature on middleware for mobile & pervasive", *ACM Mobile Computing and Communications Review*, Volume 6 Issue 4 (October 2002).

[CDFIK02] Chervenak, A., Deelman, E., Foster, I., Iamnitchi, A., Kesselman, C., Hoschek, W., Kunst, P., Ripeanu, M., Schwartzkopf, B., Stockinger H., Stockinger, K. and Tierney, B., Giggle: A Framework for Constructing Scalable Replica Location Services. In Proc. of SuperComputing 2002, Baltimore, Maryland, November 11-16, 2002.

[CDKNRS03] Castro, M., Druschel, P., Kermarrec, A.-M., Nandi, A., Rowstron A., and Singh, A. SplitStream: High-bandwidth content distribution in a cooperative environment. In *Proc. IPTPS'03*, Berkeley, CA, February, 2003.

[CFFK01] Czajkowski, K., Fitzgerald, S., Foster, I., Kesselman, C**.,** Grid Information Services for Distributed Resource Sharing. Proceedings of the Tenth IEEE International Symposium on High-Performance Distributed Computing (HPDC-10)*,* IEEE Press, August 2001.

[CL99] Castro, M., and Liskov, B. Practical Byzantine Fault Tolerance. In Proc. OSDI 1999.

[CN02] CUENCA-ACUNA, F. M., AND NGUYEN, T. D. Text-based content search and retrieval in ad hoc p2p communities, 2002.

[CRSZ01] CHU, Y., RAO, S., SESHAN, S., AND ZHANG, H. Enabling conferencing applications on the internet using an overlay multicast architecture. Tech. rep., 2001.

[CRZ00] CHU, Y., RAO, S., AND ZHANG, H. A case for end system multicast. Tech. rep., 2000.

[CSWH01] CLARKE, I., SANDBERG, O., WILEY, B., AND HONG, T. W. Freenet: A distributed anonymous information storage and retrieval system. L*ecture Notes in Computer Science 2009* (2001).

[C-TR02] COMMITTEE ON RESEARCH HORIZONS IN NETWORKING. Looking over the fence at networks: A neighbour's view of networking research. Tech. rep.

[D02] Douceur, J. The Sybil Attack. In *Proc. of the First International Workshop on P2P Systems (IPTPS '02)*, Cambridge, MA, March 2002.

[DBF-TR] DESHPANDE, H., BAWA, M., AND GARCIA-MOLINA, H. Streaming live media over a P2P network. Tech. rep.

[DCMI] Dublin Core Metadata Initiative http://www.dublincore.org/

[DFM00] Dingledine, R., Freedman, M., Molnar, D. The Free Haven Project: Distributed Anonymous Storage Service. In *Proc. of the Workshop on Design Issues in Anonymity and Unobservability,* 2000.

[DKKMS01] DABEK, F., KAASHOEK, M. F., KARGER, D., MORRIS, R., AND STOICA, I. Wide-area cooperative storage with CFS. In *Symposium on Operating Systems Principles* (2001), pp. 202-215.

[FI03] Foster, I., Iamnitchi, A., On Death, Taxes, and the Convergence of Peer-to-Peer and Grid Computing. In *Proc. IPTPS'03*, Berkeley, CA, February, 2003.

[FIPS180] FIPS 180-1 Secure hash standard. Technical Report Publication 180-1, Federal Information Processing Standards, NIST, US Dept. of -Commerce, April 1995.

[FJJJRSZ01] FRANCIS, P., JAMIN, S., JIN, C., JIN, Y., RAZ, D., SHAVITT, Y., AND ZHANG, L. Idmaps: a global internet host distance estimation service. *IEEE/ACM Transactions on Networking (TON) 9*, 5 (2001), 525-540.

[FK02] Foster, I., and Kesselman, C. The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. Technical Report, Globus Project, 2002.

[FKG99] Foster, I., and Kesselman, C. Globus: A Toolkit-Based Grid Architecture. In *The Grid: Blueprint for a new Computing Infrastructure*, Foster, I and Kesselman, C., eds. 1999.

[FS02] Feigenbaum, J., Shenker, S., Distributed Algorithmic Mechanism Design: Recent Results and Future Directions, in *Proceedings of the 6th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, ACM Press, New York, 2002.

[FV02] FREEDMAN, M. J., AND VINGRALEK, R. Efficient P2P lookup based on a distributed trie.
In *Proceedings of the 1st International Workshop on P2P Systems (IPTPS02)* Cambridge, MA, March 2002).

[G99] R. GOLDING, E. B. Fault tolerant replication management in large scale distributed storage systems. In *Proceedings of Symposium on Reliable Distributed Systems* (1999).

[GHITS] GRIBBLE, S., HALEVY, A., IVES, Z., RODRIG, M., AND SUCIU, D. What can P2P do for databases, and vice versa?

[GK02] Geels, D., and Kubiatowicz, J. Replica Management Should Be A Game. In Proc. *SIGOPS European Workshop* 2002.

[GSGK02] GUMMADI, K. P., SAROIU, S., AND GRIBBLE, S. D. King: Estimating latency between arbitrary internet end hosts. In *SIGCOMM Internet Mesurement Workshop 2002, Marseille, France* (November 2002).

[H01] Hong, T. Performance. In *P2P: Harnessing the Power of Disruptive Technologies*, ed. A. Oram. O'Reilly and Associates, 2001.

[H03] Klaus Hermann, "MESHMdl - A Middleware for Self-Organization in Ad hoc Networks", Proc. IEEE Workshop on Mobile and Distributed Computing (MDC 2003) in conjunction with ICDCS 2003, 19 May 2003.

[HDV01] Sumi Helal, Nitin Desai and Varum Verma, "Konark - A Service Discovery and Delivery Protocol for Ad-hoc Networks", Proceedings of the Third IEEE Conference on Wireless Communication Networks (WCNC), March 2003.

[HJSTW03] Harvey, N., Jones, M. B., Saroiu, S., Theimer, M., and Wolman, A. SkipNet: A Scalable Overlay Network with Practical Locality Properties. In Proc *of Fourth USENIX Symposium on Internet Technologies and Systems (USITS '03)*, March 2003.

[HKMNS88] Howard, J., Kazar, M., Menees, S., Nichols, D., Satyanarayanan, M., Sidebotham, R., and West, M. Scale and Performance in a Distributed File System. In *ACM Trans. on Computer Systems*, Feb. 1988.

[HLL99-a] HANNA, K. M., NATARAJAN, N., AND LEVINE, B. Evaluation of a novel two-step server selection metric. Tech. rep., 2001.

[HLL99-b] HARCHOL-BALTER, M., LEIGHTON, F. T., AND LEWIN, D. Resource discovery in distributed networks. In *Symposium on Principles of Distributed Computing* (1999), pp. 229-237.

[JGJKO00] JANNOTTI, J., GIFFORD, D. K., JOHNSON, K. L., KAASHOEK, M. F., AND J. W. O'TOOLE, J. Overcast: Reliable multicasting with an overlay network. Tech. rep., 2000.

[JXTA] http://www.jxta.org.

[KAZAA] Kazaa Media Desktop.http://www.kazaa.com/

[KBCE00] KUBIATOWICZ, J., BINDEL, D., CHEN, Y., EATON, P., GEELS, D., GUMMADI, R., RHEA, S., WEATHERSPOON, H., WEIMER, W., WELLS, C., AND ZHAO, B. Oceanstore: An architecture for global-scale persistent storage. In *Proceedings of ACM ASPLOS* (November 2000), ACM.

[KF] Kavitha Ranganathan, Ian Foster, paper in Journal of Grid Computing.

[KLLLLP97] Karger, D., Lehman, E., Leighton, F., Levine, M., Lewin, D., and Panigrahy R. Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the World Wide Web. In Proc. ACM SOTC, May 1997.

[LB03] Lee, S., and Bhattacharjee, B. Cooperative Peer Groups in NICE.In Proc. *IEEE Infocom,* 2003.

[LCCLS02] Lv, Q., Cao, P., Cohen, E., Li, K., and Shenker, S. Search and replacement in unstructured P2P networks. In Proc. of the 16th ACM International Conference on Supercomputing (ICS), 2002.

[LLMC88] Litzkow, M., Livny, M. and Mutka, M. Condor – A Hunter of Idle Workstations.In Proc 8th Intl Conf. on Distributed Computer Systems, 1988.

[LRW03] Leibowitz, N., Ripeanu, M., and Wierzbicki, A., *Deconstructing the Kazaa Network*. In proc. of Third IEEE Workshop on Internet Applications (WIAPP'03), June 2003, San Jose, CA.

[LSZ03] J. Liu, K. Sohraby, Q. Zhang, B. Li, W. Zhu, "Resource Discovery in Mobile Ad Hoc Netwoks", Chapter 26 in *The Handbook of Ad Hoc Wireless Networks*, M. Ilyas (Editor), CRC Press, 2003.

[MC02] Renè Meier, Vinny Cahill, "STEAM: Event-Based Middleware for Wireless Ad Hoc Networks", Proceedings of the 22nd International Conference on Distributed Computing Systems Workshops (ICDCSW'02).

[MCE02] Cecilia Mascolo, Licia Capra, Wolfgang Emmerich, "Middleware for Mobile Computing (A Survey)" in Advanced Lectures on Networking, Enrico Gregori, Giuseppe Anastasi, Stefano Basagni (Editors) LNCS 2497, 2002.

[MCE02a] C. Mascolo, L. Capra, S. Zachariadis, W. Emmerich, "XMIDDLE: A Data-Sharing Middleware for Mobile Computing," Wireless Personal Communications, vol. 21, pp. 77–103, 2002.

[MMGC02] Ivy: A Read/Write P2P File System. Muthitacharoen, A., Morris, R., Gil, T., and Chen, B. In *Proc. of the 5th USENIX Symposium on Operating Systems Design and Implementation (OSDI '02)*, 2002.

[MMK02] MAYMOUNKOV, P., AND MAZIÈRES, D. Kademlia: A P2P information system based on the xor metric. In *1st International Workshop on P2P Systems (IPTPS '02)* (March 2002), MIT Faculty Club, Cambridge, MA, USA.

[MPH02] MORETON, T. D., PRATT, I. A., AND HARRIS, T. L. Storage, Mutability and Naming in *Pasta*. In *Proceedings of the International Workshop on P2P Computing at Networking 2002, Pisa, Italy.* (May 2002).

[MPR01] A. L. Murphy, G. P. Picco, G.-C. Roman, "Lime: A middleware for physical and logical mobility," in Proceedings of the 21st International Conference on Distributed Computing Systems (ICDCS-21), Phoenix, AZ, USA, pp. 524–233, April 16-19 2001

[MSN84] MICHAEL D. SCHROEDER, A. D. B., AND NEEDHAM, R. M. Experience with grapevine: The growth of a distributed system. *ACM Transactions on Computer Systems, vol. 2, no. 1, pp. 3-23* (Feb. 1984.).

[MT02] Moreton, T., and Twigg, A. Enforcing Collaboration in P2P Routing Services, In *Proc. First International Conference on Trust Management*, May 2003.

[MTX02] MAHALINGAM, M., TANG, C., AND XU, Z. Towards a semantic, deep archival file system.
Tech. rep., Hewlett-Packard Research Labs, July 2002.

[N] NAPSTER. Napster media sharing system. http://www.napster.com/.

[NZ02] NG, E., AND ZHANG, H. Predicting internet network distance with coordiantes-based approaches. In *INFOCOM'02, New York, USA* (2002).

[PCW02] PIAS, M., CROWCROFT, J., AND WILBUR, S. Lighthouse: A QoS metric space to maintain network proximity. UNPUBLISHED, October 2002.

[PRR97] PLAXTON, C. G., RAJARAMAN, R., AND RICHA, A. W. Accessing nearby copies of replicated objects in a distributed environment. In *ACM Symposium on Parallel Algorithms and Architectures* (1997), pp. 311-320.

[PS01] PADMANABHAN, V. N., AND SUBRAMANIAN, L. An investigation of geographic mapping techniques for internet hosts. *Proceedings of SIGCOMM'2001* (2001), 13.

[RD01] A. Rowstron and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems". IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), Heidelberg, Germany, pages 329-350, November, 2001.

[RD01a] ROWSTRON, A., AND DRUSCHEL, P. Pastry: Scalable, decentralized object location, and routing for large-scale P2P systems. *Lecture Notes in Computer Science 2218* (2001), 329-350.

[RD01b] ROWSTRON, A. I. T., AND DRUSCHEL, P. Storage management and caching in PAST, a large-scale, persistent P2P storage utility. In *Symposium on Operating Systems Principles* (2001), pp. 188-201.

[REGWZ03] Rhea, S., Eaton, P., Geels, D., Weatherspoon, H., Zhao, B., and Kubiatowicz, J. Pond: the OceanStore Prototype. In *Proc. the 2nd USENIX Conference on File and Storage Technologies (FAST '03),* March 2003.

[RFHKS00] RATNASAMY, S., FRANCIS, P., HANDLEY, M., KARP, R., AND SHENKER, S. A scalable content addressable network. Tech. Rep. TR-00-010, Berkeley, CA, 2000.

[RFHKS01] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. "A scalable content-addressable network". In Proc. ACM SIGCOMM 2001, August 2001.

[RFI00] RIPEANU, M. FOSTER, I., IAMNITCHI, A.
Mapping the Gnutella Network: Properties of Large-Scale Peer-to-Peer Systems and Implications for System Design In *IEEE Internet Computing* special issue on *Peer-to-Peer Networking, vol. 6(1), 50-57, February 200.*
http://people.cs.uchicago.edu/~matei/PAPERS/ic.pdf

[RH02] ROSCOE, T., AND HAND, S. Transaction-based Charging in Mnemosyne: a P2P Steganographic Storage System. In *Proceedings of the International Workshop on P2P Computing at Networking 2002, Pisa, Italy.* (May 2002).

[RPMES98] REED, D., PRATT, I., MENAGE, P., EARLY, S., AND STRATFORD, N. Xenoservers: Accountable execution of untrusted programs.
http://www.cl.cam.ac.uk/Research/SRG/netos/ xeno/hotos1/index.html, November 1998.

[RR98] Reiter, M., and Rubin, A. Crowds: anonymity for Web transactions. In *Proc. ACM Transactions on Information and System Security,* 1998.

[RW] Security for structured P2P overlay networks. Castro, M., Druschel, P., Ganesh, A., Rowstron, A., and Wallach, D. Submitted for publication.

[SETI] Seti@Home Project http://setiathome.ssl.berkeley.edu/

[SGKWL85] Sandberg, R., Goldberg, D., Kleiman, S., Walsh, D., and Lyon, B. Design and Implementation of the Sun Network Filesystem. In *Proc. Summer USENIX*, June 1985.

[SH03] Spence, D., and Harris, T. XenoSearch: Distributed Resource Discovery in the Xenoserver Open Platform. In *Proc. 12th International Symposium on High Performance Distributed Computing*, 2003.

[SMKKB01] I. Stoica, R. Morris, D. Karger, F. Kaashoek, H. Balakrishnan. "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications". In *Proceedings ACM Sigcomm 2001*, San Diego, CA, Aug. 2001.

[SMKKB01] STOICA, I., MORRIS, R., KARGER, D., KAASHOEK, F., AND BALAKRISHNAN, H. Chord: A scalable P2P lookup service for internet applications. In *Proceedings of the ACM SIGCOMM 2001 Conference (SIGCOMM-01)* (New York, August 2001), R. Guerin, Ed., vol. 31, 4 of *Computer Communication Review*, ACM Press, pp. 149-160.

[SRU] Singla, A., Rohrs, C., Ultrapeers: Another Step Towards Gnutella Scalability. http://www.limewire.com/developer/Ultrapeers.html

[ST-TR] SHI, S., AND TURNER, J. Routing in overlay multicast networks. Tech. rep.

[SUN02] SUN. Jxta peer-peer system, April 2002. http://www.jxta.org/.

[TBSL01] Thain, D., Basney, J., Son, S.-C. and Livny, M., The Kangaroo Approach to Data Movement on the Grid. *Proceedings of the Tenth IEEE Symposium on High Performance Distributed Computing*, 2001, 7-9.

[TH01] Todd D. Hodes and others, "Composable ad-hoc Mobile Services for Universal Interaction", Book Mobile Computing and Networking, Pages 1-12, 1997 (citeseer.nj.nec.com/hodes97composable.html).

[TTPDSH95] Terry, D., Theimer, M., Petersen, K., Demers, A., Spreitzer, M., and Hauser, C. Managing Update Conflicts in Bayou, a Weakly Connected Replicated Storage System. In Proc. of 15th Symposium on Operating Systems Principles (SOSP-15) , Cooper Mountain, Colorado, 1995

[TXM02] TANG, C., XU, Z., AND MAHALINGAM, M. pSearch: Information Retrieval in Structured Overlays. In *First Workshop on Hot Topics in Networking* (October 2002).

[UD] United Devices. http://www.ud.com/

[W01] M.F. Worboys, "Nearness relations in environmental space", International Journal of Geographical Information Systems, 2001.

[W02] B. Wilcox-O'Hearn. Experiences deploying a large-scale emergent network. In *Pro.c of the First International Workshop on P2P Systems (IPTPS '02)*, Cambridge, MA, March 2002.

[WCB01] Welsh, M., Culler, D., and Brewer, E. SEDA: An Architecture for Well-Conditioned, Scalable Internet Services. In *Proc. 18th Symposium on Operating Systems Principles (SOSP-18)*, Banff, Canada, 2001.

[WN01] Webnoize Web site http://www.webonize.com/ February 2001.

[YG02] Yang, B., and Garcia-Molina, H. Efficient Search in P2P networks. In Proc. of the 22nd IEEE International Conference on Distributed Computing Systems (ICDCS), 2002.

[ZAFB00] ZEGURA, E., AMMAR, M., FEI, Z., AND BHATTACHARJEE, S. Application-level anycasting: a server selection architecture and use in a replicated web service. Tech. rep., 2000.

[ZDHJK02] Brocade: Landmark Routing on Overlay Networks. Zhao, B., Duan Y., Huang, L., Joseph, A., and Kubiatowicz J. In *Proc. First International Workshop on P2P Systems (IPTPS '02),* Cambridge, MA, March 2002.

[ZKJ01] ZHAO, B. Y., KUBIATOWICZ, J., AND JOSEPH, A. D. Tapestry: an infrastructure for fault-resilient wide-area location and routing. Tech. Rep. UCB//CSD-01-1141, University of California at Berkeley, April 2001.

[ZPS-TR00] ZHANG, Y., PAXSON, V., AND SHENKER, S. The stationarity of internet path properties: Routing, loss, and throughput. *ACIRI Technical Report* (2000).

6. APPLICATIONS

6.1. Status of the Art

In principle each application could be provided on an ad hoc environment. But we must consider some typical aspects of ad hoc networks, that make some kind of applications less or more attractive. We can summarize some of these points as:

i. dynamic topologies,
ii. absence of pre-existing infrastructure,
iii. cooperation among nodes is mandatory.

As a consequence of i-iii above:

- Connections are unreliable as frequent nodes' movements cause frequent set-up and tear-down of existing connections.
- The absence of existing infrastructure implies that well-known traditional servers cannot be easily available.
- Nodes could be, at the same time, active nodes (provide services like forwarding) and passive (access services as users).
- Cooperation among nodes is important to guarantee the right behaviour (nodes should be active to provide forwarding packet features to other nodes).
- Service Discovery is difficult. While in the traditional environment with infrastructure services are easy to be discovered and addressed, in an ad-hoc environment this becomes an important and difficult problem. In general, a service cannot be addressable via a well known address (e.g. IP address or Name) but a way to know how to address a service is needed.

It follows that applications running in a mobile ad hoc network face a new set of challenges. They must adapt to the connectivity changes as mobile or dedicated nodes move relatively to each other. Moreover, the applications must limit use of bandwidth for management so that enough bandwidth remains available to end-users.

6.2. Scenarios

Traditional there are several target fields where this technology has good chance to success. Areas where the infrastructure cannot be set up, for instance because of cost or unavailability of necessary equipment; examples are Disaster Recovery, and Building Site. Other scenarios could be represented by a campus where could be attractive to be able to set up network at low cost without any infrastructure. The needing is different in both cases but always interesting. In the first case an ad hoc solution could be the only one applicable or easy to be implemented, while in the second case is the nature of the scenario (crowded area) that makes easy to set up a network because of potential large number of terminals with the same affiliation. Number (density) and nature of users are important aspects for ad hoc networks because they guarantee the connectivity.

In general we can identify two main application models:

- Client-Server
- Peer-to-Peer

The first case represents traditional applications like, for instance, Web Applications, and e-Mail. This requires the existence of fixed elements (servers) that should be easily addressable. Web servers are for instance well known via the URL. Web servers are always available if connected to a fixed network. In the case of ad hoc networks this constraint (server availability) becomes difficult to satisfy. Other solutions could replace this schema, like the concept of distributed server.

In this case, a server is not represented by a component (or set of components) but from a set of elements. A logical server becomes the union of several elements, and then the classical server (unique or main component) disappears, replaced by a set of components that change. In any case a central element (still in a distributed server) is needed to coordinate all other components.

The second model provides a flat architecture where all elements are at same layer. Applications include some examples of video conferencing or cooperative working applications (e.g. NetMeeting). This scenario fits better with the mobile ad hoc architecture because the nature of the network is flat and all elements can be considered at the same level. Co-operative working represents an important topic for ad hoc networks. Ad hoc networks are based on cooperation among nodes at the network level, while co-operative working provides/requires co-operation among users.

6.3. Content Sharing Application

We consider an example of a distributed collaboration tool to share contents in an ad hoc networks, Content sharing, that is the opportunity to access documents (or more in general contents) exported by other actors. Content sharing applications could be used in several target domains, for instance a large-scale construction site, to share document each other among engineers or architects. Other target domains are of course possible like a campus, where only the content type changes but the application is the same. Another interesting target could also be a hospital to provide interconnectivity among team members everywhere within the hospital. Nature of content is not important and it changes in the different target.

6.3.1. Technical Aspects

Each element of the network represents a client that owns some contents, which exports to other users and gets access to other user's contents.

Main problems to be approached are:

- *Service discovery*: how an actor can discover other actors. In MANET architecture the peer-to peer manner is a good solution for mobile computing system where it is necessary to discover resources and dynamically route information through the network. To obtain an efficient service discovery, we think to implement the concept of multi-hop routing protocol to discovery the service. This idea introduces a new level, under application level, independent from the supplied service.
- *Content location*: how an actor can manage owned and other actors' contents. In MANET a server-based solution is not appropriate for a number of reasons, but this functionality should be distributed across nodes in the MANET to minimize the dependency on any particular node. The nodes participating in ad hoc networks are often resource-constrained devices, such as PDAs or mobile phones. Because of this, it is advantageous for the processing and networking overheads to be distributed across the nodes rather than concentrated at a single server node. So the client-server paradigm is moved to a server-less model where the information needs to be maintained in a distributed manner. Other aspect is the "classification of the contents", the classification is important to search and to public the contents of a service.
- *Content retrieving*: how an actor retrieves other contents. Once the user has found the service, he must retrieve the contents. At this point is necessary to have a stable connection because the service can require a connection TCP for a long time. But the wireless connections are not stable and, to get out this, a middleware level must be introduced. This level makes to seem to the applications having a stable TCP connection.

All the aspects listed above are current area of research.

6.3.2. Social and Economical Aspects

A distributed co-operative application is based on the idea that people need features to cooperate. This kind of tool includes inside the main idea of ad hoc network: cooperation supports the network. In fact the network can increase providing large interconnectivity among users if all users co-operate; the content database increases if many users export their contents to other and stay alive for all the time.

Economical aspects can be also addressed if we move the target towards the market. If we consider that the basis of ad hoc network is the density of population we can also include in our target other reality like the beach or large area with many people and no infrastructure. In this case contents can assume a different interest because people want to sell owned contents to other people. In this case other problems arise to manage application:

- Application security
- P2P Billing

Both these items are important and each-other depending. Indeed, security is needed to provide secure contents (nobody can access them if not authorized) and to guarantee identification of provider and customer. The P2P model seems to be quite conformance to the C2C business model that is perhaps the youngest Business Model in the E-Commerce area.

6.4. References

[TG] C.Tuduce, T.Gros “Organizing a Distribute Application in a Mobile Ad Hoc Network” Second IEEE Internetional Symposium on Network Computing and Application

[WZ] J.Wu, M.Zitterbart “Service Awareness and its challenges in Mobile Ad Hoc Networks”

[CN02] F. Matias Cuenca-Acuna, T.D.Nguyen “Text Based Content Search and Retrivieal in ad hoc P2P Communities” P2P Computing, IFIP Networking 2002 Pisa

7. ECONOMIC ISSUES

7.1. Introduction of Approach: Promoting Cooperation

There are good reasons why nodes in a mobile ad hoc network, that lacks the networking infrastructure which has been deployed through the investment of a telecommunications corporation, would prefer not to cooperate. When nodes do cooperate, they form the necessary ad hoc infrastructure that makes multi-hop communication achievable, allowing traffic from a node to reach destinations that would either require a significant amount of transmission energy using single hop communication, or simply not be possible without routing the traffic through other nodes. However, this means that nodes must be willing to forward traffic for other nodes, and in this way expend energy without receiving any direct gain from doing so. If a node only considers its own short-term utility, then it may not choose to participate within the network.

Thus, the concept of introducing incentives for collaboration into the architecture of this type of network is an important step, and one which allows us to consider the dynamics of the cooperation and preferences of nodes within a system. This leads us naturally to the use of pricing mechanisms, which have found application in rate control in wireline networks and resource control in wireless networks. The difference in this situation is that nodes recover costs, associated with energy losses and traffic loading at a particular node, through the credit arising from pricing mechanisms. This has been shown to stimulate cooperation within ad hoc networks. Determining energy-efficient routes is also an important consideration in ad hoc networks, and pricing mechanisms provide the means of guiding a system to its optimal operating point.

In this research work, we have specifically considered the issue of how prices can be determined automatically by the ability of nodes to pay the costs for transmitting traffic, and the routes that are subsequently used. We have shown that cooperation is a natural outcome that emerges from incentives created by the pricing mechanisms. We have further studied the way that the mobility of the users affects the sharing of resources.

7.2. System Description

We model our network as a set of N mobile nodes that are equipped with directional, wireless antennas. Amongst the set N of nodes, there is a set S of sources that have destinations D to send traffic to. To do this, a set of routes between each source and destination pair has been determined. These routes can be determined using routing protocols like AODV or DSR. We observe that a node has limits on its capacity to transmit or receive, where the capacity limit is defined by its spectrum allocation and medium access protocol. Power consumption is also constrained at a node, due to the rate of discharge of the node's battery which leads us to defining a power constraint at each individual node.

Our approach to route selection and flow allocation closely follows the theoretical formulation given in [KMT98]. The formulation develops a mechanism to allow nodes to make decentralized decisions concerning the choice of the flows on potential routes. The nodes make these decisions based on congestion prices announced by relevant nodes. In this way, nodes with a given willingness-to-pay for congestion costs can adjust their resource usage accordingly. The approach in this paper builds on [KMT98] by the incorporation of power as well as bandwidth prices to reflect additional constraints that arise in wireless networks.

So far, our model produces a traffic allocation across possible routes determined by the willingness-to-pay parameters, $w_s(t)$, for each user. We now seek to provide an incentive for a user j, say, to act as a transit node for other users' traffic by supposing that user j receives a notional

credit for the congestion costs it incurs from each individual source with routes passing through *j*. The credit thus accumulated can then be re-cycled as payment to other nodes which act as transits for traffic originating at the resource. In this way, users will have the strongest incentive to act as transits where there is the greatest excess demand for traffic, since they earn the most in transit fees. Note that we consider a node to represent a composite resource, having both capacity and energy resources.

We suppose that each user maintains a credit balance, $b_s(t)$, which receives an initial endowment of 1 when the user arrives into the system, where we here identify a node with the user, *s* say, whose routes originate at that node. The user's credit balance is then adjusted by transferring credit equal to the congestion costs to each of the downstream resources. Each user will seek to control their credit balance, $b_s(t)$, and we envisage them doing so by dynamically adjusting their willingness-to-pay parameters, $w_s(t)$, according to the level of their credit balance by following a rule of the form: $w_s(t) = \alpha_s b_s(t)$ for some parameter $\alpha_s > 0$. In this way, the users' sending rates would become coupled with their credit balance and they would thereby naturally reduce their sending rate whenever their credit balance was low.

7.3. Simulations

The dynamics of the system described in Section 7.2 are illustrated here using a simulation model. In particular, we demonstrate the stability of prices at nodes and their credit balances. With regards to performance, we also investigate the throughput of the system. Certain dynamics of the system are also studied, including the arrival and departure of users from the system and how this affects the total credit. Finally, we consider how user mobility affects their individual throughput and also how it contributes to the overall system throughput.

Initially, we studied a given network of ten users located randomly according to a uniform distribution within a geographical area of *100m* by *100m*, as shown in Figure 7.1.

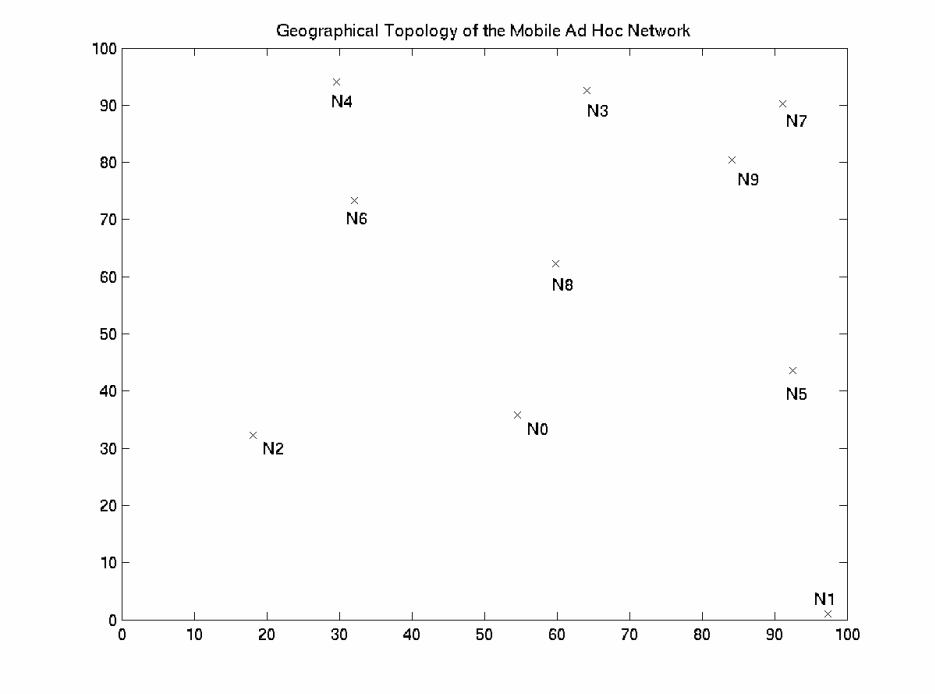


Figure 7.1: Topology of the mobile ad hoc network

It has the features of higher clustering of nodes towards the top-right corner, a node closely situated within the geographical centre of the network (node N8), and nodes with higher geographical isolation (particularly nodes N1 and N2). This allows us to study a set of nodes with diverse geographical locations and topological relationships.

Each node is equipped with a single transceiver with range 56 meters, which defines the neighbors that it has within the network. Notice that for nodes in the centre of the network, this means that nodes have a large set of neighbors and hence have a higher number of routes to choose from, in order to send traffic to a particular destination. Nodes, such as N1 and N2, have only a few neighbors, and so can only select routes from a smaller set of possible paths.

With regards to traffic model, we assume that a particular user establishes a connection with a randomly selected recipient, where the connection duration is exponentially distributed. Once the connection with a particular destination node terminates, the user remains idle for an exponentially distributed period before randomly selecting another node to initiate a connection with. Consequently, a particular user will communicate with a number of different users during the course of a simulation, which we believe is a realistic approach for modeling the traffic behavior within the network.

When a user initiates a connection with another user, it determines the lowest cost route and then continues to use that particular route for the duration of the connection. Notice that this is a departure from the model described in Section 7.2, where users continually monitor all available routes to the recipient user and always route traffic through the route with minimum cost. However, this departure from the model is a realistic one, because we want to minimise the amount of routing information that has to be distributed within the network. When using one of the proposed ad hoc routing protocols, such as AODV [PBD03] or DSR [JMHJ02], it is reasonable to assume that the integrity of routes will need to be checked before routing a stream of packets along a particular path. However, it is unlikely that nodes will continuously monitor all paths at the granularity level of transmitting each packet. The consequence of this departure from the model is that the system will not achieve optimal performance, but there is a trade-off between optimality and the overhead involved in continuously monitoring the prices of other routes to the destination. Another advantage of this approach is that route-flapping is avoided, which may occur if the price of another route drops below the route currently used, and then a user begins to frequently swap traffic between these two routes.

7.3.1 Static Network: Stability of Price and Credit Balances

To demonstrate the stability of the system, we simulate a static network topology for 10,000s where the mean duration of a connection is 0.5s, and a user is idle for a mean period of 0.5s after completing a connection. The users update their prices every 0.01s. The system parameters used in the system are set as $\alpha = 0.3$, $\beta = 0.01$ and $\kappa = 0.05$. The bandwidth capacity is set to $C = 10$ for all nodes in the network, while the maximum power is $\Gamma = 0.5$.

The prices and credit balance of four representative nodes in the network are shown in Figure 7.2. Node N1 has been selected, as it is the most extreme node in the network, while node N7 is an extreme node with nodes N3 and N9 in close proximity. The prices of the node nearest the centre of the network, namely N8, have also been plotted, together with those of node N9, which is also frequently used as a transit node.

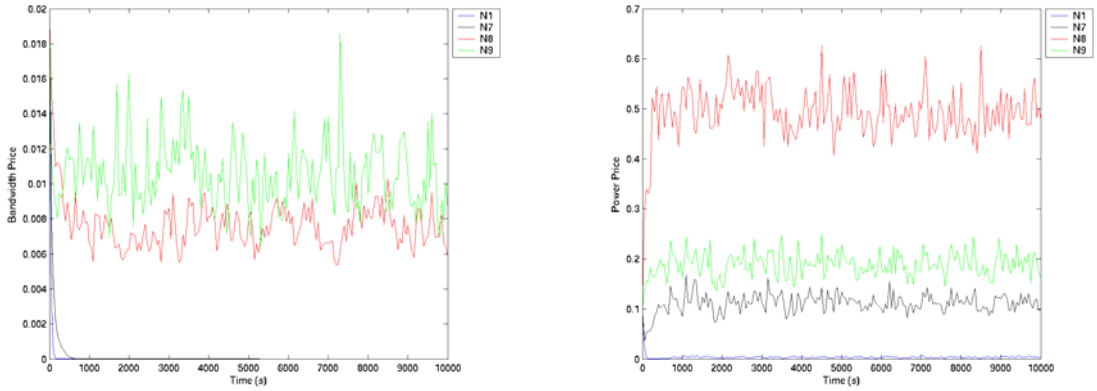


Figure 7.2: Bandwidth and power prices curves of four representative nodes.

It can be observed from these plots that each price stabilizes about a mean value, hence providing evidence that the overall system is stable. It should be noted that this occurs with the sub-optimal routing policy that minimum cost routes are only selected when connections are established. A second observation is that prices for node N1, which is on the edge of the network, all decay rapidly to zero. This is because no routes are selected which use N1 as a transit node, and the only flows which consume bandwidth or power resources at this node are those originating or terminating at N1.

It is also interesting to compare the prices of nodes N8 and N9. The bandwidth price is the highest for N9, while N8 has the highest power price. The reason for this is that while node N8 is the closest to the centre of the network, distances to its neighboring nodes are all relatively high. Hence, as more power is consumed by N8 in transmitting to other nodes, the power price will be driven up. In comparison, N9 is not near the centre of the network, yet is close to nodes N3 and N7, and will be carrying larger amounts of traffic for these nodes and for other nodes that route traffic around this cluster of nodes. Hence, its capacity usage will be reasonably high, as reflected by its bandwidth price.

The credit balances and throughputs, for the same nodes, are plotted in Figure 7.3. Throughput is determined by logging the accumulative traffic originating from the node in 50s intervals. Once again, note that these quantities stabilize around their individual mean value. The mean value for each node's credit balance is largely dependent on their geographical location within the network. As would be expected, node N8 maintains the highest credit balance, as it will be carrying a large amount of transit traffic. In addition, N8 will be charging high power prices for doing so, and thus accruing significant credit in the process. As the location of a node gets closer to the edge of the network, its credit balance is seen to decrease.

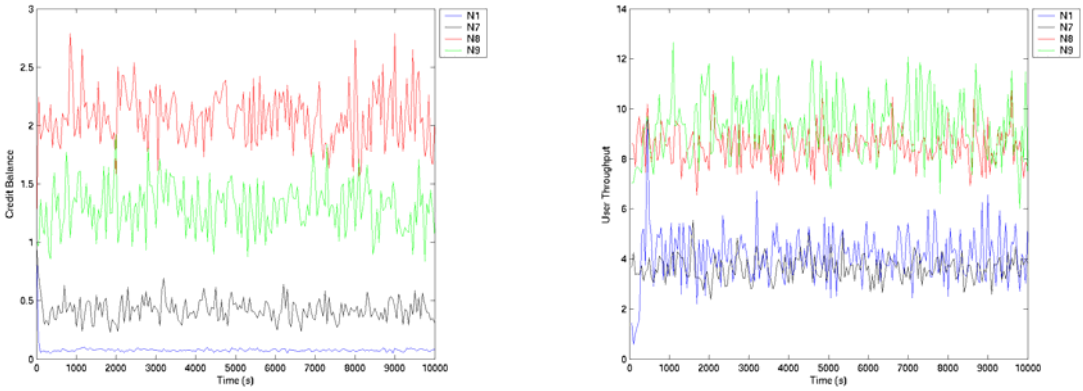


Figure 7.3: Credit balances and throughput curves of four representative nodes.

7.3.2 Dynamic Network: Arrivals and Departures of Users

Having demonstrated that the system does stabilize for a static network using simulations, we now investigate a dynamic network where users can join and leave the network, depending on decisions based on conditions that are external to the network. Consequently, we model the arrivals and departures as a random process. In particular, we consider the arrivals as a Poisson process and the "lifetime" of a user in the network is exponentially distributed. The location of user, when it arrives, is randomly distributed within the *100m* by *100m* square area in which the network is situated.

We are concerned with the consequence to the total credit in the system when users join or leave the network. A user who arrives will always increase the total credit by one, due to its initial credit endowment. However, the situation is not the same when a user leaves the system, as the user may have accumulated a large amount of credit from other users, because of its ability to act as a transit node. When such a node leaves the network, the total credit will decrease by more than one. Otherwise, if a user has spent most of its credit paying transit fees for traffic routed through other nodes, then the total credit will not decrease significantly. In both cases, at the instant when the user departs, the total credit will not reflect the true number of users in the system.

The parameter β has been introduced into the update algorithm for the credit balance to discount the balance of users, over time, so that the total credit in the system will adjust to the true number of users in the system. This property of the system is shown in Figure 7.4. The mean arrival rate of users is 3.6 users per hour, and each user remains in the system for a mean period of 16.7 minutes. Figure 7.4 shows that the total credit in the system tracks the number of users in the system. The rate of decay to the actual number of the system is defined by the value of β.

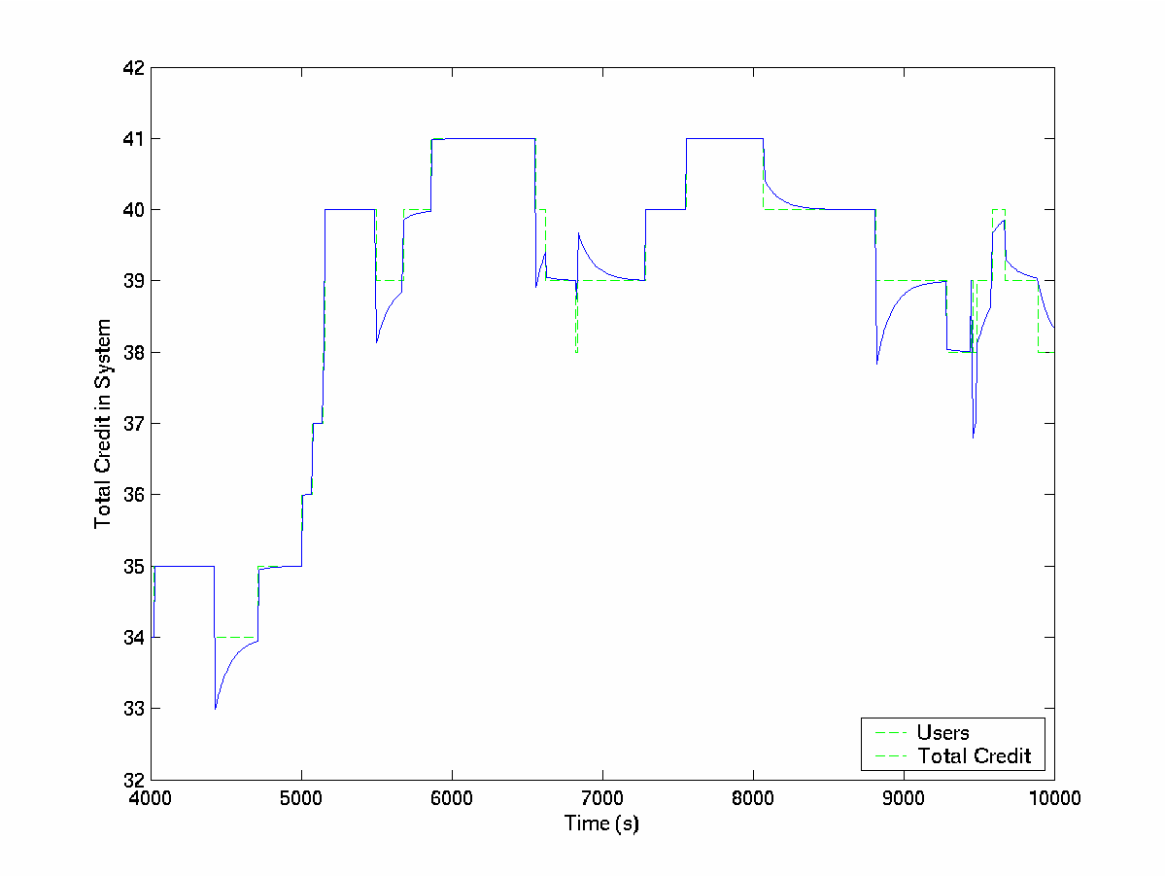


Figure 7.4: The number of users and the total credit in the dynamic network.

7.3.3: Mobile Network: User Prices and Overall Throughput

The final objective of this paper is to study the effect of mobility on the performance of our ad hoc system, where nodes have incentives to collaborate. Returning to the original topology considered earlier in Figure 7.1, the most extreme node N1 is mobilized, and follows the path, shown in Figure 7.5, through the geographical centroid of the static network consisting of the remaining nodes. We observe the performance of the system, as the N1 moves across the network and reaches the other edge of the network by the end of the simulation which is run for 10,000s. To reach this final location, the velocity of N1 is set to (-0.0074,0.0126)m/s.

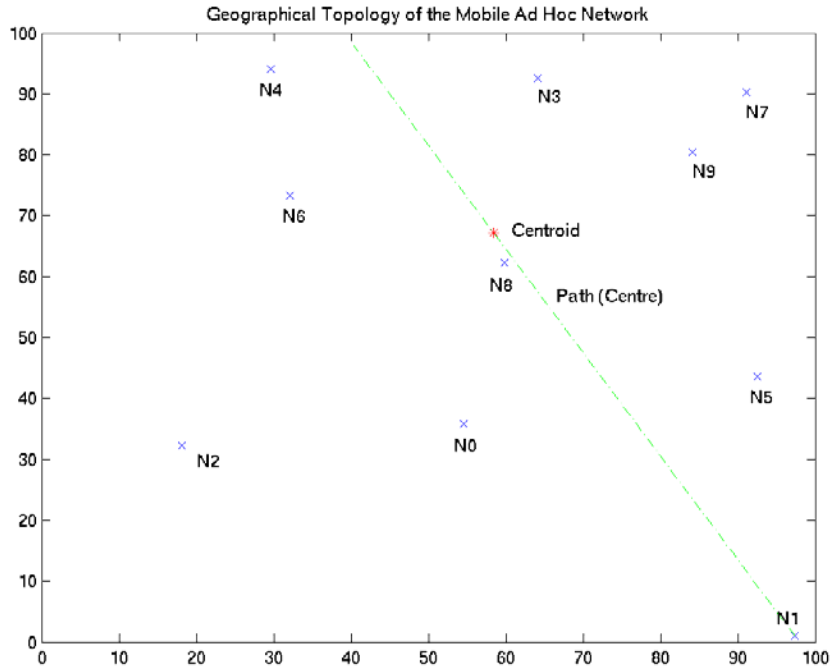


Figure 7.5: Path of mobile node through the centre of the network

As it approaches the centre of the network, *N1* will be used more frequently as a transit node to carry traffic between other nodes, and this can be observed from the increase in both the bandwidth and power price of node N1 in Figure 7.6. At the same time, other nodes will now have a choice of sending traffic through either N8 or N1, when both nodes are near the centre of the network, so the effect of N1 moving to the centre is to reduce substantially the power price of N8. As node N1 moves away from the centre of the network, these effects on the node prices subside.

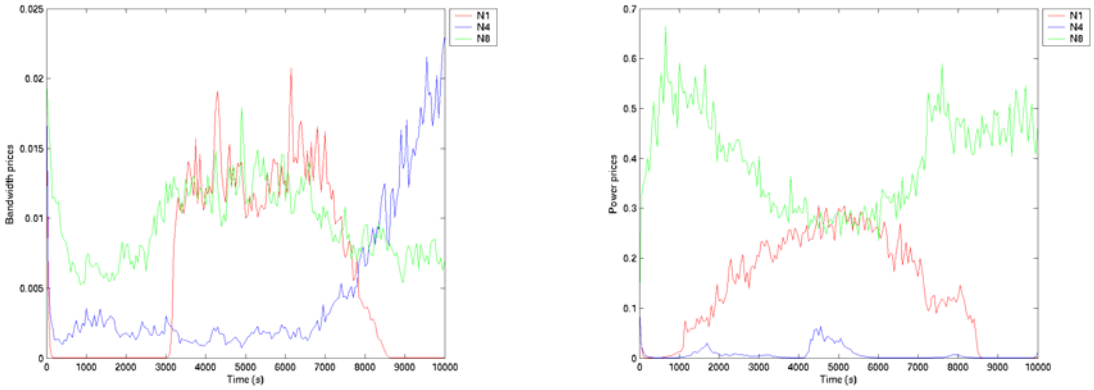


Figure 7.6: Bandwidth and power prices curves of the mobile node N1, and two stationary nodes N4 and N8

The increase in prices associated with node N1, when it is near the centre of the network, and its increased traffic load which it forwards for other nodes, means that its credit balance also grows, as shown in Figure 7.7. This increases the ability of N1 to generate traffic, as its willingness-to-pay is related to its credit balance. Consequently, its throughput increases, as can be observed in Figure 7.8. Due to the competition between N1 and N8, the credit balance of node N8 decreases slightly. However, it should be noted that while N8's credit balance decreases, its overall throughput increases. This is principally due to the fact that N1 is now much closer to N8, and so the actual cost of sending traffic to N1 becomes substantially less. This results in increasing the traffic load between N1 and N8, and so the bandwidth price of node N8 increases accordingly. This increase in throughput and bandwidth price, when a node moves closer to another particular node, is also observed in Figures 7.6 and 7.7 when N1 moves away from the centre of the network and closer to node N4.

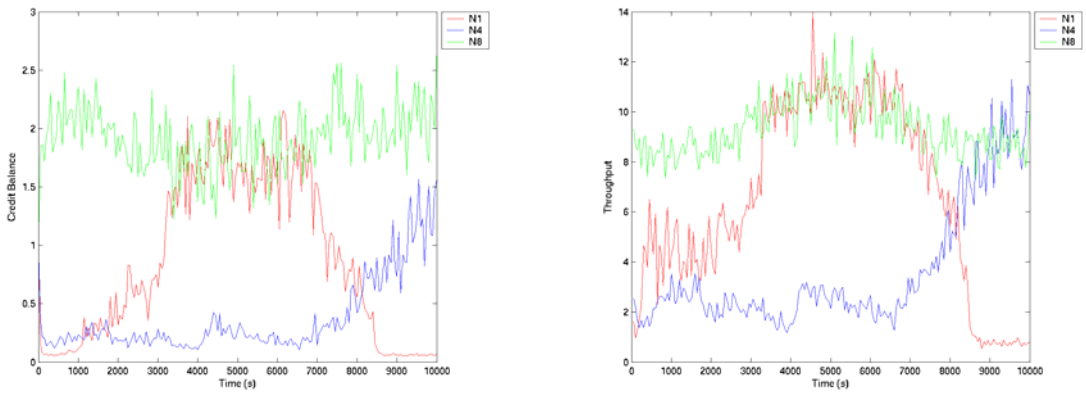


Figure 7.7: Credit balances and throughput curves of the mobile node N1, and two stationary nodes N4 and N8

The remaining question is whether the mobility of node N1 through the centroid of the network effectively increases the overall throughput of the system. Figure 7.8 shows that the overall throughput within the network has increased as N1 moves towards the centroid of the network. In comparison, when N1 moves from the centre to the edge of the network, the overall throughput decreases. Thus our results indicate ways in which the overall performance varies with the current geographical distribution of the users. Moreover, mobile users can affect not just their own performance, but also the overall performance of the network.

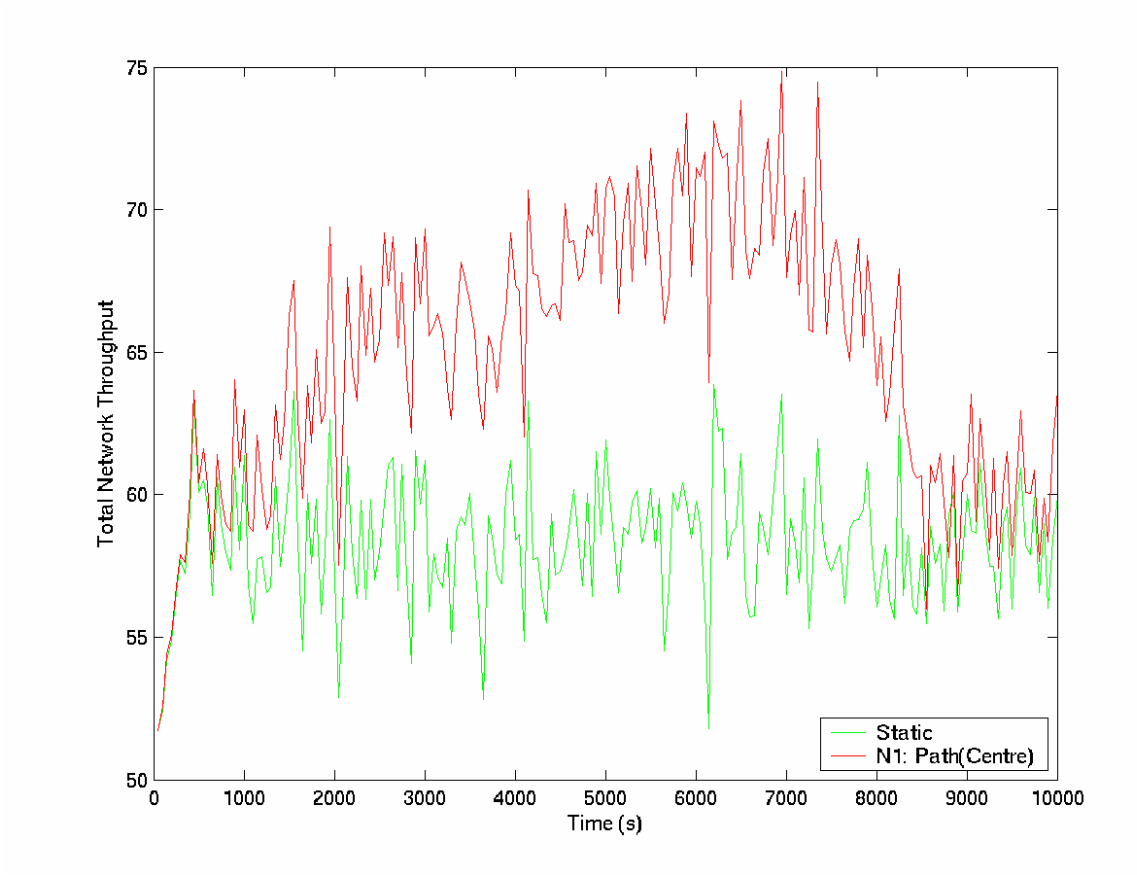


Figure 7.8: Comparison between the total throughput for the static network, and the networks with nodes N1 and N8 following mobility paths.

7.4. Conclusions

We have considered how incentives can be integrated into the operation of a mobile ad hoc network, so that the cost of resources consumed at transit nodes, when forwarding traffic along multi-hop routes, can be recovered using pricing mechanisms. These prices are determined in a distributed fashion, where algorithms are used by individual users to update their prices based on their bandwidth and power usage. Routes for connections from a user to a particular destination are chosen such that the route price is minimal. This forms a dual algorithm for traffic management within the network.

Incentives for collaboration have been provided through the concept of a user having a credit balance, which receives an initial endowment when the user joins the network. The credit balance accumulates notional credit accrued by forwarding traffic for other users, while any traffic generated from a particular user decreases the credit balance based on the cost of forwarding the traffic to its destination. The amount of traffic that a user can generate is directly related to its current credit balance---hence the user's incentives to both act as a transit node for other users and move to locations within the network where it can forward more traffic.

In this section, we have studied this system through fluid-level simulations. These simulations have demonstrated that user' prices and credit balances stabilize for a static ad hoc network and shown the advantages in being near the centre of the network, as this allows nodes to act as transit nodes for a larger number of routes. We have also shown that mobility through the centre of the

network can increase an individual user's throughput, as well as increasing the overall throughput of the system.

Further work includes exploring analytically the stability of the model, with the view of selecting appropriate parameters for updating user prices and discounting their credit balances. Further work also includes attempting to incorporate the effects of delays and interference. It would also be of interest to investigate re-routing protocols that minimize the routing information that needs to be distributed in the network, while at the same time achieving near-minimal cost routing. In general, we have found that our model captures many of the fundamental trade-offs within the collaborative setting of an ad hoc network.

7.5. References

[BH03] L. Buttyan and J. Hubaux "Stimulating Co-operation in Self-organizing Mobile Ad hoc Networks", *ACM/Kluwer Mobile Networks and Applications*, Vol. 8, No. 5, Oct 2003.

[CT00] J. Chang and L. Tassiulas "Energy Conserving Routing in Wireless Ad-hoc networks" *INFOCOM*, March 2000.

[CGKO03] J.Crowcroft, R. Gibbens, F. Kelly & S. Östring "Modelling Incentives for Collaboration in Mobile Ad Hoc Networks", Accepted for *Performance Evaluation*, August 2003

[GK99] R. Gibbens and F. Kelly "Resource Pricing and the Evolution of Congestion Control" *Automatica*, Vol. 35, pp 1969-1985, 1999.

[GT02] M. Grossglauser and D. Tse "Mobility Increases the Capacity of Ad-hoc Wireless networks" *IEEE/ACM Transactions on Networking*, Vol 10, No 4, pp 477-486, Aug. 2002.

[GK00] P. Gupta and P. Kumar "The Capacity of Wireless Networks" *IEEE Transactions on Information Theory*, Vol 46, No. 2, pp 388-404, Mar. 2000.

[JT01] R. Johari and D. Tan "End-to-end Congestion Control for the Internet: Delays and Stability" *IEEE/ACM Transactions on Networking*, Vol 9, No 6, Dec. 2001.

[JMHJ02] D. Johnson, D. Maltz, Y. Hu and J. Jetcheva "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR). *IETF Internet Draft* `draft-ietf0manet-dsr-07.txt`. Feb 2002.

[K00] F. Kelly "Models for a Self-managed Internet" *Philosophical Transactions of the Royal Society of London*, Vol A358, pp 2335-2348, 2000.

[KMT98] F. Kelly, A. Maulloo and D. Tan "Rate Control for Communication Networks: Shadow prices, proportional fairness and stability" *Journal of the Operational Research Society*, Vol 49, No. 3, pp 237-252, March 1998.

[LWC02] R. Liao, R. Wouhaybi, and A Campbell, "Incentive Engineering in Wireless LAN-based Access Networks" *ICNP* Nov 2002.

[LL99] S. Low and D. Lapsley "Optimization Flow Control I: Basic algorithm and convergence" IEEE/*ACM Transactions on Networking*, Vol 7, No 6, pp 861-874, Dec. 1999.

[MB02] P. Marbach and R. Berry, "Downlink Resource Allocation and Pricing for Wireless Networks" *INFOCOM*, June 2002.

[PBD02] C. Perkins, E. Belding-Royer and S. Das, "Ad hoc On-demand Distance Vector (AODV) Routing" *IETF Internet Draft* `draft-ietf-manet-adov-12.txt`, Nov 2002.

[QM03] Y. Qui and P. Marbach "Bandwidth Allocation in Ad Hoc networks: A price-based approach" *INFOCOM*, March 2003.

[S02] V. Siris "Resource Control for Elastic Traffic in CDMA Network" *MOBICOM* Sept 2002.

[SCNR02] V. Srinivasan, C. Chiasserini, P. Nuggehalli and R. Rao, "Optimal Rate Allocation and Traffic Splits for Energy Efficient Routing in Ad Hoc networks" *INFOCOM*, June 2002.

[WPL03] W. Wang, M. Palaniswai and S. Low "Optimal Flow Control and Routing in Multi-path Networks" *Performance Evaluation*, Vol 52, No 2, pp 119-132, April 2003.

8. APPENDIX B: CHOICE OF THE ACCESS TECHNOLOGY DEVELOPMENT SYSTEM

In order to choose an appropriate access technology development system, a detailed investigation of the existing technologies and products has been conducted.

8.1. Recall of the project goals concerning wireless access technologies

It is useful to remember first that the MobileMAN project (SUPSI-DIE) goals regarding access technologies is not the development of a complete new wireless access technology (i.e. spanning from the MAC to the RF), but rather to provide a research-, test- and experimentation-bed (proof of concept) for new techniques, especially in the MAC protocol domain. Synthetically stated:

- Provide a development platform for the new protocol:
  - Start with an existing state-of-the-art 802.11/a/b solution; modify the MAC protocol software.
  - Develop only the required software; if necessary choose new processor: ARM9, DSP...)
- Provide a small amount of integrated test/demo devices to project consortium:
  - PC-cards or at least small "portable" systems.
- Contribute to the development/proposition of "new" standards:
  - Contribute to a new 802.11x standard, which will be (perhaps) proposed by the project consortium.
  - Provide guidelines for the hardware development.
  - Provide development system specifications.

8.2. Development strategy

We first of all selected an appropriate development strategy, i.e. a strategy that would a) allow to reach the required technical goals within the project time-span, b) contain the costs into the project approved budgets, c) suited for future modifications and enhancements both from the hardware and the software point-of-view, this in order to be able the use of the developed system beyond the project scope and time-span (for instance in follow-on projects) for researching new wireless access technologies. We therefore identified 3 possible (for the MobileMAN project) development strategies:

- *Totally ad-hoc development*: develop an access board from scratch using off-the-shelf components (chips). While theoretically very good (flexible, independent...), this strategy has quickly proven unfeasible given the project financial and temporary limits. A new WLAN card, provided one can obtain the chips at small quantities (which is not often the case), takes way too much time to develop and has prohibitive non-recurring costs (order of several 100 k€), moreover the development risks are also very high.
- *Step-by-step refinement*: take an-off-the-shelf product, which is suitable to be modified for the project scope and begin with replacing the MAC software. If the MAC processor is not powerful enough, replace the MAC processor itself with on that fits the task. This solution has the advantage of a moderate cost (since the system is anyway already been developed); moreover, the simple replacement of the MAC processor is relatively simple. Finally this solution has room for future enhancements, because BB and/or RF parts could also be replaced in case of new research projects in the BB/RF area.
- *Customization of an existing solution*. One possibility for researching new MAC algorithms with standard equipment would be to take an off-the-shelf WLAN-card, bypass the MAC processor (on some of them it is possible) and execute all the MAC software on the host processor. This

solution has the advantage of a relatively short development time, which is sure in line with the MobileMAN time-span. The system is also easily modifiable, i.e. the BB/RF parts may be replaced (however, it is not sure that both the hardware and software interfaces to the MAC could stay unchanged). This solution is however not totally free as it may seem: some vendors charge large sums only to give the specifications and API for the MAC/BB chipset. Moreover, since we don't know yet the exact amount CPU power we will have on the host processor, troubles could result if the host processor will not be powerful enough; a real-time kernel could possibly be required under the normal OS (i.e. Linux) in the extreme case.

In order to choose both the technology and the development strategy, deep evaluation of the WLAN technologies currently available on the market was first performed.

8.3. Current technologies evaluation

Most of the today's (on the market) WLAN technologies have been investigated and classified according to different criteria. Roughly speaking, the devices may be classified in 3 main categories:

- *Development systems*. These are complex systems, mostly large IP-blocks (Intellectual Property) libraries, including both software and hardware (VHDL) modules, which are, used in conjunction with complex experimentation boards (embedding many digital and analog ICs, FPGAs...). Development systems are specifically designed to allow the rapid prototyping of complete WLAN systems (form the RF front-end to the MAC protocol stack); once a system has been sufficiently tested with the development system, ICs may be developed by retargeting the IP-blocks to a specific VLSI technology. Development systems are mainly used by researchers and large companies developers in order to quickly verify new architectural solutions for existing systems (or even for new systems, i.e. new standards) before going to silicon.

  Examples of development systems are BOPS Inc., DCM Technologies and Tality.

  Although a WLAN development system could appear a good solution for the MobileMAN project, their prices are prohibitively high. Moreover, the MobileMAN project goal is, as stated above, more the step-by-step improvement of current WLAN technologies rather than the development of a totally new access technology. For these reasons, development systems have been discarded.

- *Chipsets*. First-generation chipsets consist of many devices (at least 3: MAC, BB and RF; usually 7 or more, not counting external RAM and FLASH memories), which are embedded into one or more boards. Commercial systems normally use chipsets in the first production years, since totally integrated solutions are not available. Examples of chipsets are reported below.

  Intersil Prism-I and Prism-II. The Intersil Prism-I and Prism-II chipsets are ideal for testing developing WLAN systems, since they give access to many points of the dataflow. The Prism-I chipset supports the IEEE 802.11 standard, i.e. 1 and 2 Mbps bitrates in the IMS band (2.4–2.48 MHz). The Prism-II supports the IEEE 802.11b standard, i.e. 1, 2, 5.5 and 11 Mbps bitrates in the IMS band (2.4–2.48 MHz).

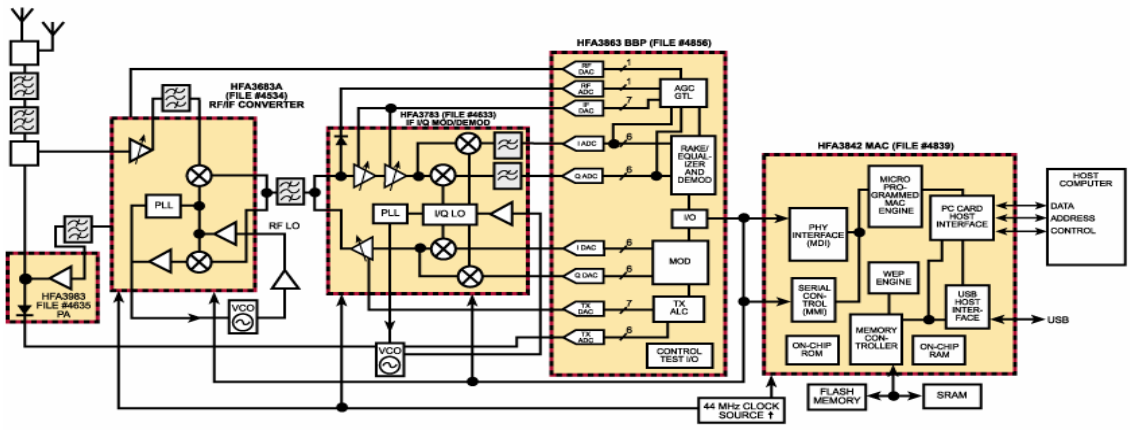


Figure A-1 The Intersil Prism-II chipset.

SystemOnIC Tondelayo1. The SystemOnIC Tondelayo1 chipset is a new and modern architectural evolution for the WLAN implementation which supports the IEEE 802.11a/b standard, i.e. 1, 2, 5.5, 11 and 54 Mbps bitrates in the IMS band (2.4–2.48 MHz) and in the 5 GHz band. Tondelayo1 is based on DSP; for this reason it is fairly easy to modify the modem behavior. It must be noted that part of the MAC protocol may be executed by the hardware, while the high level parts must be

executed by the host processor; this may be an advantage since it gives the user an extended control, but it may set very hard real-time constraints on the host operating system.

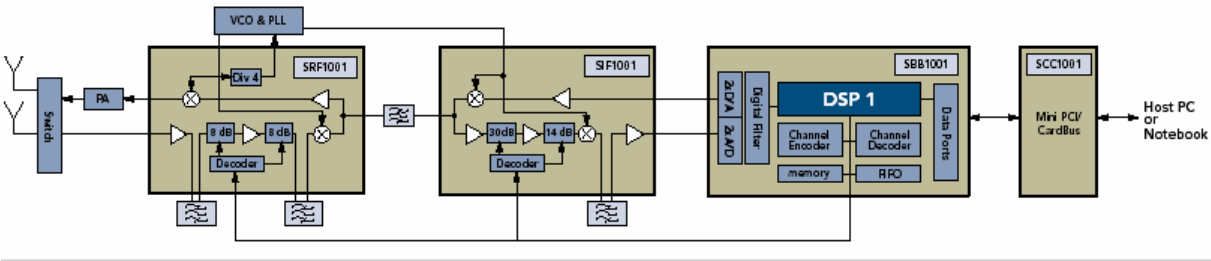


Figure A-2 The SystemOnIC Tondelayo1 802.11a/b chipset.

A chipset-based solution is clearly a very good solution for the MobileMAN project needs, since it guarantees full access to the internal resources; one can start with a commercial system (e.g. an evaluation kit) and implement any new MAC algorithm by reprogramming the MAC processor (or by replacing it with a more suited processor (e.g. an ARM9). Later, even the BB and RF parts may be replaced if necessary, allowing a modular and gradual system development.

- *Fully integrated (single-chip) solutions*. Once a WLAN technology get widespread acceptance (millions of pieces are sold), the development and mass-production of single-chip solutions is economically advantageous (in reality, single-chip solutions are very rare, since RF and BB/MAC usually require different VLSI technologies, i.e. bipolar or SiGe for the RF part and CMOS for the MAC/BB part).

  Examples of fully integrated solutions are: Intersil Prism-III, Intersil Prism-IV, Intersil Indigo (Prism-V), Atheros AR5000, Envara WiND502, Mobilian TrueRadio, Texas Instruments TnetW1100B, Synad Mercury5G, ATMEL AT76C502A.

  Fully integrated solutions could be used in MobileMAN only if a sufficient access would be granted to the chip internal resources (drivers, documentation, ...), which is practically never the case.

A *summary* of the existing technologies with advantages and disadvantages is presented below:

Category	Company	Model	Main advantages	Main disadvantages
Development system	BOPS Inc.	WirelessRay	Flexibility	Price, complexity, overkill
Development system	DCM Technologies	Wi Talk	Flexibility	Price, complexity, overkill
Development system	Tality (Cadence)	IEEE802.11 MAC IP	Flexibility	Price, complexity, overkill
Chipset	Intersil	Prism-I	Flexibility (access to internal resources)	Availability (access to the technology as developer)
Chipset	Intersil	Prism-II	Flexibility (access to internal resources)	Availability (access to the technology as developer)
Fully integrated	Atheros	AR500	Price	Flexibility, availability
Fully integrated	ATMEL	AT76C502A	Price	Flexibility, availability
Fully integrated	Envara	WiND502	Price	Flexibility, availability
Fully integrated	Intersil	Prism-III	Price	Flexibility, availability
Fully integrated	Intersil	Prism-IV	Price	Flexibility, availability
Fully integrated	Intersil	Indigo (Prism-V)	Price	Flexibility, availability
Fully integrated	Mobilian	TrueRadio	Price	Flexibility, availability
Fully integrated	Synad	Mercury5G	Price	Flexibility, availability
Fully integrated	Texas Instruments	TnetW1100B	Price	Flexibility, availability

Table A-1 WLAN technology evaluation summary.

8.4. Choice of a development system

The criteria for the choice of the development system are many, the most important of them are:

- Cost.
- Short delivery time.
- Flexibility.
- Availability in small quantities.
- Availability of technical documentation.

Given the above criteria (cost, time and flexibility are the most stringent for MobilMAN), it has been decided to chose a deelopment system based on a chipset. Moreover, given Intersil widespread acceptance (> 50% of the market) and low integration level in the first generation chipsets, a Prism-I or Prism-II has been preferred.

After having researched the market for boards, development systems or products which could fit the project needs, a products by Elektrobit AG that filled all the criteria was retained.

The chosen system is the DT-20 wireless modem from Elektrobit AG. The system is based on the Intersil Prism-I chipset and supports the 802.11 standard (1 or 2 Mbps in the IMS band).

Figure A-3 The Elektrobit DT-20 wireless modem.

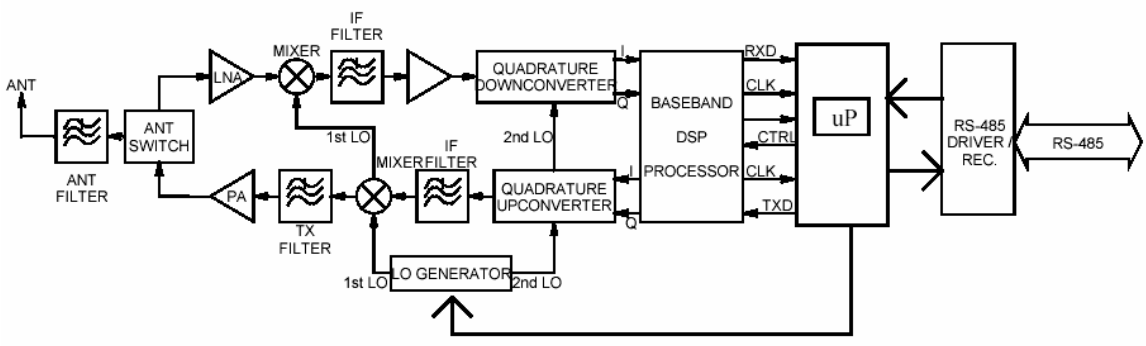


Figure A-4 DT-20 modem block diagram.

The DT-20 modem has a full-duplex asynchronous serial data interface to the host equipment. The interface includes differential Tx-data and Rx-data signals. This interface is used to carry both the transmit/receive data and the modem control information. A proprietary framing and protocol is implemented in the small DSP microcontroller. The radio (BB and RF) part is based on a commercial spread-spectrum chipset (Intersil Prism-I) and consists of a spread-spectrum baseband processor, a quadrature up/down converter, IF filter stages, LO generators, a RF up/down converter and LNA and power amplifier stages after the antenna switch.

Both DSP microcontroller and host serial interface are clearly to be replaced for the MobileMAN project needs. DSP and BB communicate via 2 bi-directional synchronous serial lines; these will remain the final system. After the required modification, the modified DT-20 modem (DT-MAN) will have the following block diagram:

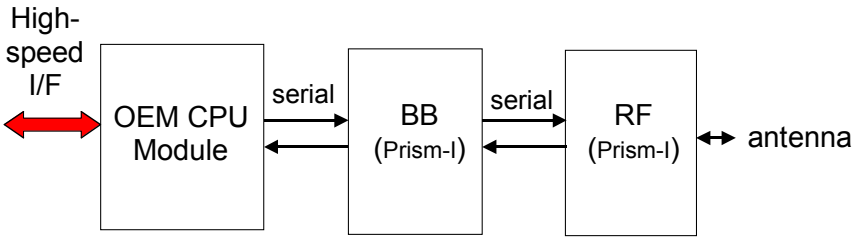


Figure A-5 DT-MAN (modified DT-20) WLAN modem block diagram.

8.5. Choice of an OEM CPU module

In order to accommodate the stringent real-time requirements of the 802.11x MAC, a powerful and flexible CPU module must be chosen. Moreover, given the speculative nature of the MobileMAN project, it is also advisable to choose a powerful enough CPU in order to allow future extensions and quick tryouts of new MAC ideas.

A list of minimal *computing* requirements for the OEM CPU module; this has been done by carefully analyzing the 802.11x standard family and the new MAC algorithms family provided by MobileMAN partner CNR. With respect to this, we have:

- CNR MAC only "improves" 802.11x MAC (v2), no radical changes.
- Still CSMA.
- Modified CA scheme.
- No more exponential (radix 2) contention window.
- Estimation of network status (# terminals, TX and RX power…)

- Each "tick" (SlotTime=20 μs for 802.11 and 802.11b, 9μs for 802.11a) decision (based on statistical model) if TX or not.
- Rest of protocol unchanged (CTS, ACK, DIFS, SIFS…)

Therefore, computing requirements and time-constraints are:

1) The SIFS, (short interframe space), that is 10μs long, in which the MAC has to receive the message, decodes it and prepares the response, and eventually transmits the response.
2) The SlotTime, that is 20μs long, in which the MAC has to perform the new backoff algorithm; in this case the MAC has to carry out the following operations in floating point:
   - 15 multiplications.
   - 3 divisions.
   - 1 square root.

It is important to notes that the SIFS time occurs between the last bit of the message received and the first bit of the message transmitted and it is the shorter interframe time utilized during the following type of communication sequencing:

- RTS received, CTS transmitted.
- DATA received, ACK transmitted.
- ACK received, DATA transmitted.
- DATA fragment received, ACK transmitted. (The data fragmentation is needed when the source data buffer is longer then the maximum data field in the MAC frame.)
- ACK received, DATA fragment transmitted.

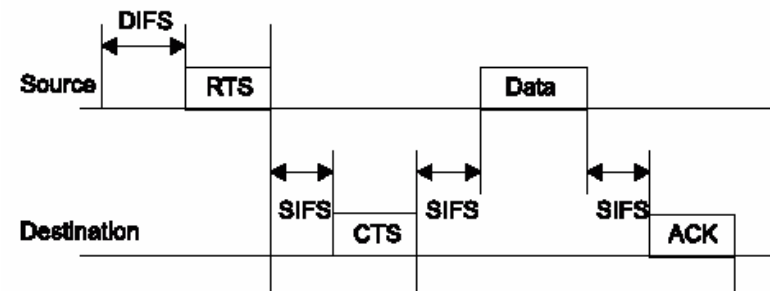


Figure A-6 DIFS and SIFS intervals.

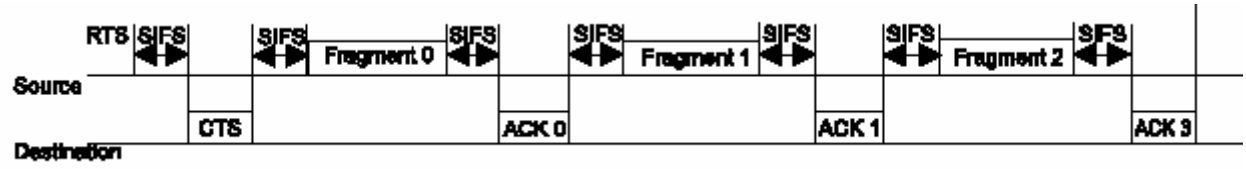


Figure A-7 SIFS and ACK intervals.

The Frame Check Sum (FCS) function is computed by the MAC during the receiving stage, bit by bit, so when the receiving stage is finished, i.e. when the SIFS time interval starts, the MAC has to do a few number of operations to terminate the FCS function. After that the MAC has to prepare the response, that could be CTS, ACK, DATA or a new fragment of DATA, which requires again a few number of operations, and wait the end of the SIFS time interval to send it back.

Obviously, the data fragmentation has to be performed before the first transmitting phase, RTS/CTS, and the data fragments must be stored in a buffer ready to be sent.

Even if the tasks are not too critical for a modern embedded microcontroller (e.g. ARM9, TIC6701...) during the fragmentation transmitting sequence the MAC must perform more operations just to check the right fragment sequence. The main constraints are:

- Main critical task is on reception and requires.
  - On-line computation of FCS.
  - Replacement of packet header (addresses) on the packet in memory.
  - Send ACK (or NACK).
- May be performed concurrently with packet reception (at each receive FIFO read)
- Packets may be prepared into RAM (frame buffer).

Accordingly to this, a list of minimal *hardware/software requirements* for the CPU module has also been compiled:

- Powerful CPU: floating point is preferable.
- Very fast host-interface: USB, Ethernet or FireWire is advisable.
- 2 full duplex high-speed synchronous serial interface (≥ 2Mb/s) for connecting the Base-Band (I2C, SPI, SSI…)
- The synchronous serial interface should also work is non-standard synchronous modes.
- General Purpose I/O (≥ 8 pins, 16 would be better.)
- External Interrupts (≥ 2.)
- External FPGA (preferable for extra functions and future enhancements.)
- 64-bit timer (counter) for timestamps.
- Timer (counter) for the Beacon Interval (0.1 s normally used); could be implemented on external FPGA.
- Easy to use development system (IDE), connecting a development PC via J-TAG I/F.
- Easy to reprogram (quick tryouts of new MAC algorithms and other protocol enhancements.)

Evaluation of the existing OEM CPU modules

Several commercial (off-the-shelf) OEM modules, mainly based on DSP or RISC processors, have been investigated and checked against the above listed criteria; the investigation result for the most interesting boards is reported in the following table:

Company	Model	CPU	Main advantages	Main disadvantages
Keith&Koep	Trizeps	Intel SA-1110	High number GPIO 3 host interface: PCMCIA, USB, JTAG	No floating point No ext. FPGA Serial interface for BB too slow (< 2Mbps)
Keith&Koep	Trizeps II	Intel XScale PXA250	High number GPIO 3 host interface: PCMCIA, Ethernet, JTAG	No floating point No ext. FPGA Serial interface for BB too slow (< 2Mbps)
Intrinsyc	CerfCube	Intel SA-1110	High number GPIO 3 host interface: PCMCIA, USB, JTAG	No floating point No ext. FPGA Serial interface for BB too slow (< 2Mbps)
Intrinsyc	CerfBoard	Intel XScale PXA250	High number GPIO 3 host interface: PCMCIA, Ethernet, JTAG	No floating point No ext. FPGA Serial interface for BB too slow (< 2Mbps)
Technologic Systems	586 SBC	AMD Elan 520	Floating point High number GPIO	No ext. FPGA No serial interface for BB

Orsys	C6713 Compact	TI C6713	Floating point Ext. FPGA Development tools well known High number GPIO	Host interface only with FireWire
Samsung	S3C2400X	ARM920T	High number GPIO 4 host interface: 3 USB, JTAG	No floating point No ext. FPGA Not useful serial interface for BB
Motorola	MCF5272	M5272 ColdFire	3 host interface: PCMCIA, Ethernet, JTAG	No floating point No ext. FPGA Low clock frequency

Table A-2 OEM CPU modules comparison.

The chosen board is Orsys C6713 Compact board (see Figure A-6 and Figure A-7), which is based on a very powerful TI C6000 platform DSP. The main reasons for this choice are:

- Max flexibility.
- High computation power (also in floating point).
- Large on-board FPGA for high-speed MAC operations and hardware non-standard serial interfaces.
- Well known development system (Code Composer Studio.)
- Easy and fast to reprogram.

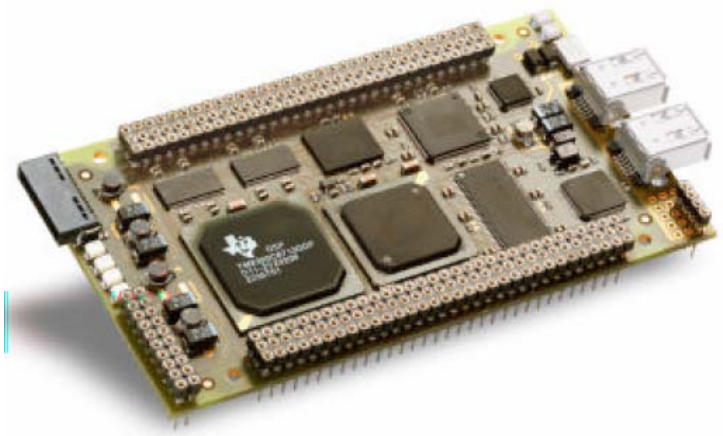


Figure A-8 Orsys C6713 Compact board picture.

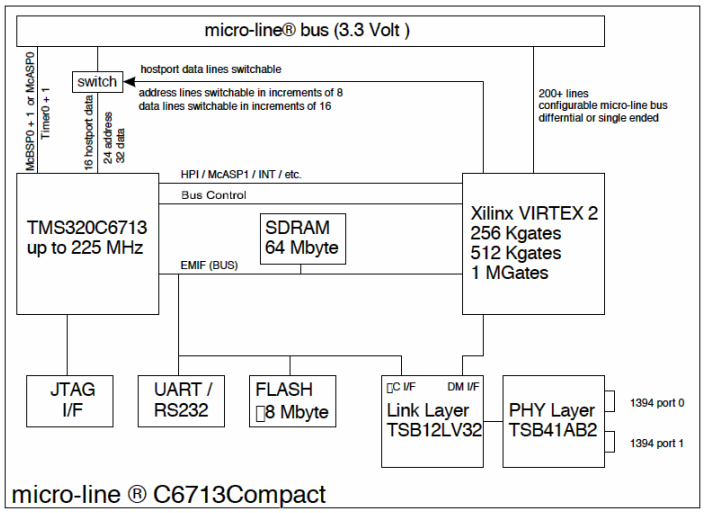


Figure A-9 Orsys C6713 Compact board block diagram.

8.6. References

[Atheros2002] AR500 Advanced Data Sheet, July 2002, www.atheros.com.

[ATMEL2002] AT76C502A Summary Document, June 2002, www.atmel.com.

[BOPS2001] WirelessRay Data Sheet, September 2001, www.bops.com.

[DCM2001] MAC, BB Product Brief, www.dcmtech.com.

[Envara2002] EN303 Technical Brief, September 2002, www.envara.com.

[Intersil2000] Prism-I -II Data Sheet, January 2000, www.intersil.com.

[Intersil2002] Prism Indigo Preliminary Data Sheet, 2002.

[Intrinsyc2002] http://www.intrinsyc.com/products/cerfboard/.

[Intrinsyc2002a] http://www.intrinsyc.com/products/cerfcube/.

[Keith2001] Trizeps Data Sheet, December 2001, www.keith-koep.com.

[Keith2002] Trizeps II Preliminary Data Sheet, March 2002.

[Mobilian2002] TrueRadio Product Brief, September 2002, www.mobilian.com.

[Motorola2002] MCF5272 User's Manual, March 2002, www.motorola.com.

[Orsys2002] C6713 Compact product preview, February 2002, www.orsys.de.

[Samsung2002] S3C2400X User Manual, September 2002, www.vitals.co.kr.

[Synad2002] Mercury5G Preliminary Information, June 2002, www.synad.com.

[Technologic2002] www.embedded586.com.

[TexasInstruments2002] TnetW1100B Product Brief, September 2002, www.ti.com.

9. APPENDIX B: A SURVEY OF EXISTING MIDDLEWARE

9.1. Lime: Linda in a Mobile Environment

Lime [MPR01] is a Java-based middleware that provides a coordination layer useful for designing applications that exhibit either logical and/or physical mobility. Lime is specifically targeted toward the complexities of the ad hoc mobile environment, where resource availability constantly changes (both data and computational elements) as hosts move through space and mobile agents move among hosts. The main goal of Lime is to simplify programming having the middleware handling (on behalf of the programmer) many of the complexities. The programming model is a shared memory in the style of Linda tuple-spaces, used to achieve coordination in classic distributed programming. In Linda, processes communicate by writing, reading, and removing data from a tuple space that is assumed to be persistent and globally shared among all processes. Lime adapts this notion to a mobile environment by breaking up the notion of a global tuple space, and distributing its contents across multiple mobile components.

In Linda, a *tuple space* is a global and persistent repository of *tuples*, essential data structures constituted by an ordered sequence of typed fields. The repository is referred to be global, as every concurrent process in the system is supposed to have access to it, and persistent as its lifetime is independent from the one of the processes. A minimal interface is provided to operate on the tuple-space:

- A function to write a tuple to the tuple space (`out`)
- A function to get a copy of a tuple in the tuple space that matches a given pattern (`rd`)
- A function to get a copy of a tuple in the tuple space that matches a given pattern and additionally remove it (`in`)

The last two operations are blocking, (i.e., if a matching tuple is not found the caller process is suspended until a matching tuple appears in the tuple space), and in case of multiple matching tuples, one is returned non-deterministically.

Lime adapts this fundamental model of coordination and communication to encompass both physical and logical mobility. In Lime, agents (the active components in the system) can roam across mobile hosts (which act as mere containers for the agents) which can roam across the physical space. The presence of mobility prevents the existence of a global and persistent tuple space. In Lime, each mobile agent owns a *Lime tuple space (LTS)* which follows the agent during migration. The notion of a global and persistent tuple space is dynamically recreated on a host by merging the LTSs of all the mobile agents there colocated, thus created a *host-level transiently shared tuple space*. Similarly, it is recreated across hosts by merging the host-level tuple spaces into a *federated transiently shared tuple space*. The configuration of these tuple spaces, i.e., their contents, are dynamically reconfigured by the system when an agent arrives or connection is established (tuple space *engagement*) and when an agent leaves or some host gets disconnected (tuple space *disengagement*). This way, Lime provides the programmer with the illusion of a global and persistent tuple space, which can still be accessed using the conventional Linda operations. This tuple space is referred to as the federated tuple space.

Lime introduces also the concept of tuple *location*, as the federated tuple-space is distributed across the mobile hosts. With this extension to the Linda interface, programmers can specify the source and the destination location (host) of the tuple functions (`out, in, rd`).

Finally, Lime introduces also the concept of *reactions* to cope with the environment dynamics. A reaction $R[\omega, \lambda](s,p)$ means that when there is a tuple matching p with location information $[\omega, \lambda]$, then the system Lime executes the action s. The (`in`) operation offers itself a certain degree of reactivity, but forces the calling process to block on the call until a match is found, or to poll the

tuple space (to avoid blocking). With reactions the programmer can register actions to the system, specifying also *when* (after which change on the federated tuple-space) they have to be fired.

9.2. Xmiddle

Xmiddle [MCE02a] allows mobile hosts (i.e., PDAs, mobile phones, laptop computers or other wireless devices) to be physically mobile, while yet communicating and sharing information with other hosts. The system does not assume the existence of any fixed network infrastructure underneath. Mobile hosts may come and go, allowing complicated ad-hoc network configurations. Connection is symmetric but not transitive as it depends on distance; for instance host H_A can be connected to host H_B, which is also connected to host H_C. However, host H_A and host H_C may be not connected to each other. Multi-hop scenarios, where routing through mobile nodes is allowed, are not yet in this system scope.

In order to allow mobile devices to store their data in a structured and useful way, tree structures have been used for data representation. Trees allow sophisticated manipulations due to the different node levels, hierarchy among the nodes, and the relationships among the different elements which could be defined. Xmiddle defines a set of primitives for tree manipulation, which applications can use to access and modify the data.

When hosts get in touch with each other they need to be able to communicate. Xmiddle therefore provides an approach to sharing that allows on-line collaboration, on-line data manipulation, synchronization and application dependent data reconciliation. On each device, a set of possible access points for the owned data tree are defined so that other devices can link to these points to gain access to this information; essentially, the access points address branches of trees that can be modified and read by peers. In order to share data, a host needs to explicitly link to another host's tree. The concept of linking to a tree is similar to the mounting of network file systems in distributed operating systems to access and update information on a remote disk.

Access points to a host's tree are a set called *ExportLink*. For example, host H_i exports the branch A, and hosts H_j and host H_k link to it, expressing the wish to share this information with host H_i. The owner of the branch is still host H_i but the data in the branch can be modified and read by the three hosts. The way data sharing, data replication and reconciliation is allowed in Xmiddle depends, however, also on additional conditions related to the connection status among the hosts.

In order to share data, hosts need to be connected. Host H_A becomes connected with host H_B when they are in the same communication range. When two hosts are connected they can share and modify the information on each other's linked data trees. The whole system works like a *distributed version controller*, using XML as the interface definition language to describe the status of the controlled documents.

A host records the branches that it links from other remote hosts in the set *LinkedFrom*, and the hosts linking to branches of the owned tree in the set *LinkedBy*. These sets contain lists of tuples (host; branch) that define the host that is linking to a branch, and from whom a branch is linked, respectively.

Xmiddle supports explicit disconnection to enable, for instance, a host to save battery power, to perform changes in isolation from other hosts and to not receive updates that other hosts broadcast. Disconnection may also occur due to movement of a host into and out of reach area, or to a fault. In both cases, the disconnected host retains replicas of the last version of the trees it was sharing with other hosts while connected, and continues to be able to access and modify the data; the versioning system will provide consistent sharing and data reconciliation.

9.3. JXTA

JXTA [JXTA] is an open-source project maintained by Sun Microsystems, Inc. with the aim of building a general architecture with an open set of XML-based protocols for creating peer-to-peer

style network computing applications and services. The emphasis of the project is more on defining standard mechanisms to be used in peer-to-peer service development, instead of pushing particular policies.

At the highest abstraction level, JXTA technology is a set of protocols. Each protocol is defined by one or more messages exchanged among participants of the protocol. Each message has a pre-defined format, and may include various data fields.

In this regard, it is akin to TCP/IP. Whereas TCP/IP links Internet nodes together, JXTA technology connects peer nodes with each other. TCP/IP is platform-independent by virtue of being a set of protocols. So is JXTA. Moreover, JXTA technology is transport independent and can utilize TCP/IP as well as other transport standards.

To underpin this set of protocols, JXTA technology defines a number of concepts including peer, peer group, advertisement, message, pipe, and more. Some concepts are explained below:

Identifiers. JXTA uses UUID, a 128-bit datum to refer to an entity (a peer, an advertisement, a service, etc.). It is easy to guarantee that each entity has a unique UUID within a local runtime environment, but because no global state is assumed, there is no absolute way to provide a guarantee of uniqueness across an entire community that may consist of millions of peers.

Peers. A peer is any entity that can speak some of the protocols specified for a peer. This is akin to the Internet, where an Internet node is any entity that can speak the suite of IP protocols. As such, a peer can manifest in the form of a processor, a process, a machine, or a user. Importantly, a peer does not need to understand all the peers protocols. A peer can still perform at a reduced level if it does not support some protocols.

Advertisements. An advertisement is an XML structured document that names, describes, and publishes the existence of a resource, such as a peer, a peer group, a pipe, or a service. JXTA technology defines a basic set of advertisements. More advertisement subtypes can be formed from these basic types using XML schemas.

Messages. Messages are designed to be usable on top of asynchronous, unreliable, and uni-directional transport. Therefore, a message is designed as a datagram, containing an envelope and a stack of protocol headers with bodies. The envelope contains a header, a message digest, (optionally) the source endpoint, and the destination endpoint. An endpoint is a logical destination, given in the form of a URI, on any networking transport capable of sending and receiving datagram-style messages. Endpoints are typically mapped to physical addresses by a messaging layer. Such a message format is designed to support multiple transport standards. Each protocol body contains a variable number of bytes, and one or more credentials used to identify the sender to the receiver. The exact format and content of the credentials are not specified. For example, a credential can be a signature that provides proof of message integrity and/or origin. As another example, a message body may be encrypted, with the credential providing further information on how to decrypt the content.

Peer Groups. A peer group is a virtual entity that speaks the set of peer group protocols. Typically, a peer group is a collection of cooperating peers providing a common set of services. The specification does not dictate when, where, or why to create a peer group, or the type of the group, or the membership of the group. It does not even define how to create a group. In fact, the relationship between a peer and a peer group can be somewhat meta-physical. JXTA does not care by what sequence of events a peer or a group comes into existence. Moreover, it does not limit how many groups a peer can belong to, or if nested groups can be formed. It does define how to discover peer groups using the Peer Discover Protocol. There is a special group, called the *World Peer Group*, that includes all JXTA peers. This does not mean that peers inside this special group can always discover or communicate with each other — e.g., they may be separated by a network partition. Participation in the World Peer Group is by default.

Pipes. Pipes are communication channels for sending and receiving messages, and are asynchronous. They are also uni-directional, so there are input pipes and output pipes. Pipes are

also virtual, in that a pipe's endpoint can be bound to one or more peer endpoints. A pipe is usually dynamically bound to a peer at runtime via the Pipe Binding Protocol. This also implies that a pipe can be moved around and bound to different peers at different times. This is useful, for example, when a collection of peers together provide a high level of fault tolerance, where a crashed peer may be replaced by a new peer at a different location, with the latter taking over the existing pipe to keep the communication going. A *point-to-point* pipe connects exactly two peer endpoints together. The pipe is an output pipe to the sender and input pipe to the receiver, with traffic going in one direction only — from the sender to the receiver. A *propagate* pipe connects multiple peer endpoints together, from one output pipe to one or more input pipes. The result is that any message sent into the output pipe is sent to all input pipes. JXTA does not define how the internals of a pipe works. Any number of unicast and multicast protocols and algorithms, and their combinations, can be used. In fact, one pipe can be chained together where each section of the chain uses an entirely different transport protocol. Pipes have been designed to be asynchronous, unidirectional, and unreliable, because this is the foundation of all forms of transport and carries with it the lowest overhead. Reliability is guaranteed by the transport protocol currently supported by the available implementation of the platform, and hence the JXTA community didn't spend time in explicit implementation.

As mentioned before JXTA is a platform that primarily aims at specifying protocols to develop distributed peer-to-peer services and applications on top of deployed network technologies. The main effort of this project is hence towards a standardization of peer-to-peer software mechanisms. The purpose of the main six JXTA protocols is briefly discussed below.

Peer Discovery Protocol. This protocol enables a peer to find advertisements on other peers, and can be used to find any of the peer, peer group, or advertisements. This protocol is the default discovery protocol for all peer groups, including the World Peer Group. Peer discovery can be done with or without specifying a name for either the peer to be located or the group to which peers belong. When no name is specified, all advertisements are returned.

Peer Resolver Protocol. This protocol enables a peer to send and receive generic queries to find or search for peers, peer groups, pipes, and other information. Typically, this protocol is implemented only by those peers that have access to data repositories and offer advanced search capabilities.

Peer Information Protocol. This protocol allows a peer to learn about other peers' capabilities and status. For example, one can send a *ping* message to see if a peer is alive. One can also query a peer's properties where each property has a name and a value string.

Rendezvous Protocol. This protocol allows a peer to propagate a message within the scope of a peer group.

Pipe Binding Protocol. This protocol allows a peer to bind a pipe advertisement to a pipe endpoint, thus indicating where messages actually go over the pipe. In some sense, a pipe can be viewed as an abstract, named message queue that supports a number of abstract operations such as create, open, close, delete, send, and receive. Bind occurs during the open operation, whereas unbind occurs during the close operation.

Endpoint Routing Protocol. This protocol allows a peer to ask a peer router for available routes for sending a message to a destination peer. Often, two communicating peers may not be directly connected to each other. Example of this might include two peers that are not using the same network transport protocol, or peers separated by firewalls or NAT. Peer routers respond to queries with available route information, which is a list of gateways along the route. Any peer can decide to become a peer router by implementing the Endpoint Routing Protocol.

9.4. Platforms for distributed content organization

The systems described below represent ways of building distributed hash table over an Internet-like network technology. They mainly answer, in a scalable and distributed way, to the following

fundamental peer-to-peer dilemma: once I have some data D or some attributes A describing it, how do I map them consistently over a zone (physical area, logical area or set of nodes) of the distributed community in a *scalable* and *resilient* way?

This problem finds several important applications in peer-to-peer systems. For example, let's say D is the information describing a service functionalities and parameters needed to bind it, while A contains some service keywords. On the one hand the service provider has to have a way to publish D, while on the other a client peer needs a way to discover where to find a service matching the keywords specified in A. Techniques based on flooding service descriptions or service queries are not scalable, while the usage of brokers (third-party nodes holding service descriptions and answering service queries) is not resilient. The systems described below are three different solutions to this problem, based on the concept of distributed hash tables.

Content-Addressable Network [RFHKS01]. A *Content-Addressable Network* (CAN) is a virtual *d*-dimensional Cartesian coordinate space on a *d*-torus. This coordinate space is completely logical and bears no relation to any physical coordinate system. At any point in time, the *entire* coordinate space is dynamically partitioned among all the nodes in the system such that every node "owns" its individual, distinct zone within the overall space. This virtual coordinate space is used to store (key,value) pairs as follows. To store a pair (K, V), key K is deterministically mapped onto a point P in the coordinate space using a uniform hash function. The corresponding (key,value) pair is then stored at the node that owns the zone within which the point P lies. To retrieve an entry corresponding to key K, any node can apply the same deterministic hash function to map K onto point P and then retrieve the corresponding value from the point P. If the point P is not owned by the requesting node or its immediate neighbours, the request must be routed through the CAN infrastructure until it reaches the node in whose zone P lies. Efficient routing is therefore a critical aspect of a CAN.

Nodes in the CAN self-organize into an overlay network that represents this virtual coordinate space. A node learns and maintains the IP addresses of those nodes that hold coordinate zones adjoining its own zone. This set of immediate neighbours in the coordinate space serves as a coordinate routing table that enables routing between arbitrary points in this space.

Intuitively, routing in a CAN works by following the straight line path through the Cartesian space from source to destination coordinates. Each CAN node maintains a coordinate routing table holding the IP addresses and the virtual coordinate zones of its immediate neighbors in the virtual space. Intuitively, the immediate neighbors are the adjacent in the virtual space: in a *d*-dimensional space, those nodes with *d-1* coordinate ranges in common with the reference node, and only one dimension along which the coordinate ranges abut. Each CAN message will be routed according to its destination coordinates, calculated by the hash function and included with the message. At each routing step, the node forwards the message to the immediate neighbor with coordinates closer to the one of the message destination. In this mechanism, resilience is guaranteed by more than one possible path reaching the destination.

As the CAN space is divided amongst the nodes currently in the system, a new node that joins the system must be allocated to its own portion of the coordinate space. This is done by an existing node splitting its allocated zone in half, retaining half and handing the other half to the new node. The process takes three steps:

1. First the new node must find a node already in the CAN.
2. Next, using the CAN routing mechanisms, it must find a node whose zone will be split.
3. Finally, the neighbors of the split zone must be notified so that routing can include the new node.

A new CAN node first discovers the IP address of any node currently in the system. It is assumed that a CAN has an associated DNS domain name, and that this resolves to the IP address of one or more CAN bootstrap nodes. A bootstrap node maintains a partial list of CAN nodes it believes are currently in the system. To join a CAN, a new node looks up the CAN domain name in DNS to retrieve a bootstrap node's IP address. The bootstrap node then supplies the IP addresses of several randomly chosen nodes currently in the system.

The new node then randomly chooses a point P in the space and sends a JOIN request destined for point P. This message is normally routed inside the CAN, until it reaches the node in whose zone P lies. This current occupant node then splits its zone in half and assigns one half to the new node. The split is done by assuming a certain ordering of the dimensions in deciding along which dimension a zone is to be split, so that zones can be remerged when nodes leave. For a 2-d space a zone would first be split along the X dimension, then the Y and so on. The (key, value) pairs from the half zone to be handed over are also transfered to the new node.

Having obtained its zone, the new node has to join routing, learning the IP addresses of its coordinate neighbor set from the previous occupant. This set is a subset of the previous occupant's neighbors, plus that occupant itself. Similarly, the previous occupant updates its neighbor set to eliminate those nodes that are no longer neighbors. Finally, both the new and old nodes' neighbors must be informed of this reallocation of space. Every node in the system sends an immediate update message, followed by periodic refreshes, with its currently assigned zone to all its neighbors. These soft-state style updates ensure that all of their neighbors will quickly learn about the change and will update their own neighbor sets accordingly.

As well as joining a CAN, nodes could leave the system. A CAN should hence ensure that left zones are taken over by the remaining nodes. If the node gracefully quits the system it explicitly hands over its zone and the associated (key,value) database to one of its neighbors. If the zones are mergeable the result is a valid single zone, otherwise the zone is handed to the neighbor whose current zone is smallest, and that node will then temporarily handle both zones.

The CAN also needs to be robust to node or network failures, where one or more nodes simply become unreachable. This is handled through an immediate takeover algorithm that ensures one of the failed node's neighbors takes over the zone. However in this case the (key,value) pairs held by the departing node are lost until the state is refreshed by the holders of the data (this should happen at the application layer). Under normal conditions a node sends periodic update messages to each of its neighbors giving its zone coordinates and a list of its neighbors and their zone coordinates. The prolonged absence of an update message from a neighbor signals its failure. Once a node has decided that its neighbor has died it initiates the takeover mechanism and starts a takeover timer running. Each neighbor of the failed node will do this independently, with the timer initialized in proportion to the volume of the node's own zone. When the timer expires, a node sends a TAKEOVER message conveying its own zone volume to all of the failed node's neighbors. On receipt of a TAKEOVER message, a node cancels its own timer if the zone volume in the message is smaller that its own zone volume or it replies with its own TAKEOVER message. In this way, a neighboring node is efficiently chosen that is still alive and has a small zone volume.

Under certain failure scenarios involving the simultaneous failure of multiple adjacent nodes, it is possible that a node detects a failure, but less than half of the failed node's neighbors are still reachable. If the node takes over another zone under these circumstances, it is possible for the CAN state to become inconsistent. In such cases, prior to triggering the repair mechanism, the node performs an expanding ring search for any nodes residing beyond the failure region and hence it eventually rebuilds sufficient neighbor state to initiate a takeover safely.

Finally, both the normal leaving procedure and the immediate takeover algorithm can result in a node holding more than one zone. To prevent repeated further fragmentation of the space, a background zone-reassignment algorithm runs to ensure that the CAN tends back towards one zone per node.

Chord [SMKKB01]. Chord provides a distributed lookup mechanism built on top of a consistent hash function. The consistent hash function assigns each node and key an *m*-bit *identifier* using a SHA-1 algorithm as a base hash function. A node's identifier is chosen by hashing the node's IP address, while a key identifier is produced by hashing the key. Below, the term "key" refers to both the original key and its image under the hash function.

Similarly, the term "node" refers to both the node and its identifier under the hash function. The identifier length *m* must be large enough to make the probability of two nodes or keys hashing to the same identifier negligible.

Consistent hashing assigns keys to nodes as follows. Identifiers are ordered on an *identifier circle* modulo *2m*. Key *k* is assigned to the first node whose identifier is equal to or follows (the identifier of) *k* in the identifier space. This node is called the *successor node* of key *k*, denoted by *successor*(*k*). If identifiers are represented as a circle of numbers from *0* to *2^m -1*, then *successor*(*k*) is the first node clockwise from *k*. The circle is also called a *Chord ring*.

Consistent hashing is designed to let nodes enter and leave the network with minimal disruption. To maintain the consistent hashing mapping when a node *n* joins the network, certain keys previously assigned to *n*'s successor now become assigned to *n*. When node *n* leaves the network, all of its assigned keys are reassigned to *n*'s successor. No other changes in assignment of keys to nodes need occur.

Over the described distribution scheme, a simple but inefficient lookup procedure consists in having each node simply forwarding lookup queries to the immediate successor in the ring, until they end up in the node maintaining the correspondent (key, value) pair. The retrieval of the correct location using this approach, may however require the linear traversal of the entire ring (linear cost). By maintaining more per node routing information (not only the link to the successor in the Chord ring), the system implements a more efficient (logatritmic cost) lookup procedure. Let *m* be the number of bits in the key/node identifiers. Each node *n* maintains a routing table with up to *m* entries, called the *finger table*. The *i*-th entry in the table at node *n* contains the identity of the *first* node *s* that succeeds *n* by at least *2^i-1* on the identifier circle, i.e., *s = successor(n+2^i-1)*, where *1 <= i <= m* (and all arithmetic is modulo *2^m*). We call node *s* the *i*-th *finger* of node *n*. A finger table entry includes both the Chord identifier and the IP address (and port number) of the relevant node. Note that the first finger of *n* is its immediate successor on the circle. The lookup operation, extended to use finger tables, worksas follows: at each node *n*, if the key *k* in the lookup query falls between *n* and its successor, the procedure ends, returning the successor identifier (of course the hashed value). Otherwise, *n* searches its finger table for the node *l* whose ID most immediately precedes *k*, and then invokes the lookup procedure at node *l*. The reason behind this choice of *l* is that the closer *l* is to *k*, the more it will know about the identifier circle in the region of *k*.This scheme has two important characteristics. First, each node stores information about only a small number of other nodes, and knows more about nodes closely following it on the identifier circle than about nodes farther away. Second, a node's finger table generally does not contain enough information to directly determine the successor of an arbitrary key k.

When a node *n* joins the Chord ring, falling between nodes n_p and n_s ($n_p < n < n_s$), it acquires n_s as its successor, asking an existing node *l* to look that up. The identity of node *l* is "assumed" to be known by some external mechanism. Having notified node n_s to upgrade its predecessor pointer to node *n*, node *n* copies over the portion of keys stored on to n_s and part of the interval $(n_p, n]$. Finally node n_p upgrades its successor pointer to *n*. This last operation is called *stabilization*, and is run periodically: a node asks its current successor to return its current predecessor, and if the identifier doesn't match, then another node has joined in between and has to be considered as the new successor. Also the finger table is initialized asking node *l* to discover all the *m* entries.

Finally, when a node quits the ring, it simply releases all its stored keys to its successor, and alerts it and its predecessor to link themselves.

Pastry [RD01]. Pastry shares with Chord the ring abstraction, but implements a radically different routing strategy.

Each node in the Pastry network is assigned a 128-bit node identifier (nodeId). The nodeId is used to indicate a node's position in a circular nodeId space, which ranges from *0* to *2^128 - 1*. The nodeId is assigned randomly when a node joins the system. It is assumed that nodeIds are generated such that the resulting set of nodeIds is uniformly distributed in the 128-bit nodeId space. For instance, nodeIds could be generated by computing a cryptographic hash of the node's public key or its IP address. As a result of this random assignment of nodeIds, with high probability, nodes with adjacent nodeIds are diverse in geography, ownership, jurisdiction, network attachment, etc.

For the purpose of routing, nodeIds and keys are thought of as a sequence of digits with base *2^b*, ranging in the same identifier space. Pastry routes messages to the node whose nodeId is numerically closest to the given key. This is accomplished as follows. In each routing step, a node normally forwards the message to a node whose nodeId shares with the key a prefix that is at least one digit (or *b* bits) longer than the prefix that the key shares with the present node's id. If no such node is known, the message is forwarded to a node whose nodeId shares

a prefix with the key as long as the current node, but is numerically closer to the key than the present node's id. To support this routing procedure, each node maintains the following routing state:

Routing table. A node's routing table *R* is organized into *(log_2^b N)* rows with *2^b-1* entries each. The *2^b-1* entries at row *n* of the routing table each refers to a node whose nodeId shares the present node's nodeId in the first *n* digits, but whose *n + 1*-th digit differs. Each entry in the routing table contains the IP address of one of potentially many nodes whose nodeId have the appropriate prefix; in practice, a node is chosen that is close to the present node, according to the proximity metric. This choice provides good locality properties. If no node is known with a suitable nodeId, then the routing table entry is left empty. The uniform distribution of nodeIds ensures an even population of the nodeId space; thus, on average, only *(log_2^b N)* rows are populated in the routing table.

Neighborhood set. The neighborhood set *M* contains the nodeIds and IP addresses of the nodes that are closest (according the proximity metric) to the local node. The neighborhood set is not normally used in routing messages; it is useful in maintaining locality properties.

Leaf set. The leaf set *L* is the set of nodes with identifiers numerically closest to the current node. A half of the identifiers in *L* are larger than the one of the present node, while the other half are smaller (it's an interval centered around the present identifier). The leaf set is used during the message routing.

In routing a given message, the node first checks to see if the key falls within the range of nodeIds covered by its leaf set. If so, the message is forwarded directly to the destination node, namely the node in the leaf set whose nodeId is closest to the message key (possibly the present node). If the key is not covered by the leaf set, then the routing table is used and the message is forwarded to a node that shares a common prefix with the key by at least one more digit that the present identifier. Sometimes, it is possible that the appropriate entry in the routing table is empty or the associated node is not reachable; in this case the message is forwarded to a node that shares a prefix with the key at least as long as the local node, and is numerically closer to the key than the present node's id. Such a node must be in the leaf set unless the message has already arrived at the node with numerically closest nodeId. And, unless a half of the nodes in the leaf set have failed

simultaneously, at least one of those nodes must be alive. This simple routing procedure always converges, because each step takes the message to a node that either (1) shares a longer prefix with the key than the local node, or (2) shares as long a prefix with, but is numerically closer to the key than the local node.

When a new node arrives, it needs to initialize its state tables, and then inform other nodes of its presence. We assume the new node knows initially about a nearby Pastry node *A*, according to the proximity metric, that is already part of the system. Such a node can be located automatically, for instance, using "expanding ring" IP multicast, or be obtained by the system administrator through outside channels. Let us assume the new node's nodeId is *X*. Node *X* then asks *A* to route a special "join" message with the key equal to *X*. Like any message, Pastry routes the join message to the existing node *Z* whose id is numerically closest to *X*. In response to receiving the "join" request, nodes *A*, *Z*, and all nodes encountered on the path connecting them, send their state tables to *X*. The new node *X* inspects this information and then initializes its own state table by:

- Taking the neighbohood set *M* from *A*.
- Taking the leaf set *L* from *Z*.
- Taking the *i*-th row of the routing table *R* from node *B_i* , which is the *i*-th node in the path from *A* to *Z* and shares a prefix of length *i* with *X*.

Finally, *X* informs any nodes that need to be aware of its arrival, transmitting a copy of its resulting state. This procedure ensures that *X* initializes its state with appropriate values, and that the state in all other affected nodes is updated.