ARUM: A cooperative middleware and an experimentation platform for mobile systems

Marc-Olivier Killijian, Matthieu Roy, Gaétan Séverac

ARUM: a Cooperative Middleware and an Experimentation Platform for Mobile Systems

Marc-Olivier Killijian
LAAS-CNRS
Toulouse, France
mkilliji@laas.fr

Matthieu Roy
LAAS-CNRS
Toulouse, France
roy@laas.fr

Gaëtan Séverac
LAAS-CNRS
Toulouse, France
gseverac@laas.fr

***Abstract*—In this paper, we present a middleware architecture for dependable mobile systems and an experimentation platform for its evaluation. The proposed architecture includes three building blocks tailored for mobile cooperative applications: a Proximity Map, a Trust and Cooperation Oracle, and a Cooperative Data Backup service. To illustrate our platform, we developed a Distributed Black-box application, whose aim is to record critical data while tolerating the failure of a node, and implemented a hardware evaluation platform of mobile systems for experimenting with the application. We provide here some insights on the development of the platform, focusing on wireless communication emulation *via* signal attenuation.**

I. Problem Statement

Finding the proper abstractions to design middleware for the provision of dependable distributed applications on mobile devices is still a big challenge[1]. Yet, the number of mobile communicating devices one can meet in every-day life is dramatically increasing; the interconnection of these systems does not only result in a huge distributed system. New technical and scientific challenges emerge due to the mobility of users and of their devices, or due to the massive scale of uncontrolled devices that constantly connect and disconnect, fail, etc. An important question is thus to know if and how can we design a sound middleware that offers useful building blocks for this type of system. Another crucial question is to study how we can correctly evaluate those highly mobile and dynamic systems.

This paper addresses these two questions: we present a *middleware architecture* dedicated to the provision of cooperative data backup on mobile nodes and a *platform* for its experimental evaluation. This architecture is exemplified by implementing a Distributed Black-Box (DBB) application which provides a virtual device, whose semantics is similar to avionics black-boxes, that tracks cars' history in a way that can be replayed in the event of a car accident. This application ensures information is securely stored using replication mechanisms, by means of exchanging positions between cars. Our implementation is based on three services tailored for mobile cooperative applications: a *Proximity Map*, a *Trust and Cooperation Oracle*, and a *Cooperative Data Backup*.

This work was partially supported by the French National Center for Scientific Research (CNRS), the European Hidenets project (EU-IST-FP6-26979), and the European ReSIST network of excellence (EU-IST-FP6-26764).

This DBB application is a good illustration of the use of the various middleware services and applications that users can benefit thanks to mobile communicating devices, such as in the automobile context with car-to-car communication. As a "classical" black-box, its aim is to record critical data, such as: engine / vehicle speed, brake status, throttle position, and even the state of the driver's seat belt switch. As a "smart" black-box, it can also be used for extending the recorded information with contextual information concerning the neighboring vehicles, possibly the various vehicles involved in an accident. Indeed, information stored by the application leverages vehicle-based parameters and communication-induced information.

The proposed architecture is based on four main middleware building blocks, namely a Networking service, a Proximity Map, a Trust and Cooperation Oracle, and a Cooperative Data Backup service. This architecture and the DBB application will be described in Section II. This distributed architecture being targeted to mobile nodes (e.g. automobiles), it has been implemented and evaluated on top of a mobile robot platform described in section III. Section IV concentrates on a particular aspect of the developed scaled-down platform, namely WiFi communication range reduction using signal attenuation. Finally, we give some further trails of research in Section V.

II. Architecture and system model

This work was conducted in the course of the Hidenets project. HIDENETS (HIghly DEpendable ip-based NETworks and Services) was a specific targeted research project funded by the European Union under the Information Society Sixth Framework Programme. The aim of HIDENETS was to develop and analyze end-to-end resilience solutions for distributed applications and mobility-aware services in ubiquitous communication scenarios.

The overall architecture used in this work is depicted on Fig. 1. The mobile platform, the hardware and other experimental settings will be described later in Section III. This architecture is a partial implementation of the Hidenets architecture and has been detailed in the projects deliverables, see e.g. [2] or [3]. Apart from standard hardware-related services (networking, localization...), we propose three new building blocks, targeted for mobile systems, that are described in the following subsections. The rationale of these building blocks is as follows:

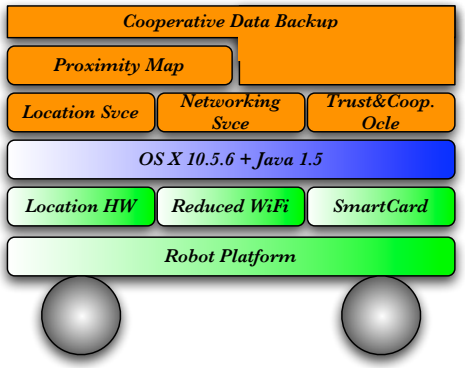

Fig. 1. Overall Architecture

a) Proximity map: Before being able to backup data, a mobile node has first to discover its neighbors and the resources and services they offer. The proximity map represents the local knowledge a node has about its vicinity. This can vary according to wideness (the number of communication hops represented on the map) and according to accuracy (how often is the map updated). Notice that, in this work, the aim of the proximity map is twofold: it is used to know which nodes can be used for cooperation, and is also used as a source of data to be backed up by the distributed black-box application, as we shall see later.

b) Trust and cooperation oracle: In order to interact with a priori unknown neighbors for critical services (e.g., collaborative backup), a node has to evaluate the level of trust it can assign to each of its neighbors. The purpose of the trust and cooperation oracle is to evaluate this level of trust and to incite nodes to cooperate with one another.

c) Cooperative data backup: The provision of a cooperative backup service at the middleware level is the major contribution of the architecture described in this paper. This service acts as a peer-to-peer storage resource sharing service for backup and restoration of critical data. There are four main tasks to achieve when considering cooperative data backup : (1) discovering storage resources in the vicinity, (2) negotiating a contract with the neighboring cars for the use of their resources, (3) handling a set of data chunks to backup and assigning these chunks to the negotiated resources according to some data encoding scheme and with respect to desired properties like dependability, privacy, confidentiality, and finally (4) taking care of the recovery phase, i.e., the data restoration algorithm.

A. Communication and network layer

Since Java provides no specific support for ad hoc networking, we implemented a specific package for handling multiple WiFi interfaces. This package supports both UDP broadcasting and TCP unicasting. It handles indexing, choping and unchoping of arbitrary size messages and deals with typed messages. As we are only interested in local interactions within an entity's neighborhood, our network layer implements one-hop interactions only, and does not address the problem of routing in an ad-hoc network.

B. Localization and Proximity Map

In many applications dedicated to mobile nodes, and especially for cooperation-based applications, a node needs to interact with its neighbors. Furthermore, the quality of service that may be provided by a given component can vary according to the vicinity, e.g. the quantity of neighbors, their density, etc. It is then necessary to formalize this view of the vicinity into a more abstract representation. To that means, we propose the *Proximity Map* building block, that provides an abstraction of the physically attainable network of entities. The aim of this building block is to provide applications with information aggregated from both localization and networking layers.

Indeed, the goal of the proximity map is to gather physical information about nodes in the vicinity. When using its proximity map, a given node has a view of the nodes in its vicinity (defined as being the nodes which are reachable within H hops), their location information, and the freshness of the pieces of information.

This problem has similarities with neighbor discovery protocols for ad-hoc routing algorithms, that can be divided into pro active schemes and reactive schemes. In a reactive scheme, information about routing is constructed on demand, i.e., as soon as a message has to be sent to a previously unknown destination. In a pro active scheme, the entity periodically sends messages on the network to look for new neighbors, and to check the availability and reachability of already discovered requested by the application/caller. Since we are only interested in local interactions, and due to the fact that the set of entities is large and unknown to participants, we designed the proximity map as a pro active service.

Intuitively each node periodically beacons its proximity map to its 1-hop neighbors, and collects similar information from its direct neighbors. When merging these pieces of information, it is able to update its proximity map with new nodes that appeared as neighbors of neighbors, nodes which have moved, nodes whose connectivity changed, etc. The preliminary ideas about the proximity map can be found in [4].

To implement the proximity map, we use location-stamped beacons. Each node keeps a map of its knowledge of the location and connectivity of other nodes, which is represented as a graph as shown in Fig. 2. This graph is regularly updated when the node receives a beacon and is also regularly sent to the node's neighbors in its beacons.

d) Map wideness: To understand the protocol, let's look at the initialization phase:

1) First, the node knows only its own location (level 0), and broadcasts this information in beacons,
2) Second, when the node receives its neighbors' beacons, including their locations, it knows about its one hop connectivity (level 1), and the next beacons it will send will include this updated information,

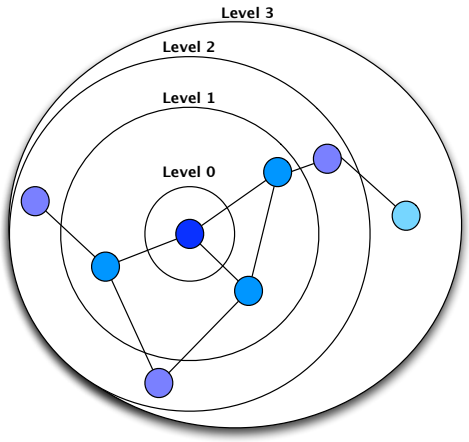

Fig. 2. The Proximity Map Knowledge

3) Then, the node receives beacons including its neighbor's knowledge (level L-1) and updates its own graph with this information (level L).

When the protocol has ended its initialization phase, information is updated by broadcasting a node's information, and collecting information, limiting the number of levels to a threshold H, as described below.

e) Map accuracy: If beacons are sent every D time units, level L information is $(L * D)$ old. Because high-level knowledge is older and because it is not desirable to have the knowledge of the whole network, the maximum level of knowledge is bounded. This bound, H, is determined statically according to the application, the density of the network and other environment parameters.

This algorithm is pro-active and enables a node to know the other nodes physically present within the area if H is sufficiently large. When H is not large enough, or the coverage obtained is not sufficient, a reactive protocol may be used to complete location information further than H hops. In addition to the location of the node, beacons may also include other useful information about the sending node's radio coverage, its battery life, etc. At any time after $(H * D)$ time units, a node knows the location and coverage of its H hops neighbors. It should however be noted that at time t, every node in the network has a different view of the connectivity since its level 1 information is D time units old, its level n is $n * D$ time units old, etc.

C. Trust and Cooperation Oracle

The trust and cooperation oracle (TCO) is our second basic building block for cooperative services. A cooperative service emerges from the cooperation of entities that are generally unknown to one another. Therefore, these entities have no a priori trust relationship and may thus be reluctant to cooperate. In cooperative systems without cooperation incentives, entities tend to behave in a rational way in order to maximize their own benefit from the system. The goal of the trust and cooperation oracle is therefore to evaluate locally the level of trust of neighboring entities and to manage cooperation incentives [5].

Synergy is the desired positive effect of cooperation, i.e., that the accrued benefits are greater than the sum of the benefits that could be achieved without cooperation. However synergy can only be achieved if nodes do cooperate rather than pursuing some individual short-term strategy, i.e. being rational. Therefore, cooperative systems need to have cooperation incentives and rationality disincentives. There are several approaches to this, some are based on micro-economy and some others are based on trust. Typically, for micro-economic approaches, a node has to spend "money" for using a service and earns "money" for servicing other nodes. Regarding trust, a common approach is to use the notion of reputation, a level representing the level of trust that may be placed on a node, which can be computed locally by a single node, or collectively and transitively by a set of nodes. Another approach based on the notion of trust relies on the use of trusted hardware, e.g. a smart-card. Whatever the most appropriate approach in a given context, the TCO leverages this information by providing a single interface with simple semantics. Given a node identifier n, it returns the probability that this node n cooperates correctly for the next interaction:

```
float trustLevel(NodeID n)
```

When the various entities participating in a cooperative service belong to the same administrative domain, or to a limited number of domains, the question of trust establishment can be answered in a simple manner. For example, if we consider the case of a single administrative domain such as an enterprise, we can make the assumption that any node within the enterprise is going to cooperate. The problem of the trust establishment is thus reduced to the question of identifying the nodes which are part of the enterprise. When multiple, but limited, administrative domains are involved, the question can sometimes be simplified in a similar manner.

In an automotive context, we consider that there are a limited number of different middleware providers. We can also state that it is at least unusual and potentially dangerous for vehicle owners to modify the software their vehicle is running, and that software updates are relatively rare. As a result, there are only a few different legacy middleware versions. We can thus consider that the middleware is certified, i.e., a trusted authority within the infrastructure domain can generate and distribute certificates. These certificates can be verified in the ad-hoc domain by a trusted hardware, in our platform a smart-card.

D. Cooperative Data Backup service

The cooperative backup service aims to improve the dependability of data stored by participating nodes by providing them with mechanisms to tolerate hardware or software faults, including permanent faults such as loss, theft, or physical damage. To tolerate permanent faults, the service must provide mechanisms to store the users' data on alternate storage nodes using the available communication means. The problem of cooperative backup of critical data can be divided in three steps: i) discovering storage resources in the vicinity (this step is performed using the proximity map service), ii) negotiating

a contract with the neighboring nodes for the use of their resources (this step uses the trust and cooperation oracle), and iii) handling a set of data chunks to backup and assigning these chunks to the negotiated resources according to a data encoding scheme and with respect to desired properties of dependability, privacy, availability and confidentiality. The service is also in charge of the recovery phase, i.e., the data restoration algorithm.

The Cooperative Data Backup service provision is designed using the following principles:

• A client of the service provides a data stream to be backed up to the `backup` operation with a unique identifier.

• The stream passes through a series of chopping and indexing operations in order to produce a set of small (meta-) data chunks to be backed up (more details can be found in [6]).

• A periodical backup thread processes the block buffer, queries the Proximity Map service and the Trust and Cooperation Oracle in order to produce a potential contributors list. Then it places data blocks on contributors according to given placement and replication strategies, as described in [6], [7].

When the client wants to restore data, it can either submit the unique identifier of the stream to the asynchronous restore operation and then poll it periodically, or it can directly call the synchronous restore operation that will return when the data has been successfully restored. To that means, a periodic thread handles the restoration waiting queue: it looks for given IDs, unpacks the received blocks and potentially adds new identifiers to the waiting queue according to the decoding operation on received data chunks (i.e. data or meta-data).

E. The Distributed Black-Box application

Using the above described services, we implemented a Distributed Black-Box application. This application backs up a stream of data for every car that consists of a periodic sampling of a car's proximity map. In our implementation, the cooperative backup service replicates these streams among neighboring cars, or to an infrastructure when connectivity permits it. The stream of any participant (be it crashed or not) can then be restored either from neighboring devices (cars in ad-hoc mode), or from the infrastructure.

III. The ARUM experimental platform

To the best of our knowledge, little research has been done on the evaluation of resilience in ubiquitous systems. Most of the literature in this domain concerns evaluation of users experience and human-computer interfaces. However, some work is also looking at defining appropriate metrics for the evaluation of distributed applications running on ubiquitous systems [8], [9]. [10] is looking at a general approach to evaluate ubiquitous systems. In the paper, the authors argue that quantitative measurements should be complemented with qualitative evaluation. The argument is that there is a number of problems for which evaluation cannot be easily quantified. Thus an evaluation should be conducted using an hybrid quantitative/qualitative strategy.

It is clear that the area of resilient computing has proposed a number of contributions concerning the evaluation of distributed systems and this paper will not survey this domain. Analytical evaluation is probably the most popular technique, such as within Assert [11] in the avionics application domain. More recently, experimental evaluation started to gain attention. The approach taken is often based on dependability benchmarking, for example DBench [12] addresses dependability benchmarking of operating systems.

In the ubiquitous and mobile computing area, evaluation of resilient mechanisms remains an open problem. In most cases, the proposed algorithms are evaluated and validated using network simulators [13], [14]. Since simulators use a model of physical components, such as wireless network cards and location systems, this raises concerns on the coverage of the assumptions that underlie the simulation [15]. Little work concerning the evaluation of algorithms in a realistic mobile experimental environment is available, see III-D.

A. "Scale ability".

This calls for the development of a realistic platform, at a laboratory scale, to evaluate and validate fault-tolerance algorithms (e.g., group membership and replication protocols, storage mechanisms, etc.) targeting systems comprising a large number of communicating mobile devices equipped with various sensors and actuators. The goal is to have an experimentation platform allowing for reproducible experiments (including mobility aspects) that will complement validation through simulation. As we will see, an important issue within this platform is related to changes of scale so as to emulate as many various systems as possible.

We are developing an experimental evaluation platform composed of both fixed and mobile devices [16]. Technically speaking, each mobile device is composed of some programmable mobile hardware able to carry the device itself, a lightweight processing unit equipped with one or several wireless network interfaces and a positioning device. The fixed counterpart of the platform contains the corresponding fixed infrastructure: an indoor positioning system, wireless communication support, as well as some fixed servers. Our platform is set up in a room of approximately $100m^2$ where mobile devices can move around. By changing scale, we can emulate systems of different sizes. Hardware modeling of this type of system requires a reduction or increase of scale to be able to conduct experiments within the laboratory. To obtain a realistic environment, all services must be modified according to the same scale factor.

For example, if we consider a VANET experiment like the DBB, a typical GPS in a moving car is accurate to within $5 - 20m$. So, for our $100m^2$ indoor environment to be a scaled down representation of (say) a $250000m^2$ outdoor environment (a scale reduction factor of 50), the indoor positioning accuracy needs to be $10 - 40cm$. The following table summarizes the required change in scale for all peripherals of a node.

Device	Real Accuracy	Scaled Accuracy
Wireless	range: $100m$	range: $2m$
GPS	$5m$	$10cm$
Node size	a few meters	a few decimeters
Node speed	a few $m.s^{-1}$	less than $1m.s^{-1}$

B. Technological aspects.

1) Positioning: Several technologies are currently available for indoor location [17], mostly based either on scene analysis (e.g. using motion capture systems) or on triangularization (of RF and ultrasound [18] or wireless communication interfaces [19]). In this section, we describe the various systems we used, and analyze their interests and limitations.

To reach our desired level of accuracy for indoor positioning, we first used a dedicated motion capture technology that tracks objects based on real-time analysis of images captured by infra-red cameras. The Cortex[1] system is able to localize objects at the millimeter scale, which is more than enough for the VANET setting. This technology uses a set of infrared cameras, placed around the room, that track infrared-visible tags. All cameras are connected to a server that computes, based on all cameras images, the position of every tag in the system. We equipped our small robots with such tags, and the computers on the robots connect to the server to get their positioning information. Although the precision attained was more than enough for our needs, the system has some drawbacks: the whole system is very expensive (in the order of $100k$), calibration is a tedious task, and infrared signals cannot cross obstacles such as humans.

To overcome these limitations, we are currently developing a new localization system, based on two different technologies that have complementary advantages. The first one is based on infrared cameras, as for Cortex, but the system is reversed: cameras are on-board, and locate themselves by tracking statically placed infrared-visible tags. This system is coupled with an Ultra-Wide-Band-based localization system, Ubisense. Ultra-Wide-Band-based localization (UWB) is performed by 4 sensors, placed in the room at each corner, that listen for signals sent by small tags that emit impulses in a wide spectrum. Such impulses can traverse human bodies and small obstacles, so the whole system is robust to external perturbation, but, from our preliminary measurements, attainable precision is about $10cm$.

We thus advocate that the coupling of these two technologies will result in a localization system with desirable properties: it is relatively cheap, it is robust to external perturbations such as obstacles, and has most of the time a precision about the order of a centimeter.

2) Mobility: Another important question is how to make the devices actually mobile. Obviously, when conducting experiments, a human operator cannot be behind each device, so mobility has to be automated. This is why we considered the use of simple small robot platforms in order to carry around the platform devices. The task of these robots is to "implement" the mobility of the nodes. The carried devices communicate with the robot through a serial port. This way they can control the mobility, i.e. the trajectory, the stops and continuations, the fault-injection, etc.

A node in the system is implemented using a laptop computer, that includes all hardware devices and the software under testing, that is carried by a simple robotic platform, the Lynxmotion 4WD rover. A 4WD rover is able to carry a full node during a few hours, running at a maximum speed of $1m.s^{-1}$, which is consistent with our assumptions.

To have reproducible patterns of mobility, the rover embarks a dedicated software that moves the robot using two different schemes. Both designs allow for testing different algorithms using the same mobility pattern, and for testing the same algorithm with different mobility scenarios.

In the simple scheme, a robot is following a black line on the floor. This solution is easy to implement but imposes that the operator "draws" the circuit for every different mobility pattern.

The second scheme couples a predefined mobility pattern with the positioning service and ensures a given node moves according to the predefined pattern, programmed by the operator. This solution is more flexible: each node has its own mobility pattern specified for each experiment.

3) Communication: The last and most important design issue for the platform concerns wireless communications. Indeed, the communication range of the participants (mobile nodes and infrastructure access-points) has to be scaled according to the experiment being conducted. For example, with a VANET experiment, a typical automobile has a wireless communication range of a few hundred meters, say 200m. With a scale reduction factor fixed at 50, the mobile devices communication range has to be limited to 4m. However, to cope with other experiments and other scale reduction factors, this communication range should ideally be variable.

A satisfying solution consists in using, for this purpose, signal attenuators placed between the WiFi network interfaces and their antennas. An attenuator is an electronic device that reduces the amplitude or power of a signal without appreciably distorting its waveform. Attenuators are passive devices made from resistors. The degree of attenuation may be fixed, continuously adjustable, or incrementally adjustable. In our case, the attenuators are used to reduce the signal received by the network interface. The necessary capacity of the attenuators depends on many parameters such as the power of the WiFi interfaces and the efficiency of the antennas, but also on the speed of the robot movements, the room environment, etc. The experiments and conclusions regarding the use of signal attenuators is the subject of Section IV.

C. Application scenario

As can be seen on Fig. 1, the middleware described in this article is running on top of Apple OS X.5.6 and Java 1.5. The hardware (Macbook with additional WiFi interface and some localization hardware) is carried by a Lynxmotion 4WD rover. The resulting platform can be seen on Fig. 3. We currently own

[1] `http://www.motionanalysis.com`

Fig. 3. The ARUM platform

four fully equipped robots. We were thus able to emulate the Distributed Black-Box in a setting with three cooperating cars and a police car coming after an accident to recover data. During the first part of the scenario, the three cars backup Black-Box data for each other, then one of the cars looses control and leaves the circuit track to crash in a wall. After the accident has been reported, including the identifier of the crashed car and the approximated time of the accident, the police enters the scene and requests restoration of the black box data for a given period of time that surrounds the accident. Once the data is successfully restored, the police is then able to replay the film of the accident, and to identify the other involved cars if there is any. A movie demonstrating this scenario is available at `http://homepages.laas.fr/mroy/hidenets/`.

D. Related work.

Relatively little work concerning the evaluation of algorithms in a realistic mobile experimental environment is available. Most of the available platforms are based on wired emulation of wireless networks [20]. Wired wireless emulators such as EMPOWER [21], and EMWIN [22] use a centralized emulation layer and rely on switching equipment to disseminate messages to "mobile" nodes. Non-centralized wireless testbed emulators such as SEAWIND [23] or SWOON [24] rely on a wired configurable testbed similar to Emulab [25]. These testbed emulators rely on various link shaping techniques to approximate a wireless link. Typically, a special node is used for one or more links that need to be emulated. The quality of the emulation can suffer since these testbeds utilize switching equipment and multiple nodes to propagate messages. Both Mobile Emulab[26] and MiNT [27] use robots to emulate mobility of wireless nodes. The mobile version of emulab embarks Motes to emulate a wireless sensor network. The wireless experiments is carried at the building scale. Like our platform, Mint uses signal attenuators to reduce the space needed for the experiments [28]. However, in order to reduce the Mint's node costs, the positioning subsystem is based on simple web-cams and henceforth is not precise.

IV. Experimenting signal attenuation

The objective of these experiments is to find the practical relationship between signal attenuation and communication range. More precisely, the ultimate goal is to be able to select the appropriate attenuation value according to some target range (corresponding to a given scaling factor). This is necessary when one wants to evaluate in a laboratory setup a certain prototype.

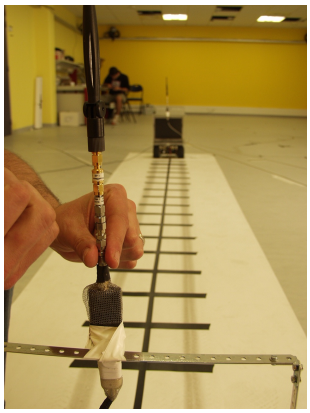

Fig. 4. Attenuators

It must be noted that within this paper, we present only the results for attenuation between 34 and 39 db. We experimentally established that the correct attenuation level for our scenario was around these values. Indeed, we are interested with the evaluation of several Vehicular Network applications, within the scope of the Hidenets project[3]. In particular, we are developing a distributed black-box application: each car possesses a virtual black-box implemented by cooperating neighbor nodes. The vehicles store recent events regarding themselves and their vicinity in this black-box. The evaluation platform will be used for verifying the resilience properties offered by the application, essentially tolerance to crash of the nodes. The evaluation for this application involves 4 cars, represented by our robots, that follow a track and overtake each other. One of the car has an accident and the black-box must remain available despite this accident. In this scenario, the correct communication range to fit our lab is about 2m. As you will see in the following section, attenuations between 34 and 39 db are appropriate for this communication range. For illustration, Fig. 4 shows a USB WiFi interface connected to its external antenna through a set of attenuators.

A. Description of the experiments

As can be seen on Fig. 4, this experiment involves two laptops mounted on robot platforms and using an external WiFi interface to communicate with each other. One of the two nodes is static and the other one moves back and forth. Equivalent attenuators are attached between each external WiFi interface and its antenna. The mobile platform moves along a line, stops every $20cm$ for $5min$ and performs a measurement at every stop. For each measurement, the moving laptop joins the ad hoc network created by the fixed one, measures the communication throughput and then leaves the ad hoc network. The time for joining the network is logged, as is the measured throughput. A complete experiment is composed of 100 repetitions of a return trip along the 5m line. This data

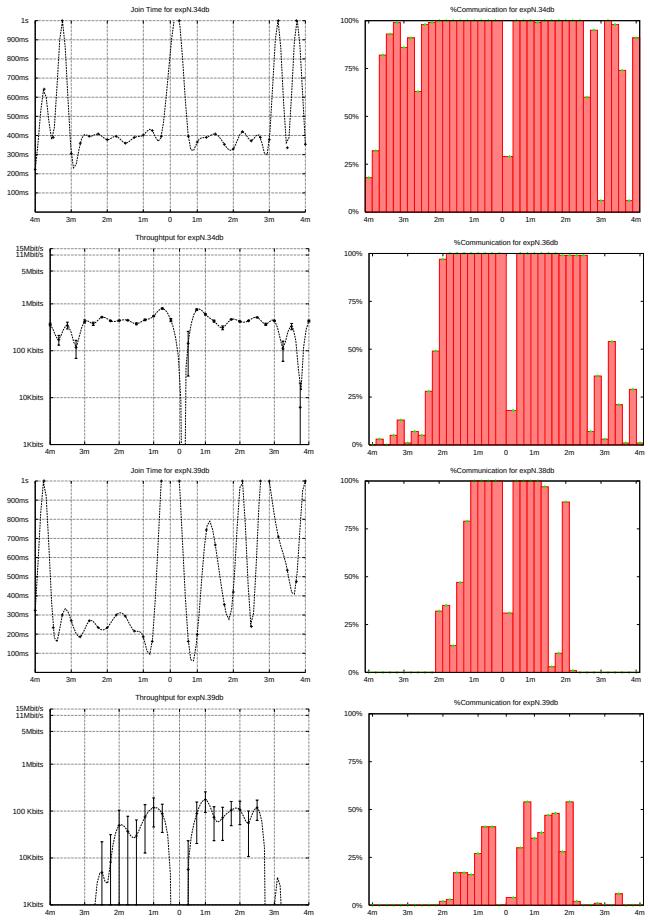

Fig. 5. Adhoc network join time, throughput and percentage of correct communication

is logged and statistically analyzed offline, as explained in the following subsection.

B. Results

As mentioned above in the experiment description, we were able to measure several parameters: time to join the adhoc network, throughput, and number of communications. In the following, we analyze the results obtained according to these various aspects of the statistical analysis.

1) Time to join the adhoc network: As one can see on Fig. 5, while signal attenuation certainly has an impact on the time to join the adhoc network, this impact is difficult to clearly establish. This is due to the fact that we used a watchdog in order to stop an experiment when it lasted too long, i.e. when we suspected that the communication will never be established. In those cases we assumed the maximum value for this particular measure in the mean value computation for the experiment. This leads to a similar problem as we had with the throughput measurements, as you will see in the following subsection.

2) Throughput: Again, as with the previous measurement, the results concerning throughput shown on Fig. 5 are difficult to interpret. The mean throughput drops from approximatively 1Mbits/s with 34db to 100kbits with 39db but the tendency is not really clear and the mean value seems to be a bad approximation. Indeed, because often no communication is established, many zeros pollute the data and make it difficult to find an appropriate metric.

3) Established communications: In order to account for the many zeros that prohibit to use means to analyze our data, we decided to compute the number of correct communications for an experiment. A communication is said to be correct when for a measurement, some bits were exchanged, whatever the quality of the link, i.e. whatever the throughput as soon as it is not zero. This gives the results presented in Fig. 5. These figures show the percentage of correct communications. We can see for example on the 34db figure that from 0 to 2.5m, a communication is always established. There's a singular point around 0, we suspect that this is related to the wavelength, the high of the antenna and the proximity, but this will necessitate further investigation. Between 2.5m and 4m almost all communications are correct even though around 4m it gets worse. If we analyze now the results for 36db, we can say that any communication up to 2m is correct and that almost no communication is possible any further. A similar conclusion can be established regarding 38db for up to 1m. Concerning 39db, we can say that signal attenuation is too strong and that this is the limit at which no communication can be established correctly.

C. Results

To summarize, we established with these experiments that signal attenuators can be used to scale communication range down in a controlled and repeatable manner. For example, the results presented above show that in our conditions, using our hardware in our laboratory, one can use the following attenuation values to reach specific communication ranges.

attenuation	approx. range
34db	2.5m
36db	2m
38db	1m

V. Conclusion and Further Work

We presented a middleware architecture for dependable ubiquitous mobile applications that provides new building blocks based on cooperative approaches. We believe that those building blocks provide useful abstractions in the context of future emerging applications in the future *mobiquitous* society. In this work, we were concerned with the provision of a critical data backup service, a trust and cooperation oracle, and a proximity map. These services help building reliable mobile applications as we shown on a distributed black box example for automobiles.

We also described a new platform for the experimental evaluation of this type of mobile application. Indeed, the evaluation of mobile systems is often based on the use of network simulators, which are often not satisfying, especially when dependability is an important issue. Notice that, due to

its design, our platform is able to emulate mobile systems of variable size by adapting communication range.

We believe that the usual mobility models used for the evaluation of mobile systems are not satisfactory. A mobility model dictates how the nodes, once distributed in the space, move. A mobility model involves the nodes' location, velocity and acceleration over time. The topology and movement of the nodes are key factors in the performance of the system under study. Because the mobility of the nodes directly impacts the performance of the protocol, if the mobility model does not reflect realistically the environment under study, the result may not reflect the performance of the system in the reality. The majority of existing mobility models for ad hoc networks does not provide realistic movement scenarios [29]. We are currently working on the use of real mobility traces from various sources in order to build more realistic mobility models to use in our analytical and experimental evaluation [30].

We are also improving the software that controls the mobility aspects of the platform. At the moment the robots follow a tape track on the ground. The currently developed version enables the setup of virtual tracks and allows for the differentiation of the various robots, i.e. , each robot is able to follow its own trajectory. Once upgraded, the evaluation platform described in this paper will be able to run experiments according to the realistic mobility models mentioned above, since the platform will be able to use any "mobility pattern" as an input for performing reproducible experiments.

REFERENCES

[1] M. Roy, F. Bonnet, L. Querzoni, S. Bonomi, M.-O. Killijian, and D. Powell, "Geo-registers: An abstraction for spatial-based distributed computing," in *Int. Conf. On Principles Of DIStributed computing (OPODIS), LNCS 5401*, 2008, pp. 534–537.

[2] J. Arlat and M. Kaâniche(editors), "Hidenets. revised reference model," LAAS-CNRS, Contract Report nr. 07456, Sept. 2007.

[3] A. Casimiro et al., "Resilient architecture (final version)," LAAS-CNRS, Tech. Rep. 08068, December 2008. [Online]. Available: `http://www.di.fc.ul.pt/tech-reports/07-19.pdf`

[4] M.-O. Killijian, R. Cunningham, R. Meier, L. Mazare, and V. Cahill, "Towards group communication for mobile participants," in *in Proceedings of Principles of Mobile Computing (POMC)*, 2001, pp. 75–82.

[5] L. Courtès, M.-O. Killijian, and D. Powell, "Security rationale for a cooperative backup service for mobile devices," in *Proceedings of the Latin-American Symposium on Dependable Computing (LADC)*. Springer-Verlag, 2007, pp. 212–230.

[6] L. Courtes, M. Killijian, and D. Powell, "Storage tradeoffs in a collaborative backup service for mobile devices," in *European Dependable Computing Conference (EDCC)*, 2006, pp. 129–138.

[7] L. Courtes, O. Hamouda, M. Kaaniche, M. Killijian, and D. Powell, "Dependability evaluation of cooperative backup strategies for mobile devices," in *Pacific Rim Dependable Computing*, 2007, pp. 139–146.

[8] P. Basu, W. Ke, and T. D. C. Little, "Metrics for performance evaluation of distributed application execution in ubiquitous computing environments," *Workshop on Evaluation Methodologies for Ubiquitous Computing at Ubicomp'01*, 2001. [Online]. Available: `http://zing.ncsl.nist.gov/ubicomp01/`

[9] P. Castro, A. Chen, T. Kremenek, and R. Muntz, "Evaluating distibuted query processing systems for ubiquitous computing," *Workshop on Evaluation Methodologies for Ubiquitous Computing at Ubicomp'01*, 2001. [Online]. Available: `http://zing.ncsl.nist.gov/ubicomp01/`

[10] M. Burnett and C. P. Rainsford, "A hybrid evaluation approach for ubiquitous computing environments," *Workshop on Evaluation Methodologies for Ubiquitous Computing at Ubicomp'01*, 2001. [Online]. Available: `http://zing.ncsl.nist.gov/ubicomp01/`

[11] J. Arlat, M. R. Barone, Y. Crouzet, J.-C. Fabre, M. Kaaniche, K. Kanoun, S. Mazzini, M. R. Nazzarelli, D. Powell, M. Roy, A. E. Rugina, and H. Waeselynck, "Dependability needs and preliminary solutions concerning evaluation, testing and wrapping," LAAS, Toulouse, Tech. Rep. 05424, 2005.

[12] K. Kanoun, H. Madeira, F. Moreira, M. Cin, and J. Garcia, "Dbench - dependability benchmarking," in *Proc. of the Lecture Notes in Computer Science (LNCS), Springer-Verlag, Fifth European Dependable Computing Conference (EDCC-5)*, April 2005.

[13] S. R. Das, R. Castañeda, and J. Yan, "Simulation-based performance evaluation of routing protocols for mobile ad hoc networks," in *Mob. Netw. Appl.*, vol. 5, no. 3. Hingham, MA, USA: Kluwer Academic Publishers, 2000, pp. 179–189.

[14] E. B. Hamida, G. Chelius, and J. M. Gorce, "On the complexity of an accurate and precise performance evaluation of wireless networks using simulations," in *11th ACM-IEEE Int. Symp. on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM)*, 2008.

[15] D. Cavin, Y. Sasson, and A. Schiper, "On the accuracy of manet simulators," in *POMC '02: Proceedings of the second ACM international workshop on Principles of mobile computing*. New York, NY, USA: ACM Press, 2002, pp. 38–43.

[16] M. Killijian, N. Rivière, and M. Roy, "Experimental evaluation of resilience for ubiquitous mobile systems," in *Proc. of UbiComp, Workshop on Ubiquitous Systems Evaluation (USE)*, 2007, pp. 283–287.

[17] J. Hightower and G. Borriello, "A survey and taxonomy of location systems for ubiquitous computing," 2001. [Online]. Available: `citeseer.ist.psu.edu/hightower01survey.html`

[18] A. Smith, H. Balakrishnan, M. Goraczko, and N. B. Priyantha, "Tracking Moving Devices with the Cricket Location System," in *2nd International Conference on Mobile Systems, Applications and Services (Mobisys 2004)*, Boston, MA, June 2004.

[19] N. S. Correal, S. Kyperountas, Q. Shi, and M. Welborn, "An uwb relative location system," in *Proc. of IEEE Conference on Ultra Wideband Systems and Technologies*, November 2003.

[20] D. Havey, R. Chertov, and K. Almeroth, "Wired wireless broadcast emulation," in *5th International workshop on Wireless Network Measurements (WiNMee)*, 2009.

[21] P. Zheng and L. M. Ni, "Empower: A network emulator for wireline and wireless networks," in *In Proceedings of IEEE InfoCom. IEEE Computer and Communications Societies*, 2003.

[22] ——, "Emwin: Emulating a mobile wireless network using a wired network," in *In Proceedings of the 5th ACM international workshop on Wireless mobile multimedia*. ACM Press, 2002, pp. 64–71.

[23] M. Kojo, A. Gurtov, J. Manner, P. Sarolahti, T. Alanko, and K. Raatikainen, "Seawind: a wireless network emulator," in *In Proceedings of 11th GI/ITG Conference on Measuring, Modelling and Evaluation of Computer and Communication Systems*, 2001.

[24] Y. L. Huang, J. D. Tygar, H. Y. Lin, L. Y. Yeh, H. Y. Tsai, K. Sklower, S. P. Shieh, C. C. Wu, P. H. Lu, S. Y. Chien, Z. S. Lin, L. W. Hsu, C. W. Hsu, C. T. Hsu, Y. C. Wu, and M. S. Leong, "Swoon: a testbed for secure wireless overlay networks," in *CSET'08: Proceedings of the conference on Cyber security experimentation and test*. Berkeley, CA, USA: USENIX Association, 2008, pp. 1–6.

[25] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar, "An integrated experimental environment for distributed systems and networks," in *Proc. of the Fifth Symposium on Operating Systems Design and Implementation*. Boston, MA: USENIX Association, Dec. 2002, pp. 255–270.

[26] D. Johnson, T. Stack, R. Fish, D. M. Flickinger, L. Stoller, R. Ricci, and J. Lepreau, "Mobile emulab: A robotic wireless and sensor network testbed," in *INFOCOM*. IEEE, 2006.

[27] T.-c. Chiueh, R. Krishnan, P. De, and J.-H. Chiang, "A networked robot system for wireless network emulation," in *RoboComm '07: Proceedings of the 1st international conference on Robot communication and coordination*. Piscataway, NJ, USA: IEEE Press, 2007, pp. 1–8.

[28] P. De, A. Raniwala, S. Sharma, and T. cker Chiueh, "Mint: a miniaturized network testbed for mobile wireless research," in *INFOCOM*. IEEE, 2005, pp. 2731–2742.

[29] M. Musolesi and C. Mascolo, "Mobility models for systems evaluation," in *State of the Art on Middleware for Network Eccentric and Mobile Applications (MINEMA). Springer*, February 2009.

[30] M. Killijian, M. Roy, and G. Trédan, "Beyond san francisco cabs: building a *-lity mining dataset," in *NetMob 2010, Workshop on the Analysis of Mobile Phone Networks*, 2010.