Software infrastructures for ad-hoc networks oriented to difficult environments

WP1T2 version 3

DISI - Universita' di Genova
in cooperation with other partners in the project

September 2005

Abstract

A MANET is made by a set of mobile devices, PDAs or laptops, connected by wireless links and communicating without control structures. MANETs have all typical problems of wireless networks like bandwidth optimization and power saving, plus those due to frequent disconnections, topology changes, and multi hop routing.

The consequences impact on several layers in the ISO-OSI stack in order to solve typical problems like location, security, resource discovery and so on. The ISO-OSI layers need to cooperate and a clear encapsulation of functions at lower levels may be impossible or not convenient; certain data belonging to lower layers should in fact be visible up to application level.

This report summarizes what features should be taken into account at each software layer, in MANET environments.

Network layer

In MANETs we may use non-conventinal routing algorithms like content-based routing or geographic routing. We have not chosen a specific routing algorithm, but to support a flexible architecture which can provide different services acording to application requests.

Routing algorithms

This section describes routing types and actual implementations of routing algorithms on MANET.

Positioning

Relative or absolute positioning is a valuable information at several layers. We may have a GPS at each node, or we may derive positioning by a self positioning algorithm.

Middleware Functions

The most important middleware services include data sharing and accessing methods. To this aim, file systems in the "traditional" meaning may be inadequate. Shared data spaces may be as useful; this section analyzes both views.

Coordination Models

This section lists some recent proposals specifically studied for a MANET.

File Systems in Wireless Networks

Most existing filesystems designed for wireless networks have been designed for infrastructured networks, not for MANETs, we list them here.

File handling in MANET

To design a file system for MANET, we have several constraints like limited memory space, limited energy, frequent disconnections, data consistency. These problems require a strong commitment to replication, caching, resource discovery and security.

An ad-hoc network is made by a set of mobile nodes, PDAs or laptops, communicating by means of wireless links, without a central control structure. As described in the scenarios of WP1T1, a MANET may be used as a standalone network, or it may be connected to wired networks by special nodes acting as bridges or routers to the Internet, for example nodes with a cellular or satellite link.
In this type of network, all problems of wireless networks can be present, such as energy saving and bandwidth optimization; besides, there are also other problems typical of the ad-hoc environment, like multi-hop routing and frequent network topology changes.
The consequences of these problems not only affect the network layer, but also other levels in the ISO-OSI stack can be affected, to handle problems like security handling, mobile node positioning, resource discovery and so on.
For these reasons the classical level separation in the ISO-OSI stack can be violated because a need for cooperation among levels may arise, in order to share common information and to send signals when system stayus is changed, as shown in the following figure.

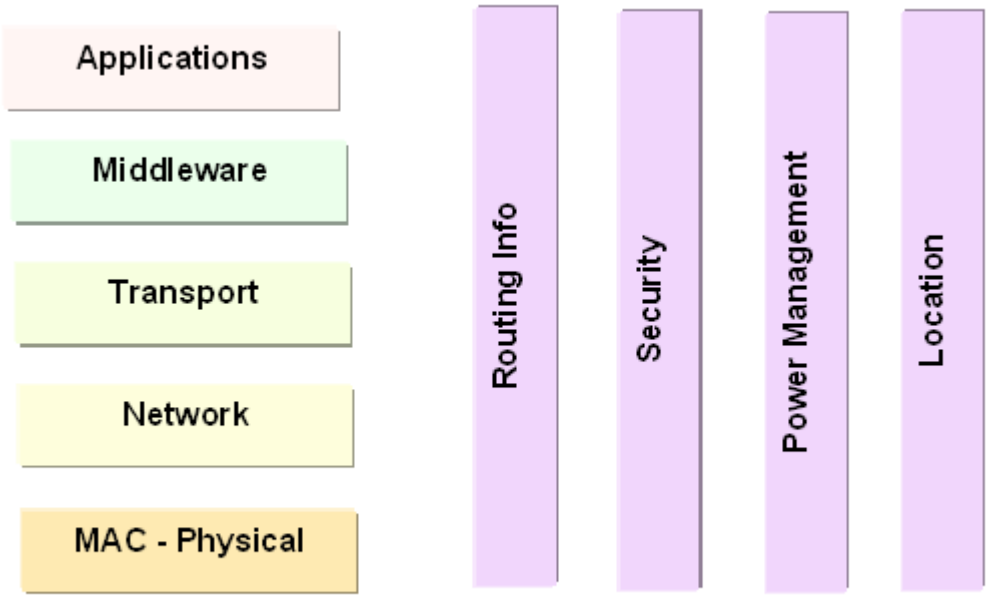

Figure 1. ISO-OSI stack and intra layer relationships

This report will describe the main features that software layers (from network layer to application layer) should contain, in order to match the corresponding problems to be solved in MANET programming.

Index

Network level functions

An analysis of existing literature has shown that the concept of routing in ad-hoc networks may be quite different from the concept of routing in traditional networks, where the latter means point-to-point communication between "well-known" node pairs. In fact, in ad hoc networks, different concepts for routing make sense, like *content-based routing* (where the receiver is isdentified by some sort of pattern-matching) or *geographical routing* (where the receiver is not uniquely identified, rather it is the set of nodes actually present at some geographical location), and they can be useful to application software at higher levels.

For these reasons, the project does not select a specific routing protocol; on the other hand, the choice of a flexible layered architecture has been made, where services offered at various levels can be flexible and adaptable in accordance with specific application needs.

In the following we shall highlight the possible routing strategies and algorithms implemented and available for MANETs.

Routing algorithms

In a MANET each node may act as a router, and it executes some Routing Protocol to forward packets to other nodes.

Since there is no standard at this level, the literature shows several possible algorithms, and each proposed algorithm has some specific objective and strategy, slightly different from others. Possible classifications are:

- **Proactive or reactive algorithms**. The main algorithms used on fixed networks are proactive, and periodically require exchanges of routing tables (or their updates). Routing algorithms for MANETs activate an information exchange when a topology change is detected. Reactive algorithms, on the other hand, build a path from sender to receiver only when such a path is needed, that is when such a path is not known, or when some time-out has elapsed, or when a previous send request has failed because a topology change has taken place. There are also hybrid algorithms (like the AODV algorithm) where no routing tables are actually exchanged, but periodical beacons are issued to test connectivity.
- **Structured or non-structured topology algorithms** from the point of view of logical network organization. Routing protocols can be cluster-based hierarchical, or flat. Cluster based protocols logically divide eache network into clusters, and each cluster has a leader node. Leader neods collectively keep all information on network topology. Flat protocols have no hierarchy and all nodes take part in message exchange, in symmetrical way. This classification of course allows for the definition of hybrid schemes as well.
- **Algorithms with or without node positioning information**. A node may be aware of positions of other nodes, e.g. because all nodes have a GPS device and such information can be actually shared; otherwise because by knowing previous movements a reasonable forecast of future movements can be assumed. Path finding algorithms may be optimized by knowing neighbours positions.
- **Algorithms with different metrics** Some algorithms try to minimize the number of hops to deliver a message from sender to receiver. Others look for paths providing the highest stability (to decrease the number of searches for new paths due to moving nodes), others try to reduce the energy overall spent along the routing path, or try to minimize the energy spent in each node's battery, to achieve a longer connection uptime. [Clem2002].

Considering implementation related issues, proactive protocols used in fixed networks require some differences in order to be implemented on a MANET because of specific problems. For example:

1. How to handle the queue of messages waiting for route discovery. On fixed networks, packets for a destination not included in some routing table entry are dropped and lost. In a

MANET, such packets should be kept in a waiting queue while the route discovery algorithm is activated. No Operating System mechanism is provided to solve such problem.

2. Entry timeouts. Routing table entries for a MANET must be associated with an expiration timer, which is not present nor needed in implementations for fixed networks.
3. Forwarding and routing separation. Some algorithms (like for example DSR), do not include periodic message exchange for advertisement or link status sensing. Thus routing functions become a part of the forwarding activity, which cause a considerable architectural confusion.
4. Cross layer information exchange among different ISO-OSI levels. Some applications (see File Systems) need such information exchange, in order to make available at the application level some information about packet routing and forwarding. Similarly, at network level some routing algorithms make use of information usually kept at physical or data link levels, like signal strength, link status sensing, geolocation input. These problems have a conceptual impact (modifications of the ISO-OSI model) as well as implementation impacts.

The third consideration above has lead to two main approaches for routing algorithms implementation:

- in-kernel approach, requiring kernel modifications
- user-space approach, where forwarding is implemented in the user space, thus forcing packets to cross all levels.

For these reasons only a part of proposed algorithms has actually been implemented and tested, in a simulated testbed or actually on a MANET.

Hereafter is a list of routing protocols available in a Linux ad hoc network. Some of them require a modification or extension to the Linux kernel, others are implemented in user space. Both approaches have disadvantages and advantages:

- the in-kernel approach can be more efficient, but requires updates at each new kernel version, and in the long run it may become too costly;
- the user-space approach is mor flexible, but incurs in higher system overhead.

This report does not discuss in details the various algorithms, leaving the interested reader with respective bibliographical links, we only highlight their main features. Certain protocols, like AODV, have been implemented with slightly different functions, and some implementations are aveilable from the Web.

DSR-Monarch

A FreeBSD implementation developed by Rice University (Monarch Project). It works entirely within the Linux kernel, modifying the IP stack, and has some limitations. For example, it does not work with TCP connections, so real-life applications such as web browsing cannot use it. It is available only for certain kernel versions, not for all of them.

AODV (Ad-Hoc On Demand Distance Vector) UCSB:

This is a user-spoace implementation, with additional modules for a Linux kernel which by making use of Netfilter allow the user level to access packets (normally at kernel level). If a path is not available in the user routing tables, the route discovery algorithm is started. Thus each packet crosses the user-kernel address space boundary twice, once for routing and once for forwarding. The implementation of this protocol is available for PCs and for PDAs (Zaurus and Ipaq).

MAD-HOC

It is an user-space AODV implementation. It monitors ARP (Address Resolution Protocol) packets to decide if a route discovery procedure must be started. It has some limitations: for example, since

ARP packets are generated at level 2, the network must be suitably configured. Other limitations are due to a short packet timeout.

AODV-UU
This implementation is similar to the above, it only has a few different protocol logic features.

Kernel AODV
This is a Linux in-kernel implementation made at NIST; it seems to suffer from some stability problems.

TORA (Temporally-Ordered Routing Algorithm)/IMEP
There is a Linux implementation of this algorithm, which has been implemented on top of IMEP (see below), however, the available information shows that at present it is not yet completely reliable, and it is based on Linux kernels of the series v2.2.x, which are not stable as well.

LUNAR:
LUNAR implements a routing discovery on-demand protocol, with broadcast and path reconfiguration typically every 3 seconds. LUNAR includes an automatic IP gatewaying system (so it does not enable manual packet forwarding as in AODV) and it supports IP unicast and broadcast. There is an in-kernel implementation for Linux kernel 2.4.x with NETLINK, TUN/TAP and ARPD support, and it is oriented to be used on relatively small MANETs (diametes of a few units).

OLSR (Optimized Link State Route):
OLSR (Optimized Link State Routing Protocol) is implemented in user-space so it does not require kernel modifications. It works in proactive table-driven way. Certain nodes are elected Multipoint relay (MPR) from their neighbours, so that they may exchange network topology information. Each MPR node announces to the network which nodes he is responsible for (since they elect him).
OLSR has been implemented for OS Linux with kernel 2.4.x by INRIA, and for OS Microsoft Windows 2000 and Pocket PC by Grupo de Investigación de Redes de Computadores at the University of Valencia.
Finally, there are also some meta-implementations, that is systems supporting the design of routing algorithms. Among them we include:

IMEP (Internet Manet Encapsulation Layer)
It is an encapsulation protocol [8] providing a framework for routing algorithms development, by means of features like abstarctions for interfaces, link status sensing and reliable broadcast.

Click Modular Router (MIT)
Click is a software architecture useful to build routers. Based on it some algorithms for MANET have been developed.

ASLib
It is a system oriented to use for MANET. At kernel level there are Routing Tables expanded with new entries, and it provides a new component to implement on-demand routing functions. Finally there is an API to handle new mechanisms in a routing daemon. AODV and DSR implementations have been based on it.

Positioning

Position knowledge (relative or absolute positioning) is an important information that can be handled at various levels.

The simplest positionig information is gathered by a GPS device connected to each mobile unit. Such information can be used at each level for its own purposes:

- at network level, certain routing algorithms use positioning information; among them, Location Aided Routing [Ko1998] and GeoTORA [Ko2000] which extends unicast TORA for geographic routing. A very useful operation is multicast to those nodes close to a given location, that is Geocasting, for instance with Position-Based Multicasting (PBM) [Mauv2003] which does not employ global structures.
- At middleware level, positioning may be used to handle a proxy cache, to keep geographically relevant information, for resource discovery and to optimize energy saving.

When nodes cannot access a GPS, there are some Self Positioning algorithms, for example [Kapc2001], which use measures like Signal Strenght, Angol of Arrival (AOA) and Time Difference of Arrival (TDOA) to estimate relative position with respect to other nodes. To apply these algorithms, such information should be collected at physical level, and should be made available up to the application level.

Middleware functions

Main services provided by middleware level include shared data access support and resource discovery/notification.
As we saw before with the routing concept, certain concepts can be defined in slightly different ways for ad hoc networks, and file systems are among them. Besides "traditional" file systems, content based and geographical models for file systems are emerging as well. Information exchange may be based on shared data space models, especially tailored with locality and network topology information. Hereafter we shall consider new coordination models based on shared data spaces, as well as the "classical" distributed file system approach.

Coordination models

A widely studied research problem for MANET is the coordination problem, as well as in any distributed application. Components must interact and coordinate their actions in any distributed environment, but in a MANET, this problem becomes especially important since the network (and the distributed application) structure is changing very fast: new nodes (thus new application components) may enter or exit the network, and network topology evolves as well. For this reason component communication patterns cannot be statically established when the application is being developed, rather they should be dynamically decided at run time. As a consequence, distributed application components cannot be statically bound (i.e. component *A* cannot invoke method *M* on component *B*: the network may not include the node hosting component *B* at a given instant). Then, components must be able to dynamically discover their current surrounding environment (*context-awareness*) and to interact and coordinate activities among themselves in a robust and flexible way.

One of the main problems to be solved in each application, especially from the software engineering perspective, is to provide components in a distributed application with a good support for their coordination. Effective tools to gather information about the environment (*context-awareness*), as well as flexible interaction mechanisms, should be provided to components, so that they can agree, plan and synchronize their own activities.
A widely studied solution to such a problem is to provide suitable tools within a middleware layer, containing coordination tools to be used by application components. In this way middleware can simplify some relevant aspects in distributed application complexity.

Unfortunately, from our point of view, most existing middleware architectures do not provide a viable coordination support for MANETs. For this reason, an extensive research on these topics appears extremely important.

Inadequacy of current approaches

Traditionally, distributed programming models and the corresponding middleware infrastructures can be classified into three main approaches:

- Message passing models [BelPR01, BelCS01],
- Shared data space models [FreHA99, CabLZ02],
- Event based models [Car02, CugFD01].

From the software engineering point of view, each approach provides a homogeneous set of functionalities, whatever the underlying implementation (e.g. centralized or distributed architectures).

Applications developed on top of message passing based models are usually designed as set of components that explicitly and directly communicate with one another. These models have been extended recently (e.g. with conversation among intelligent agents and advanced client server models [BelPR01, BelCS01]), but the highly dynamic environments of MANETs make them hardly suited to solve application requirements. In fact, mechanisms to explicitely identify entities are needed in order to allow applications to communicate with them, and middelware supporting these features are typically very complex and costly. Multicast or broadcast based methods do not appear well suited for the huge amount of generated traffic (consider for example the dramatic effects of Gnutella on Internet traffic). To this we must add a consideration about application components: in such models, they are situated inside a "conecptually empty space". The model by itself does not provide context information, and application components may be aware, and may interact, only with other components: no high-level environment abstraction is provided [Kle00]. Then, any component must explicitely become context-aware, asking for context information to specific local services, or to users: this requires more and more computation and communication, and increases development costs.

Models based on shared data spaces use physically shared memories so that application components may collect common information, interact and coordinate among themselves. These data structures may be hosted in some given centralized data space (such as, for example, a tuple space), as in JavaSpaces [FreHA99], or they may be completely distributed across the network, as in MARS [CabLZ02]. In such a case, component interactions are no longer rigorously coupled, since they are made possible by tuple space, used as efficient local and context storage. However notice that such context information may only represent a local context view, and may hardly be used for complex, globally coordinated tasks.

Publish-subscribe event based models [Car01, CugFD01], model an application by means of a set of components, which interact generating events and reacting to interesting events. An event-driven model promotes a very loose component coupling (interactions are asynchronous and anonymous, and no shared data space is assumed to exist), as well as a greater degree of *context-awareness* (components may be considered to be inserted inside an active environment which can inform them about what is happening). However, event-generated context information is typically at a very low level to be useful in sophisticated programming models: flexible application adaptation to environment changes, represented by events, implies a very dynamic and very complex design.

Synthesis of some novel approaches

From the above considerations, several proposals have been or are being studied to provide more sophisticated tools, supporting coordination of distributed application components. TO BE UPDATED with other references

Recent proposals address sophisticated models for components interaction in a MANET by means of shared data structures. Lime [PicMR01] and XMiddle [MasCE01] middleware provide an implementation of such an idea. Each device in the network has a private data structure: in Lime each device has a private tuple space, while in XMiddle each device has a tree, reperesenting an XML document. When two or more devices are connected inside a MANET, their private data structures are joined together. The resulting shared data structure enables component interaction. For instance, in Lime the shared tuple space, created by the meeting of two nodes, allows a component on one device to access some tuple included in the other device by another component. The strong point of these systems is that no connection to external infrastructure is required: they communicate with one another over the MANET.

EgoSpace [RomJ02] is a middleware especially developed to help in context information collection in a MANET. In this system each network node may state an "interest" in some context information, possibly selecting a geographic area where such information should be found (e.g. a component may be interested in fuel suppliers within 10 km radius from its current position).
Then EgoSpace takes in charge to build a distributed data structure involving all devices within the area of interest. This data structure is known to all devices involved in its distribution, and it serves two main purposes: it communicates the interest of the source node in some context information, and it creates a routing structure to collect the collected information to the requesting node.

Anthill [BabMM02] and SwarmLinda [MenT03] are two middleware, with close similarities, which support communication and coordination in peer-to-peer applications on dynamic networks, including MANETs. Both systems have been built based on swarm systems analogy [BonDT99]. In details, these systems are based on launching a large number of mobile components over the network. Along their movement across the network, these components deposit data structures, which in turn influence their further movements across the network. The underlying model of these systems is inspired by the way ants communicate with one another, that is leaving feromon traces in the environment. This system creates distributed data structures which directly connect the various nodes, and help their coordination.

The system TOTA (Tuples On The Air) [MamZ04] provides application components with an integrated toolset to support interactions with other components, as well as to represent context information about the outside environment where components are working in. This integrated toolset is implemented by distributed data structures (tuples) which application components may inject in the environment. These distributed tuples on one hand enable component interactions, since they may be used to convey messages. On the other hand, they may represent context information since they are persistent in the environment. In addition, TOTA's tuples are self-modifying, in dependency of environment dynamics (e.g. MANET reconfigurations), in order to keep the structure coherent. Components then take their decisions based on the local configuration of tuples present in the network.

File Systems in Wireless Networks

The following section describes problems and solutions for mobile and wireless environments, then those specific of MANETs.

Almost any application for a MANET makes use of a file system, specialized for the underlying networking structure. In fact due to the mobility and limitations of a WiFi connection, data access may not be always guaranteed.
In order to minimize such inconveniencies, we must employ those FileSystems which are capable of handling data replication, consistency checks, and read-write conflicts, in order to avoid failures or errors due to temporary disconnections.
Most file systems which allow some form of mobility support have been designed for infrastructured networks, that is they assume connections through Access Points rather than ad-hoc networking. They make provisions for disconnections and consistency checks for data stored on mobile nodes. Consider for example the following systems.
CODA is a distributed file system originating as an evolution from AFS2 (Andrew FileSystem), distributed for free, wich makes a distinction between server, which can be replicated, and client, which may be mobile and may communicate with servers by means of Remote Procedure Calls. A client must synchronize with a group of Available Replicated Servers in order to receive a file; then it may modify the file locally, and when the file is finally closed after modifications, it is transferred to all members in the replicated servers group. CODA is also capable of self-configurating in accordance with the available bandwidth.
Ficus is another replicated file system for intermittently connected mobile nodes. Unlike CODA, it does not make distinctions among replicas, and it enables synchronization among clients which have one replica. Its coherency management protocol is an optimistic one, based on One Copy Availability: modifications are made on the available replica, and then notified to other copies. In case of need a reconciliation protocol is activated.
Bayou is a client server system supporting database application on a mobile network. Servers propagate write operations to database copies, and when some nodes are disconnected, only weak consistency is achieved. An anti-entropy protocol is used, which ensures that all database copies are converging to the same state.
Bengal [Eken2001] is not a file system, rather it is a distributed and replicated database which supports mobile nodes. Here too, coherency with respect to disconnected clients is approached with an optimistic protocol, allowing reconciliation of modifications when connectivity is recovered.
However, these distributed file systems, as well as other, do not take into account some MANET features, especially

- the need of routing a file through several nodes before reaching the actual destination, and
- the discovery of where a certain file is stored, if no clear distinction between servers and clients is possible (useful if no connection to wired network is available)

Notice also that mechanisms like the Remote Procedure Call used by CODA are not suited for MANET communication.
The next section then summarizes File System requirements for a MANET, and how these requirements have been met so far.

File handling in MANET

Considering the scenarios proposed in the WP1D1 report, we can view a MANET under two perspectives.

- The first is the simplest one, as in Scenario A: the MANET is isolated from the wired network, and it is symmetrical, that is, no node is privileged with respect to the others. For load sharing, it is meaningful to distribute files composing the File System among many nodes, possibly replicating them for increasing availability. Each file has some primary server which keeps its copy and makes it available upon requests to other nodes.
- The second is Scenario C, where the MANET is asymmetrical: there are some privileged nodes (*Base Stations,* either mobile or fixed ones) representing gateways to the fixed network, or at least data servers with unlimited storage and power. Files needed by other nodes in the MANET shall be fetched from these data servers.

In the latter case the MANET is not considered as separated from the infrastructured network, rather it is considered an extension to it, hence it should be compelmentary to the infrastructured network. Some problems may be solved in both scenarios, others will require different solutions. For example:

1. Mobile clients may have memory limitations, e.g. PDAs, and this requires continuous interactions with the data servers to fetch required information, since it is not possible to distribute all useful data and store them in the mobile node once and for all.
2. Power is limited as well. Each time a node requests a file to a server, the file must be moved along all the nodes in the path connecting the client to the server: this takes energy, bandwidth and latency.
3. Since it is possible to disconnect, some file may be unavailable (at least temporarily).
4. When files can be updated, there is a data consistency problem. File consistency is usually achieved by imposing file locks, a technique which cannot be used on a MANET

Replication and Caching Proxy

The above point 3 requires some form of replication in order to increase file availability; from the above point 2, it follows that locating a replica on some node close to the client would decrease the overall energy requirements for file transfer.

The solution to the above problems requires a well known concept on fixed networks, that is cooperative caching, which allows file sharing and coordination of caching among many nodes. The implementation of the caching proxy concept on MANET, with obvious differences with respect to fixed networks, allows to solve both problems above, together with the transparent distribution of file replicas to local cache on various nodes.

In our scenarios of interest, that is to provide tools to support disaster recovery intervention of civil protection groups, we can see that geographically close people tend to need and share common information, especially those dependent on locality like maps or data on local buildings and activities. In this case certain requests could be foreseen, and information relevant to actual positioning could be accordingly cached on nodes in such area, acting as caching proxies for their neigbours.

Resource discovery

File access from mobile users in a MANET may be compared with Peer-to-Peer file sharing, with some similarities. In fact, in a MANET we have already seen some reasons against centralizing files on one or few servers, which are responsible for file and replicas placement; in any case, there is no longer a clear separation between clients and servers, in order to reduce file unavailability in case of disconnections.

P2P systems consist in a set of nodes, which communicate with one another and keep community-relevant information in decentralized way. In this case structural information, information about available services, as well as the information itself (i.e. files) are decentralized, and can be accessed by means of cooperative protocols involving some or all nodes. These protocols depend on peers connections at the application level, and represent an overlay network on top of the physical level, allowing browsing and search over all system information.

Unliked wired networks, in a MANET overlay network topology varies over time because of node mobility, introducing a significant overhead to maintain such connections.

Several protocols have been proposed for data replication and resource discovery, but only a few of them have actually been deployed, as we will describe in the following.

In ***[Sail2003]*** we find a strategy for cache handling and accessing on MANET nodes, where file search is performed much like what UMTS does to extend the communication rank of an infrastructure. It is based on the ZRP (Zone Routing Protocol) from which an interaction is required (information cross over from the network level to the application level). File is first searched in the local cache, then on nodes belonging to the same Zone, then on the file owner.

ORION [Klem2003], Optimized Routing Independent Overlay Network, is a file search and transfer algorithm, which reconstructs route discovery, as performed at the network level, at the application level. File query messages are broadcasted to neighbouring nodes; each proxy makes a search based on routing tables, and reconstructs a local file allocation table in accordance with answers from contacted nodes. Thus it is possible to transfer a file from the closest cache to the requesting node.

Replica consistency and security

Both these concerns are afforded by FS designed for wireless networks, such as those mentioned above. Replica consistency becomes a problem if updates are allowed when disconnected (data synchronization) and implies solving possible conflicts among updates. Locking files in a MANET environment is not practical, so the only possible approach is the optimistic one: we must be able to realize if a conflict has occurred, and we need a reconciliation algorithm to be applied afterwards, based on log updates or other techniques.

Security is an important issue in MANET environments as well, in order to ensure end-to-end confidentiality and user data integrity. Symmetric key criptography is a widely used method, in order to ensure data confidentiality during transfer with a secret key. Data integrity can be obtained with a safe hash function (one way function).

A safe data distribution can also be achieved by use of access privileges, and a dependable data storage can be implemented with fragmentation and dispersion techniques based on erasure code, coupled with cryptography [Ches2003].

A proposal for a File System especially designed for MANETs is called ***AdHocFS*** [Boul2993], developed at INRIA and organized around a "traditional" distributed hierarchical FS, keeping into account data consistency and security in data transfers. This File System has a mixed strategy for data consistency: at the network level optimistic replication is used, in order to allow file updates also when disconnected. However, it uses also the peers group concept, and inside a group file replicas can be strongly synchronized to prevent conflicts. The system has been implemented on top of Extended 2 FS (Ext2), using the encryption algorithm Blowfish, and it can be used on Linux laptops. It is not freely downloadable from the web.

References

Routing protocols

[DSR] http://www.monarch.cs.cmu.edu/dsr-impl.html

[AODV-UCSB] http://moment.cs.ucsb.edu/AODV/aodv.html

[MAD-HOC] http://mad-hoc.flyinglinux.net/

[AODV-UU] http://www.docs.uu.se/~henrikl/aodv/

[Kernel AODV] http://w3.antd.nist.gov/wctg/aodv_kernel/

[TORA/IMEP] http://www.isr.umd.edu/CSHCN/research/index.html

[LUNAR] http://www.docs.uu.se/docs/research/projects/selnet/lunar

[OLSR] http://hipercom.inria.fr/olsr Linux version;
http://reptar.grc.upv.es/~calafate/olsr/olsr.htm Windows version.

[ASLib] Ad Hoc Support Library http://aslib.sourceforge.net

[Click] E. Kohler, R. Morris, B. Chen, J. Jannotti, .F. Kaashoek, "The Click modular router", *ACL Transaction on Computer Systems*, 18(3), pp.263-297, 2000.

[IMEP] S. Corson, S. Papademetriou, P. Papadopulos, V. Park, A. Qayyum, "An Internet MANET Encapsulation Protocol (IMEP) Speification", *IETF Draft*, 1999.

[Clem2002] A. Clematis, D. D'Agostino, V. Gianuzzi, "A Local Decision Algorithm for Maximum Lifetime in Ad Hoc Networks", *Proc. EUROPAR 2002*, Paderborn, Sept. 2002

Coordination methods

[BabMM02] O. Babaoglu, H. Meling, A. Montresor, "Anthill: A Framework for the Development of Agent-Based Peer-to-Peer Systems", *Proc. of the 22nd International Conference on Distributed Computing Systems (ICDCS '02)*, Vienna, Austria, July 2002.

[BelCS01] P. Bellavista, A. Corradi, C. Stefanelli. "Middleware Service for Interoperability in Open Mobile Agent System",. *Microprocessors and Microsystems*, 25(2):75-83, May 2001.

[BelPR01] F. Bellifemine, A. Poggi, G. Rimassa. "JADE - A FIPA2000 Compliant Agent Development Environment", *Proc. 5th Conference on Autonomous Agents*, Montreal (CA), May 2001.

[BonDT99]E. Bonabeau, M. Dorigo, G. Theraulaz, "Swarm Intelligence", Oxford University Press, 1999.

[CabLZ02] G. Cabri, L. Leonardi, F. Zambonelli. "Engineering Mobile Agent Applications via Context-Dependent Coordination,. *IEEE Trans. on Software Engineering*, 28(11), Nov. 2002.

[Car02]A. Carzaniga, D. Rosenblum, A. Wolf, "Design and Evaluation of a Wide-Area Event Notification Service", *ACM Transaction on Computer System*, 19(3):332-383.

[CugFD01] G. Cugola, A. Fuggetta, E. De Nitto. The JEDI Event-based Infrastructure and its Application to the Development of the OPSS WFMS. *IEEE Trans. on Software Engineering*, 27(9): 827-850, Sept. 2001.

[FreHA99] E. Freeman, S. Hupfer, K. Arnold. "JavaSpaces Principles, Patterns, and Practice", Addison-Wesley, 1999.

[Kle00] L. Kleinrock. "On Some Principles in Nomadic Computing and Multi-Access Communications*". IEEE Communication Magazines*, 38(7):46:50, July 2000.

[MamZ04]M.Mamei, F.Zambonelli, "Programming Pervasive and Mobile Computing Applications with the TOTA Middleware", *Proc. IEEE Percom*, Orlando (FL), USA, March, 2004.

[MasCE01] C. Mascolo, L. Capra, W. Emmerich, "An XML based Middleware for Peer-to-Peer Computing", *1st IEEE International Conference of Peer-to-Peer Computing*, Linkoping (S), Aug. 2001.

[MenT03]R. Menezes, R. Tolksdorf, "A New Approach to Scalable Linda-systems Based on Swarms", *ACM SAC 2003,* Orlando, Florida, USA, March 2003.

[PicMR01] G. P. Picco, A. L. Murphy, G. C. Roman, "LIME: a Middleware for Logical and Physical Mobility", *Proc. of the 21st International Conference on Distributed Computing Systems*, IEEE CS Press, July 2001.

[RomJ02] G. C. Roman, C. Julien, Q. Huang, "Network Abstractions for Context-Aware Mobile Computing", *ICSE '02,* Orlando (FL), ACM Press, May 2002.

File Systems

[Boul2993] M. Boulkenafed, V. Issarny, "AdHocFS: Sharing Files in WLANs", *Proc. 2nd Int. Symp. On Network Computing and Applications*, April 2003.

[CODA] http://www.coda.cs.cmu.edu Home Page of the CODA filesystem.

[Eken2001] T. Ekenstam, C.. Matheny, P.L. Reiher,G.J. Popek, "The Bengal database replication system", *Distributed and Parallel Databases*, 9(3), pp.187-210, 2001.

[Hara2001] T. Hara, "Effective replica allocation in ad hoc networks for improving data accessibility," *Proc. IEEE INFOCOM 2001*, pp. 1568—1576.

[Klem2003] A. Klemm, C. Lindemann, O.P. Waldhorst, "A Special-Purpose Peer-to-Peer File Sharing System for Mobile Ad Hoc Networks", *Proc. VTC2003*.

[Nugg2002] P. Nuggehalli, V. Srinivasan, C.-F. Chiasserini, "Energy-Efficient Caching Strategies in Ad Hoc Wireless Networks ", *Proc. 39th IEEE A Design Automation Conference (DAC'02)*, Invited paper, New Oleans, LU, USA, June 2002.

[Sail2003] F. Sailhan, V. Issarny, "Cooperative Caching in ad Hoc Networks", *Proc. 4th Int. Conf. On Mobile Data Management*, 2003, pp.13-28.

[Ches2003] S. Chessa, P. Maestrini, “Dependable and Secure Data Storage and Retrieval in Mobile, Wireless Networks”, *Proc. IEEE DSN 2003*, International Conference on Dependable System and Networks, San Francisco, 2003.

Positioning

[Ko1998] Y-.B. Ko, N.H. Vaidya, “Location Aided Routing (LAR) in mobile ad-hoc networks”, *Proc. MOBICOM*, 1998.

[Mauv2003] M. Mauve et al., “Position-Based Multicast Routing for Mobile Ad Hoc Networks”, *Proc. MobiCom*, 2003.

[Ko2000] Y-B. Ko, N.H. Vaidya, “GeoTORA:a Protocol for Geocasting in Mobile Ad Hoc Networks”, *Proc. 8th IEEE Int. Conf Network Protocols*, 2000.

[Kapc2001] S. Kapcun, M. Hamdi, J-P. Hubaux, “GPS-Free Positioning in Mobile ad-hoc Networks”, *Proc. HICSS*, 2001