Data Backup for Mobile Nodes : a Cooperative Middleware and Experimentation Platform

Marc-Olivier Killijian, Matthieu Roy, Gaétan Séverac, Christophe Zanon
CNRS ; LAAS ; 7 avenue du colonel Roche; F-31077 Toulouse, France
Université de Toulouse ; UPS, INSA, INP, ISAE ; LAAS ; F-31077 Toulouse, France
Contact: http://www.laas.fr/~mkilliji ; mkilliji@laas.fr

Abstract

In this paper, we present a middleware for dependable mobile systems and an experimentation platform for its evaluation. The middleware proposed is based on three original building blocks: a Proximity Map, a Trust and Cooperation Oracle, and a Cooperative Data Backup service. A Distributed Black-box is used as an illustrative application and evaluated on top of the proposed mobile platform.

1. 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]. The number of mobile communicating devices one can meet in every-day life is dramatically increasing: mobile phones, PDAs, handheld GPS, laptops and notebooks, portable music and video players. Those devices benefit from an amazing number of sensors and communication interfaces. 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. To handle those systems' dynamics, cooperation-based approaches *à la* peer-to-peer seem attractive. 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 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 paper answers 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 original services: a *Proximity Map*, a *Trust and Cooperation Oracle*, and a *Cooperative Data Backup*.

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 2. 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 3. Finally, we give some further trails of research in Section 4.

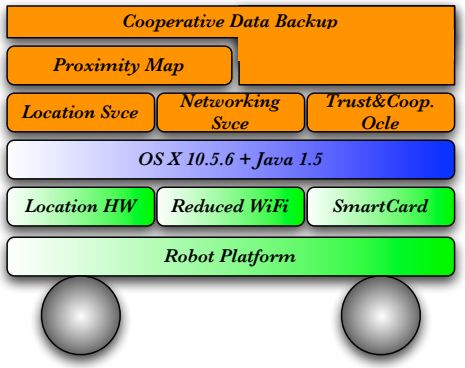

Figure 1. Overall Architecture

2. 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 3. 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:

- 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.
- 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.
- 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.

2.1. 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.

2.2. Localization and Proximity Map

For many applications for 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 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.This graph is regularly updated when the node receives a beacon and is also regularly sent to the node's neighbors in its beacons.

2.3. 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)
```

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.

2.4. 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 for the stream.
- 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 backup thread runs periodically. It 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] and [7].

When the client wants to restore data, it can either submit the unique identifier of the stream to the asyn-

chronous 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).

2.5. 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.

3. 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 environment is available.

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 $10 - 40cm$.

Technological aspects. To reach such a level of accuracy for indoor positioning, we used a dedicated motion capture technology that tracks objects based on real-time analysis of images captured by infra-red cameras. The Cortex system is able to localize objects at the millimeter scale, which is more than enough for the VANET setting.

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.

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. We tried to characterize the relationship between all these parameters but had to fall back to an empirical method to choose the appropriate attenuators. For example, for the Distributed Black-Box case we used 26DB attenuators.

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. 2. We currently own four fully equipped robots. We were thus able to emulate the Distributed Black-Box in a setting with three cooperating cars and a police coming after an accident has taken place. During the first part of the scenario, the three cars backup the 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 ID 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 of this scenario is available at *http://theresumeexperience.blogspot.com/* and will be shown at DSN'2009. The software described in this paper should be released under a GPL license soon, a link will be posted on the same blog.

Figure 2. The ARUM platform

4. 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, 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. This platform is able to emulate mobile systems of variable size.

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 the existing mobility models for ad hoc networks do not provide realistic movement scenarios [17]. 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.

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 next version will enable the setup of virtual tracks and allow for the differentiation of the various robots: i.e. each robot will be 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. deliverable nr. d1.2," LAAS-CNRS, Contract Report nr. 07456, September 2007.

[3] A. Casimiro(editor), "Hidenets resilient architecture (final version). deliverable nr. d2.1.2," LAAS-CNRS, Contract Report nr. 08068, 2008.

[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] M. Musolesi and C. Mascolo, "Mobility Models for Systems Evaluation. A Survey," in *Middleware for Network Eccentric and Mobile Applications. State of the Art*. Springer, to appear.