# A Practical Framework to Enable the Self-Management of Mobile Ad-Hoc Networks

Apostolos Malatras and George Pavlou
Centre for Communications Systems Research, Department of Electronic Engineering,
University of Surrey, UK
a.malatras@surrey.ac.uk, g.pavlou@surrey.ac.uk

**Abstract— Mobile ad hoc networks (MANETs) form the underlying networking paradigm upon which pervasive and ubiquitous environments are founded. Combined with the growing need for mobility and flexibility in network infrastructures, the prominence and endurance of MANETs as a networking trend becomes evident. The main characteristics of MANETs include energy and bandwidth constraints, dynamic topologies and platform heterogeneity. Traditional network management principles and approaches are not applicable to MANETs, while on the other hand self-management principles are eminently suitable given the dynamic nature of MANETs. In this paper we propose, implement and evaluate a framework to enable self-management of MANETs built on an adaptive organizational model. We present the design of our proposed framework and we elaborate on the context-driven self-management cycle. We also present its evaluation through practical experiments using management case-studies.**

***Index Terms*—context awareness, mobile ad hoc networks, self organization**

## I. Introduction

THE proliferation of mobile ad hoc networking solutions experienced in the last few years and the high rates of user adoption of wireless technologies leads us to consider that there will be a paradigm shift from traditional, infrastructure-based networking towards wireless mobile, operator–free, infrastructure-less networking, with MANETs playing a key [4]. MANETs together with other emerging networking technologies, such as sensor networks, will constitute the foundations for future pervasive applications. The major strengths of this technology lie in the fact that it is easy to be deployed at a relatively low cost, while allowing for user creativity through the lack of central, authoritative management and control [7].

MANETs undoubtedly are not a solution for every networking problem of the emerging pervasive realm. Noteworthy drawbacks include their highly dynamic topology, since a node participating in a MANET is potentially mobile. These constant topological variations will eventually lead to a continuous state of network instability, which in turn can deteriorate the performance of services and applications on these networks. It becomes evident that to fully benefit from the many potentials of ad hoc networking one needs to cater for a wide variety of requirements and provision for reliable, secure and efficient management [4], [7].

There is an obvious need for frameworks that can support the self-management of MANETs according to predefined goals or policies. We assert that such a highly dynamic environment can potentially benefit from context information that will drive its self-management, resulting in a degree of autonomy. The enabling technologies of autonomy and self-management also include policy-based management and programmability. Policy-based management provides the means to infer management decisions in a flexible and dynamic manner by matching contextual information to predefined rules. Programmability is also beneficial to enforce the required configuration changes on the system. This closed-loop adaptive management can thus lead to self-configuration, self-optimization, and hence a degree of autonomy.

This paper addresses the design of a context-aware policy-based framework to achieve this, focusing mainly on its design and implementation aspects. The rest of the paper is organized as follows. After this brief introduction, related work is reviewed in section 2, while section 3 lays the foundations of our proposed organizational model. The design of our practical framework to enable self-management of MANETs is the focus of section 4. Implementation issues and related software metrics are presented in section 5 with indicative evaluation results presented in section 6. Finally, section 7 concludes the paper.

## II. Related Work

Autonomic computing refers to the self-managed operation of computing systems and networks, without the need for human administrators but with high-level objectives dictating the system's functionality. The IBM autonomic computing blueprint [1] defines four distinct concepts behind autonomy, namely self-configuration, self-optimization, self-healing and self-protection [2]. The building block of all autonomic

This work was supported in part by the EU EMANICS Network of Excellence project (IST-026854).

solutions is an autonomic element. This refers to the collection of one or more managed elements that are handled by an autonomic manager. The latter monitors the state of the elements, analyzes it and acting upon high-level objectives (typically defined as policies) imposes the execution of configuration changes on the managed elements. This process is repetitive [2], [3].

Most autonomic computing platforms are targeted to systems with sufficient resources that are relatively stable [1], [5]. The application of autonomic principles to MANETs has not been extensively researched. In [9] we presented our initial approach and results on self-configuring and optimizing MANETs. In [10] a policy-based network management system for MANETs is proposed but the hierarchical approach adopted assumes the existence of several "thick" nodes in the network, which may not always be the case. The work for the Terminodes project is considered as fundamental for the self-organization of MANETs [6]. It is though mostly targeted to organization and cooperation issues, while it fails to address practical aspects.

The exploitation of context information in network management has been addressed before given that the potential benefits can be tangible. Relevant adaptive systems can be deployed, interacting with the surrounding environment and functioning according to monitored conditions [11], [12]. The main drawback of all these approaches is the static evaluation of context against predefined rules. The use of dynamic context monitoring in conjunction with network policies to achieve a degree of dynamicity at a higher conceptual level has not been considered in the past and this is one of the particular innovative aspects of our approach.

## III. MANET ORGANIZATION

A key consideration in a MANET is the organizational model to be deployed. In order to cope potentially with large scale, the most common practice is to organize the MANET into clusters, each managed by an elected local leader or cluster head (CH). Assuming such a hierarchical approach for the purpose of management, CHs then cooperate and either elect a global leader or network head (NH) or a set of CHs that collectively undertake the role of NH. The NH takes key management decisions, such as triggering and coordinating the clustering process and its maintenance. A diametrically different approach is a fully distributed one, in which all the nodes are deemed as equal and determine collectively any management decisions to be taken.

We assert that a purely hierarchical or a purely distributed approach in respect to the organizational model of a MANET is not suitable for such a network. The former is too cumbersome and rigid and does not allow for the flexibility required by organizational models applicable to MANETs while it also suffers from a single point of failure threats. The latter considers all the nodes as having "equal rights" and determining collectively any management decisions to be taken. This approach requires more complex cooperation and coordination protocols and does not scale well for a large number of nodes.

These observations guide our design choice of introducing a hybrid approach, namely a distributed and hierarchical organizational model for MANETs to exploit the benefits of each of those approaches. The hierarchical approach scales well for large networks by limiting interactions within a cluster or among cluster heads. It also allows operation in a controlled fashion. The distributed organizational model allows for the flexibility required in MANETs and is adaptive to the distributed nature of ad hoc networks.

The proposed organizational model adopts a 3-tier architecture with nodes being assigned three roles, namely the Cluster Node (CN), the Cluster Head (CH) and the Manager Node (MN). Nodes that have the MN role encapsulate the CH role as well. Nodes can assume these roles in a dynamic manner. Our proposed organizational model has been presented in detail in [8] and is depicted in Figure 1. The particular example refers to a hyper-cluster consisted of two MNs and one CH managing collectively the MANET. We will describe the components that comprise each node's functionality in the next section.

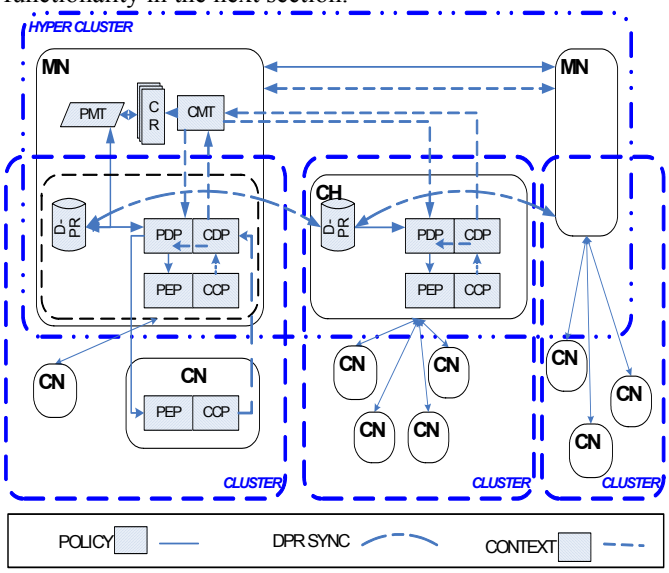


Fig. 1. Organizational model and node roles

The role notion denotes the functionality and responsibilities of each node in the proposed organizational model. To avoid the rigid view of pure hierarchical approaches, we introduce the hyper-cluster entity referring to the set of mobile nodes that are assigned the MN and CH roles. Effectively, the CHs together with the MNs form the hyper-cluster and collectively manage the MANET. Another differentiation from the traditional hierarchical model is the fact that our model allows for more than one Manager Nodes (MNs) in the MANET. The idea behind the multi-manager paradigm lies in the nature of ad hoc networks and the purpose of their formation. Having more than one manager gives the flexibility to form networks between distinct trusted

administrative authorities. This is performed without any MN being forced to forfeit its management privileges. Instead managers cooperatively guide the overall network's behavior. The multi-manager paradigm and the hyper-cluster formation offer a balance between the strictness of hierarchical models and the fully-fledged freedom of distributed ones. At the same time our model embraces both as it can be deployed as either of these.

## IV. Self-management of MANETs

To achieve any form of network management, let alone self-management, it is essential to establish a concrete organizational model that will cater for and correspond to the requirements set by the nature of the particular networking paradigm, i.e. MANET in this case. The design of a generic network management framework will be based on this model.

Self-management necessitates awareness regarding the environment, implying the need for context-awareness and management of this context information. Context management refers to monitoring the environment and distributing the collected information so that all the network nodes – especially in a MANET where centralization is non-existent – to have a uniform understanding of the current context conditions. It is also evident that the self-management framework must build on a particular management scheme. Redundancy to achieve reliability, synchronization to gain consistency and uniformity, policies to benefit from flexibility and generic applicability, are amongst the main aspects that will be incorporated in the proposed management scheme. It should also conform to the principles of the proposed organizational model. Network management implies the constant re-configuration of network nodes according to higher-level management decisions. Coming to self-management, dynamic configuration is of paramount importance, since dynamic conditions drive the need for configuration enforcement and in turn feed the monitoring of the environment to identify similar needs. We therefore present a software plugin-based approach for MANET configuration, building on network programmability principles.

We put forward a generic framework to achieve MANET self-management that operates in the following fashion. Context awareness achieved through efficient context modeling and monitoring is the foundation of our framework. It feeds the management scheme that is built on the requirements of a MANET-targeted organizational model. The management scheme based on this dynamic context information enforces through an appropriate platform configuration changes in the MANET nodes so as to conform to the high-level management decisions. These decisions may in turn result in new context information being generated and thus new management decisions need to be taken. Consequently a closed loop self-management cycle is generated, which can lead to self-optimization of MANETs, subject to the appropriate network policies having been introduced beforehand.

### A. Context-modeling

Network self-management relies on accurate information being collected from the environment, in order to identify conditions that could trigger the need for management changes and subsequent actions being taken. Raw data as collected from a variety of sensors are modeled with the use of a generic context model, in order for high-level context information to be deduced and handled by our system. The abundance of sensors available to a device could lead to an abundance of collected information, storage of which may be non economical given the memory requirements of a mobile device. The administrator introduces the context models that will be used to support management decisions and through these models the sensors that will be utilized are identified. This information is stored in the Context Repository component that can be found at every node of the MANET and stores diverse types of context, according to the role of each node (CN, CH or MN).

### B. Management scheme

Policy-based network management (PBNM) principles applied to the requirements of the MANET realm constitute the foundation of our proposed management scheme. It directly interacts with the organizational model to specify node assignment to PBNM roles and the context management framework in order to monitor conditions that trigger the various policies. Administrators introduce the policies that guide the self-management of MANETs through the initialization of the Manager Node modules. When the conditions of the policies are met then actions need to be enforced. The management scheme then interacts with the dynamic configuration component of our framework and employs the corresponding changes to the appropriate nodes.

Traditional PBNM architectures are comprised of the Policy Management Tool (PMT), which introduces policies into the system, the Policy Repository (PR) where these policies are stored, the Policy Decision Point (PDP) that evaluates at runtime policy conditions and decides upon their activation and the Policy Enforcement Point (PEP) that enforces the policy decisions on the nodes. Bearing in mind the MANET requirements and adhering to the aforementioned organizational model we propose adapting the PBNM paradigm to serve our needs. In particular, every node can support full PBNM functionality but according to its role certain functionality remains dormant. CNs have only the PEP entity installed, CHs have PDP functionality and MNs have PMT responsibilities. Every level of the hierarchy encompasses the functionality of the lower levels. Our management scheme supports multi-manager operation, which though raises issues of synchronization and uniform decision

making.

In order to tackle the aforementioned deficiencies we propose the DPR (Distributed Policy Repository) component. DPR is an enhanced version of the Policy Repository and consists of repository replicas distributed among hyper-cluster's nodes. Each manager node (MN) or cluster head (CH) accommodates a DPR replica of the repository and serves as access points for repository requests within their cluster, balancing this way processing load and traffic in the network.

### C. Dynamic configuration

Once management decisions have been inferred, there exists the need to carry out the necessary configuration changes to the mobile nodes so as to enforce these decisions. This occurs through a dynamic, programmable configuration platform. It receives input from the management scheme by means of high-level management decisions and based on rules set out by the administrator upon initialization, translates them into low-level configuration changes. Dynamic configuration is performed with the use of software plugins that bear the necessary configuration functionality and are distributed among the mobile nodes. These, in their simplest form are scripts that are executed in the configured device and change local variables; this avoids having to perform these series of changes remotely which could be difficult in a flaky environment such as a MANET. Uniformity and synchronization are essential prerequisites for the success of the dynamic configuration of nodes. The employment of new configuration in the MANET may result in new context being generated and thus an update to the Context Repository to be performed.

### D. Context management

The context management component cooperates and works in parallel with the policy-based scheme for MANETs that was previously introduced. Context information is gathered locally at every MANET node and after basic processing it is passed to the corresponding CH that is responsible for its aggregation and processing to higher level contexts. Cluster-wide decisions based on this context can be imposed by the CHs, provided that certain conditions as specified by policies are met. At regular intervals, aggregated context from CHs is passed to the MNs in order to establish if MANET-wide configuration changes are necessary. Based on our organizational model, we identify 3 main entities that constitute our context management framework, namely the Context Collection Point (CCP), Context Decision Point (CDP) and Context Management Tool (CMT).

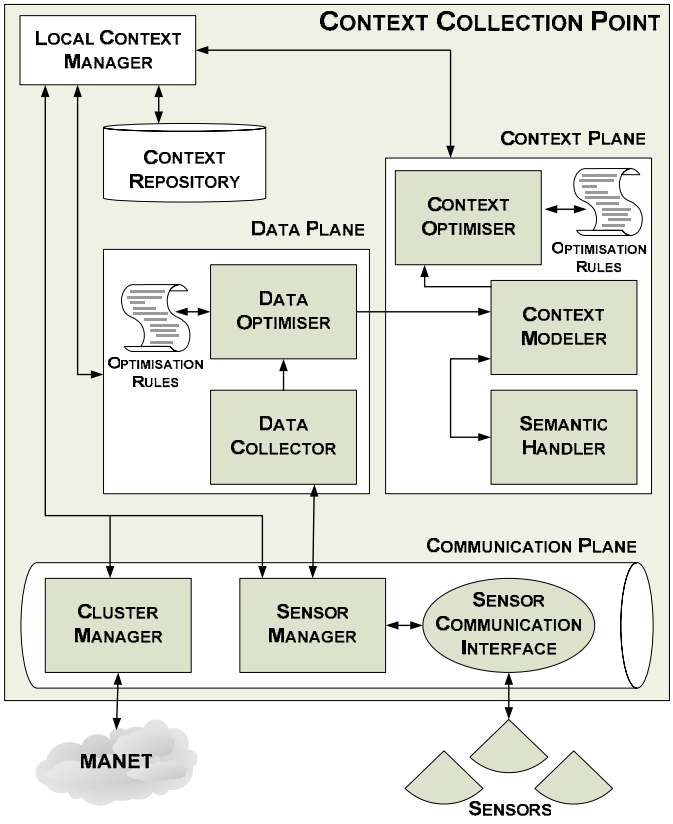


Fig. 2. Context Collection Point design

The CCP is deployed on every MANET node and is responsible for communicating with available sensors e.g. GPS, storage media, battery. The Sensor Manager interacts with the Sensor Communication Interfaces and presents the data and events collected from the sensor in a uniform way to the CCP to balance sensor diversity. The main entity of the CCP is the Local Context Manager, which is responsible for managing context information locally and for the communication with other nodes of the MANET. The Cluster Manager is the entity that performs all the activities related to the hyper-cluster formation and maintenance. The Data Collector gathers the diversely formatted data as received from the Sensor Manager and passes them to the Data Optimizer. Optimization rules guide the operation of the Data Optimizer whose responsibilities include pruning the collected set of data from invalid values and transforming the data to appropriate formats. The data is then passed to the Context Modeler that converts it to context using a generic model. The Semantic Handler feeds the Context Modeler with information regarding the type of data to be converted and the way this should be performed according to predefined context inference rules. Useful context is then passed to the Context Optimizer that based on Optimization Rules prunes the collected context and limits its size. Finally context information is passed to the Local Context Manager that stores it in the Context Repository (Figure 2).

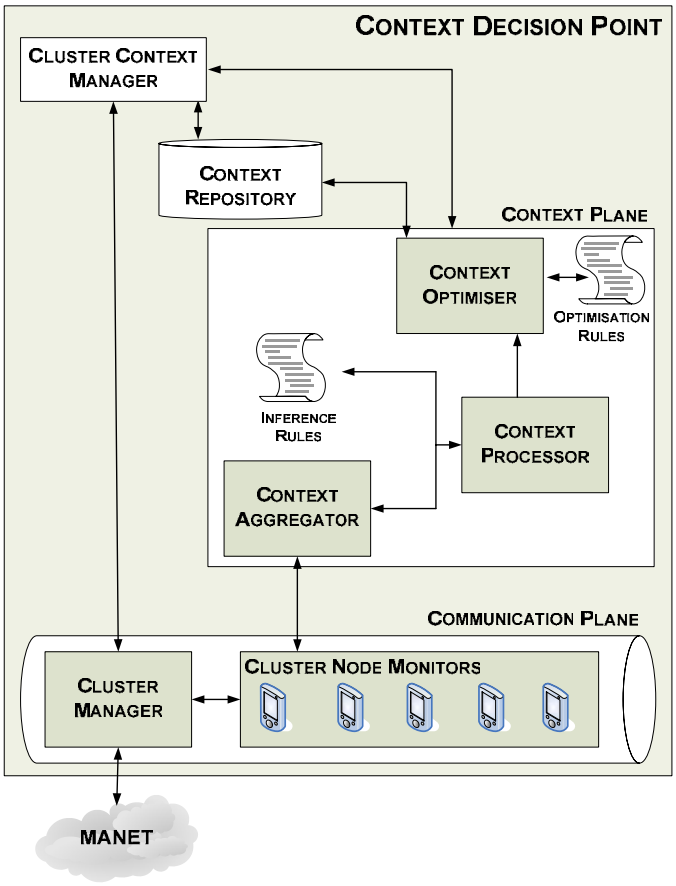


Fig. 3. Context Decision Point design

The Cluster Context Manager is the main entity of the Context Decision Point (CDP) installed on every CH. It is responsible for monitoring and interacting with the CCP modules of the nodes that are associated with this CH and also with the corresponding MN. The Cluster Manager, as with the CCP, is responsible for monitoring and informing the Cluster Context Manager regarding changes in the clustering process. The CDP has a series of Cluster Node Monitors that are responsible for collecting the context of the CNs associated to the CH. The Cluster Node Monitors periodically passes the collected context to the Context Aggregator. The Context Aggregator after having gathered the context from all managed CCPs produces average values to reduce the amount of data available to the CDP. Predefined, hard-coded Inference Rules combined with the aggregated context are used by the Context Processor to deduce higher-level contexts that have cluster-wide applicability. For example, mobility patterns from CNs collected by the CH can yield a cluster-wide view of the volatility of the whole cluster. Context optimizing occurs in the same fashion as in the CCP. The context collected at the cluster level can be used for cluster-wide adaptation when certain conditions are met. To accommodate this, the CDP communicates with the PDP also located at the CH and evaluates context against the monitored objects specified at the DPR to establish the need for cluster-wide configuration changes (Figure 3).

The CMT runs in MNs and allows the manager to trigger a management decision either directly or indirectly by modifying policies through the PMT, hence the CMT/PMT analogy and relationship. The main entity of the CMT is the CMT Manager whose responsibilities include communicating with the CMTs of other MNs and exchanging information regarding the context of the CHs each manages. This way ensures that all MNs have a uniform understanding of the context of the whole MANET in a distributed and efficient manner. The CMT Manager also interacts with the PMT and the PDP available at the MN in order to establish the need for MANET-wide configuration changes, by matching monitored context against monitored objects as specified in the policies stored in the distributed policy repository. The Cluster Manager keeps track of the clustering process and notifies the CMT Manager for any changes, while retaining CDP Monitors for every CH it manages. At the same time it retains CMT Monitors for other MNs, if any. The CDP Monitors receive context from CDPs and the CMT Monitors exchange MN-wide context. The CMT functionality apart from that is essentially equivalent to the CDP one (Figure 4).

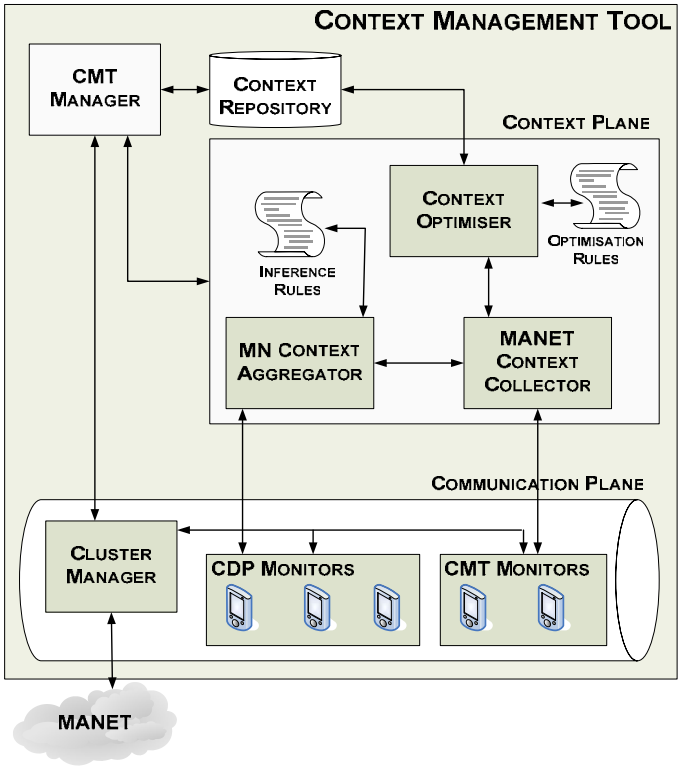


Fig. 4. Context Management Tool design

## V. IMPLEMENTATION

To test the platform's performance and efficiency and also examine its operation in a real environment, we deployed it in our experimental MANET testbed that comprises 3 laptops and 4 Personal Digital Assistants (PDAs). The testbed is a 6-hop MANET and is considered as a relatively reliable environment so that results can be extrapolated (Table 1).

TABLE 1
EXPERIMENTAL TESTBED CONFIGURATION

| Platform | Attribute | Description |
|---|---|---|
| PDA | Processor | 400 MHz Intel XScale |
| | Memory | 48 MB ROM, 128 MB RAM |
| | OS | Familiar Linux 2.4.19 |
| | Wireless NIC | IEEE 802.11b |
| Laptop | Processor | 1,7 GHz Intel Centrino |
| | Memory | 512 MB RAM |
| | OS | Debian Linux 2.6.3 |
| | Wireless NIC | IEEE 802.11b |

The most time consuming activity in setting up the testbed consisted of installing Linux-based operating systems in all the equipment. This was necessary to accommodate certain operating requirements that were expected from our framework, such as the ability to interact with lower-level protocols i.e. routing, that are inaccessible through Windows-based systems. While this process is simplified as far as laptops were concerned, the same did not stand for the PDAs. To date these equipments do not have a standardized version of Linux for their configuration. The only available solution is an extremely useful, open-source effort, namely Familiar Linux [16]. We were forced to implement certain changes to support proper operation of our framework, mostly in terms of allowing Java applications to be executed on the PDAs. These were in the form of certain kernel re-compilations. Extensive testing was performed before the actual deployment of our platform on the testbed to examine its proper operation under real ad hoc network conditions. Upon that we set forward to implement our platform in a manner that caters for the limited resources of such wireless devices, but in parallel does not make performance or operational compromises.

The platform is implemented using the Java 2 Micro Edition (J2ME) [15]. This version requires a much smaller memory footprint than the standard or enterprise edition, while at the same time it is optimized for the processing power and I/O capabilities of small mobile devices. We also used the Connected Device Configuration (CDC) framework instead of the limited one (CLDC), as the latter lacks support for required advanced operations. We chose to use Java because of its ubiquity, platform independence and the fact that it integrates naturally with Java-based plugins for node configuration. Our platform though caters also for C/C++-based plugins. The use of Java requires MNs to have the Java Runtime Environment (JRE) installed. Although this is relatively memory-hungry, our hands-on experience confirms that even the resource-poor PDAs can comfortably support the execution of the JRE. Trivial FTP (TFTP) [14] was used for the distribution of the plugins. It is less complex than FTP and consumes less network resources. The communication between the mobile nodes uses the lightweight XML-RPC protocol [13]. We chose an XML-based approach because we also use XML to represent contextual data collected by mobile nodes. XML handling necessitates a lightweight XML API and in that respect we used the J2ME kXML2 parser (http://kxml.sourceforge.net/).

TABLE 2
JAVA IMPLEMENTATION PACKAGES

| Package | Description |
|---|---|
| **Utils** | Generic classes to handle network connectivity and neighbor discovery, file parsing and information handlers. |
| **Tftp** | Classes that implement the TFTP protocol for file transfers. |
| **Sensors** | Classes that interact with the sensors to collect raw data. A generic sensor interface guides extensibility. |
| **Models** | Specific context information models introduced into the system by an administrator. |
| **Context** | Functionality needed to parse and process context models as these were defined in the "models" package. |
| **Plugin** | Classes that provide the functionality for the dynamic, programmable framework for the dynamic (re-) configuration of mobile nodes. |
| **ccp** | Refers to the CCP entity of the context management framework. |
| **cdp** | Implements the CDP and PDP entities. Cluster manager classes support the organizational model's operation. |
| **cmt** | CMT and Manager Node functionality. The organizational model is supported by means of cluster manager classes. |
| **cr** | Context Repository (CR) related classes. Indexing and storage optimizers are also supported by relevant classes. |

We have designed, developed and implemented the proposed context-aware self-management framework for MANETs. Having utilized J2ME that is targeted for small mobile devices, we assert that the system implementation requirements are satisfied. For the purposes of the proposed context-aware self-management framework we developed 10 packages that fully support the desired functionality in terms of the self-management loop that was previously described. Table 2 describes briefly these packages.

The proper operation of our framework is proved by the experiments performed in the real ad hoc network, i.e. our experimental testbed. These are presented in the following section. Table 3 summarizes software metrics pertaining to our implementation. It is evident that the platform is lightweight since the implementation is considered to be limited in terms of lines of code, a metric which is approximately 7000. A lot of effort was placed on code optimization and refinement and thus led to a small size of the produced code, in the range of a few Kbytes, i.e. 50.

TABLE 3
IMPLEMENTATION SOFTWARE METRICS

| Metric | Value |
|---|---|
| **# of packages** | 11 |
| **Total lines of code** | 7009 |
| **# of classes** | 99 |
| **# of attributes** | 373 |
| **Nested block depth** | Mean: 1.485<br>Std deviation: 0.852 |
| **# of methods** | 499 |
| **McCabe Cyclomatic Complexity** | Mean: 1.858<br>Std deviation: 0.913 |
| **Depth of inheritance tree** | Mean: 1.394<br>Std deviation: 0.489 |

## VI. EVALUATION

The scenario we chose to test on our experimental testbed includes the dynamic change of the routing protocol used in the MANET. MANET routing protocol performance is dependent on the stability of the network itself. Reactive routing protocols are better suited for very volatile network topologies, while proactive approaches for more static MANETs. The scenario involves collecting context information from the surroundings of the mobile nodes and using these to predict node mobility. This information is gathered from the hyper-cluster and a MANET-wide mobility ratio is derived indicating the frequency of prospective topological changes. This ratio is constantly being monitored. In our application scenario, we consider a volatile MANET where the AODV protocol is running (AODV-UU implementation). By monitoring the mobility ratio the hyper-cluster nodes collectively infer – through the appropriate policy monitored objects - that the network is becoming less mobile and thus the routing protocol should switch to a proactive one like OLSR (www.olsr.org implementation) in order to achieve better efficiency. The identified configuration change is signaled through the PEPs to all mobile nodes in the MANET and the AODV plugin is terminated, while the OLSR plugin is activated as previously described.

The scenario serves the purpose of presenting both the self-configuration and self-optimizing aspects of the platform, as well as the platform functionality. The self-configuration aspect is apparent from the scenario itself, while in this case the self-optimizing aspects refer to the fact that by changing the network protocol we achieve better performance of the network by means of bandwidth consumption (proactive and reactive routing protocols consume different amount of bandwidth and work better in different network states).

Results from testbed measurements prove first of all that the framework functions properly, since the routing protocol dynamic switching occurs smoothly and in accordance with the network mobility, while the situation can revert to the original configuration if the necessary conditions are met. The framework as evaluated in our testbed seems to fulfill its goal as being lightweight and deployable on devices with limited resources, e.g. PDAs. The time needed for its initialization is 20 msec for the laptops and 736 msec for the PDAs, while the memory utilization was 47 KB and 49 KB respectively. The differences in time are attributed to the different processing capabilities, while memory consumption is almost identical.

The other parameter of the testbed experimentation validates the efficiency of the framework. From the moment the hyper-cluster identifies the need to alter the routing protocol, up until the activation of the new routing protocol the time required is at acceptable levels, being dependent on the size of the routing plugin and the network size. The dependency is almost linear, as is evident from Table 4, for a network expanding from 5 to 7 nodes (bus topology is considered). The OLSR routing plugin has a size of 150 KB. The times are acceptable since the plugin has a notable size and routing protocol termination (AODV) and activation times (OLSR) are considered.

TABLE 4
OLSR ACTIVATION TIME VS. NETWORK SIZE

| Network Size | OLSR Plugin Distribution & Activation | Similarity Ratio to Linearity |
|---|---|---|
| 5 nodes | 18763.3 msec | 94,8 % |
| 7 nodes | 26681.4 msec | |

## VII. CONCLUSIONS

The quest to achieve self-management of mobile ad hoc networks was what drove the research undertaken in this paper. To this extent we studied the diverse aspects of providing management for MANETs in an autonomic manner. These included the monitoring of surrounding context information; organizing the MANET in a scalable and reliable manner; employing a policy-based management scheme in the MANET realm; managing context information effectively so as to ensure network-wide understanding of conditions in a dynamic fashion; dynamically configuring the MANET according to high-level management decisions; and providing for actual deployment of the proposed self-management framework on a MANET.

Our proposed framework serves as an operating proof-of-concept for our research. Based on our work we assert that we can exploit context awareness to provide a generic and efficient practical framework to achieve self-management of MANETs taking into account their inherent characteristics. Nevertheless, it is essential to stress that self-management cannot be achieved in the absence of the human user. Pure autonomy requires artificial intelligence, long-term research of which has pinpointed the difficulties in achieving relevant goals. The framework we propose exploits user intelligence and intertwines it with system efficiency to realize the task of enabling context-driven self-management of MANETs.

We plan to further experiment with additional case-studies on the self-management of MANETs and complement the

initial evaluation presented in this paper, as the main focus so far was has been the design and implementation of the proposed framework.

## REFERENCES

[1] Haas, R., Droz, P. & Stiller, B., "Autonomic service deployment in networks", IBM Systems Journal, Vol. 42(1), 2003
[2] Kephart, J.O. and Chess, D.M., "The Vision of Autonomic Computing", IEEE Computer, January 2003
[3] Ganek, A. & Corbi, T.A., "The dawning of the autonomic computing era", IBM Systems Journal, Vol. 42(1), 2003
[4] Perkins, C. E., Ad Hoc Networking, 2001 Addison Wesley Longman Inc.
[5] Dong, X., Hariri, S., Xue, L., Chen, H., Zhang, M., Pavuluri, S. and Rao, S., "AUTONOMIA: An Autonomic Computing Environment", IEEE International Conference on Performance, Computing and Communications, April 2003
[6] L. Blazevic, L. Buttyan, S. Capkun, S. Giordano, J-P. Hubaux, J-Y. Le Boudec "Self-Organization in Mobile Ad-Hoc Networks: the approach of Terminodes", IEEE Communication Magazine, Vol. 39 (6), June 2001
[7] Chlamtac, I., Conti, M. & Liu, J. J.-N., "Mobile ad hoc networking: imperatives and challenges", Ad Hoc Networks 1(2003), pp. 13-64, ScienceDirect, Elsevier 2003
[8] A. Hadjiantonis, A.Malatras and G. Pavlou, "A context-aware, policy-based framework for the management of MANETs", 7th IEEE Intl Workshop on Policies for Distributed Systems and Networks (POLICY 06), June 2006.
[9] A. Malatras, G. Pavlou S. Gouveris, S. Sivavakeesar and V. Karakoidas, "Self Configuring and Optimizing Mobile Ad Hoc Networks", Poster, IEEE International Conference on Autonomic Computing (ICAC 2005), June 2005
[10] Chadha, R., Cheng, H., Cheng, Y.-H., Chiang, J., Ghetie, A., Levin, G., and Tanna, H., "Policy-based mobile as hoc network management", 5th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY04), 2004
[11] Bellavista, P., Corradi, A., Montanari, R., Stefanelli, C., "Context-aware middleware for resource management in the wireless Internet", IEEE Transactions on Software Engineering, Vol 29(12), December 2003
[12] Yau, S.S., Karim, F., Wang, Y., Wang, B., Gupta, S.K.S., "Reconfigurable Context-Sensitive Middleware for Pervasive Computing", IEEE Pervasive Computing, July-September 2002
[13] XML-RPC specifications web site, http://www.xmlrpc.com/spec, visited February 2007
[14] K. Sollins, TFTP Protocol, IETF RFC 1350, July 1992
[15] Java 2 Mobile Edition Homepage, http://java.sun.com/javame/index.jsp, visited February 2007
[16] The Familiar Project Homepage, http://familiar.handhelds.org/, visited February 2007