Designing a Context-aware Middleware for Asynchronous Communication in Mobile Ad Hoc Environments

Mirco Musolesi
Department of Computer Science, University College London
Gower Street, London, WC1E 6BT, United Kingdom
m.musolesi@cs.ucl.ac.uk

ABSTRACT

In mobile ad hoc networks, synchronous communication mechanisms appear to be not suitable, since these assume that the involved entities are present at the same time during the interactions. On the other hand, asynchronous mechanisms seem to be better suited for mobile environments characterised with frequently intermittent connectivity. Therefore, message oriented middleware based on the asynchronous communication paradigm seems a reasonable choice in these environments.

This paper outlines the expected research contribution of my PhD work in the area of the message oriented middleware for mobile ad hoc environments. The key aspect of my approach is the exploitation of context information to enable communication also in extreme scenarios, where topology reconfigurations are frequent and network partitions are the norm rather than the exception, in an efficient way in terms of the use of the possibly scarce resources of mobile devices.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems—*Distributed Applications*; C.2.1 [**Network Architecture and Design**]: Wireless Communication

General Terms

DESIGN, ALGORITHMS

Keywords

Message oriented middleware, middleware for mobile computing, epidemic protocol, mobile ad hoc networks, context-awareness

1. BACKGROUND AND MOTIVATION

The recent progresses in wireless communication technologies pose new questions and challenging problems to computer scientists and engineers. While studies in networked fixed systems have reached sufficient maturity, in wireless and especially in mobile ad hoc environments, where no backbone network is assumed, many issues and questions are still open because of the recent establishment of this research area and its intrinsic complexity. However, for the same reasons, the design of systems for mobile scenarios is, at the same time, challenging and stimulating, since you have to cope with an extreme setting characterised by high degrees of changeability and unpredictability

Developing applications for mobile computing is extremely difficult and challenging, since designers have to deal with many aspects such as unpredictable disconnections and topology variability. Mobile networks vary very widely in their characteristics. There are more conservative types of mobile networks, in which mobile nodes are at the edges of the network, consuming services offered by a wired backbone network. These kinds of networks are often referred to as *wireless infrastructure-based networks*. At the other end of the range, *mobile ad hoc networks* are usually completely wireless and unstructured. Mobile nodes are able to roam as needed. No backbone network is assumed and services can only be offered by other accessible mobile nodes that may act also as routers of messages for the other hosts. In between these two types of networks, there is a number of possible heterogeneous combinations, hybrid systems, where communication is based both on ad hoc and infrastructure based technologies. These systems will be very common in the next years with the integration of the different existing network infrastructure and use of different interfaces.

In systems that rely on fixed networks, middleware platforms are widely used, since they provide a higher level of abstraction hiding the heterogeneity and the complexity due to the distribution of software components. However, it is not possible to use these systems directly in mobile environments for many reasons, such as the computational capabilities that are necessary to run these platforms. Therefore, middleware platforms targeting specifically mobile computing systems have been designed [13]

A number of types of middleware for fixed networks rely on synchronous communication mechanisms; in other words, the interaction primitives, such as remote procedure calls, assume a constant and reliable connection between the distributed components. Disconnections are considered as failures and not as possible events to be dealt with as normal behaviour of the system.

In mobile environments (especially in the case of the most extreme scenarios), synchronous communication mechanisms appear to be not suitable, since these assume that the in-

1st International Middleware Doctoral Symposium Toronto, Canada

volved entities are present at the same time during the interactions. In other words, they consider disconnections as failures, whereas these may be the norm rather the exception. On the other hand, asynchronous mechanisms seem to be better suited for mobile environments characterised with frequently intermittent connectivity.

2. THE STATE OF THE ART

An entire category of middleware systems, the so-called Message-Oriented Middleware (MOM), is based on a pure asynchronous communication based architecture. Some interesting solutions have been presented, both in academia (i.e., Mobile JMS [23], STEAM [14], Pronto MobileGateway [24] and REBECA [6]) and in industry (i.e., WebSphere MQ EveryPlace [8] and Software iBus Mobile [12]).

It is possible to outline some common characteristics of these systems. A large part of these systems targets infrastructure based mobile scenarios with some remarkable exceptions such as STEAM and Mobile JMS. However, in general, these systems are not able to deal with the delivery of messages to hosts that are not present in the cloud of the sender when the message is sent. In STEAM, for example, the communication is limited to the hosts that are in the same radio range. Moreover, the systems are not context-aware, that is context information is not used for the optimisation of the performance of these platforms.

A foundamental aspect of the design of a message oriented middleware is the definition of an efficient message delivery mechanism. The problem of the design of routing algorithms is really challenging in mobile ad hoc networks due to the characteristics of this settings such as intermittent connectivity, frequent topology reconfigurations and limited bandwidth.

In general, it is possible to classify routing protocols according to the assumed underlying network topology in two classes, synchronous and asynchronous. With the term synchronous protocols we refer to protocols that are based on the assumption that a connected path exists between the hosts in order to communicate. Many routing protocols have been developed in the recent years, such as DSDV [18] and AODV [19].

Viceversa, with the term asynchronous protocols we indicate the class of protocols that assumes that a connected path between these hosts may not exist when the routing decisions are made. Examples of asynchronous protocols presented in literature are the epidemic routing protocol by Vahdat and Becker [21], the Disconnected Transitive Communication paradigm by Chen and Murphy [2], the Asynchronous Probabilistic Protocol by Lindgren et alii [11] and the Delay Tolerant Routing Protocol proposed by Fall [5].

It is important to note that only the approach proposed by Chen and Murphy, a refinement of the epidemic model by Vahdat and Becker, is based on the evaluation of context information. However, the authors provide a general framework rather than a detailed instantiation, and so it can be considered only a generic conceptual model and not a completely specified working protocol.

3. EXPECTED CONTRIBUTION

The aim of my PhD work is to design and implement a context-aware message oriented middleware platform for mobile ad hoc networks[1]. I believe that it is possible to use the information deriving from the context (such as the change degree of connectivity of hosts, the tracking of the variations of the network topology, etc.) to improve the efficiency of asynchronous communication mechanisms in ad hoc networks in terms of the use of the resources, in order to achieve a reliable message delivery process. More specifically, my contribute will be the definition of a middleware characterised by mechanisms that enable an efficient asynchronous delivery of messages also in presence of intermittently disconnected clouds of hosts in the systems, by evaluating the available context information and predicting the evolution of the mobile scenario. In other words, we agree with the current research trend which states that a cross-layering could be an effective way of designing more efficient middleware for mobile computing [3]. In fact, I believe that the middleware platform should adapt his behaviour in an autonomic way, like the autonomous system of human beings, to the changing context. In other words, middleware must be based on transparent mechanisms to deal with the variability of mobile environments and to guarantee, at the same time, the best possible degree of reliability and efficiency way in terms of the use of the possibly scarce available resources.

3.1 Epidemic Messaging Middleware for Ad hoc networks (EMMA)

The first result of my investigation was the design of EMMA (Epidemic Messaging Middleware for Ad hoc networks) [17], an implementation and adaptation for mobile ad hoc networks of JMS (Java Messaging Service) [7]. The key aspect of EMMA is the persistence mechanism based on an epidemic protocol that ensure the dissemination of the information also in intermittently disconnected portions of the network but that is not optimised in terms of number of replicas by definition. Moreover, I believe that the use of a popular interface facilitates the development process of application environments.

During the design of EMMA, I investigated the foundamental issues related to the adaptation of the semantics of publish/subscribe systems to mobile ad hoc environments and find out the necessary improvements in terms of efficiency.

3.1.1 Overview

Java Messaging Service (JMS) [7] is a collection of interfaces for the asynchronous communication between distributed components. It provides a common way for Java programs to create, send and receive messages. JMS users are usually referred to as clients. Moreover, the JMS specification defines providers, as the components in charge of implementing the messaging system and provide administrative and control functionality (i.e., persistence and reliability) required by the system. Clients can send and receive messages, asynchronously, through the JMS provider, which is in charge of the delivery and, if necessary, of the persistence of the messages.

There are two supported types of communication : the so-called point to point and publish-subscribe models. The point

[1]I am designing a middleware based on mobile ad hoc networks, since this deployment scenario represents a sort of limit case. I plan to extend my investigation to the case of systems based on hybrid networks in the future.

to point model is a one to one communication mechanism: hosts send messages to queues. Receivers can be registered with some specific queues and can asynchronously pick up the messages and then acknowledge them. More than one receiver could be registered with a queue; however, a message is consumed by only one subscriber.

The publish-subscribe model is based on the use of topics that can be subscribed to by clients. Messages are sent to topics by other clients. The messages are then received in an asynchronous mode by all the subscribed clients. Clients learn about the available topics and queues through Java Naming and Directory Interface (JNDI) [20]. Queues and topics are created by an administrator on the provider and are registered with the JNDI interface for look-up. The JMS specification is supported and implemented by most part of the middleware platforms vendors.

While the JMS specification has been extensively implemented and used in traditional distributed systems, adaptations to mobile environments have been proposed only recently [23, 14]. The challenges of porting JMS to mobile settings are considerable; however, the advantages are in terms of adaptation of existing applications to mobile, and integration of mobile groups with wired networks running the same middleware.

Current prototypes target mainly nomadic mobile settings; if, however, JMS is to be adapted to completely ad hoc environments, where no fixed infrastructure is available, and where nodes come and go very dynamically, more issues must be taken into consideration.

I designed an adaptation of both the two models (point to point and the publish/subscribe) that are described in the JMS specification [7]. More details can be found in [17].

In EMMA the message delivery is based on a typical pure epidemic-style routing protocol [21]. A message that needs to be sent is replicated on each host in reach. In this way, copies of the messages are quickly spread through connected networks, like an infection. If a host becomes connected to another cloud of mobile nodes, during its movement, the message spreads through this collection of hosts. Epidemic-style replication of data and messages has been exploited in the past in many fields starting with the distributed database systems area [4].

3.1.2 Evaluation

I am simulating the platform using OMNet++ [22]. In terms of scalability of the approach, the simulation results show that the performance provided by the platform in terms of delivery ratio and delay of persistent messages and messages with higher priorities is good. However, it is worth noting that, in general, it is quite difficult to offer high degree of scalability in peer-to-peer middleware for mobile computing due to the characteristics of the devices (limited memory to store temporarily messages) and the number of possible interconnections in ad hoc settings. Nevertheless, the number of nodes of many potential application scenarios that could be envisaged, is quite limited due to the intrinsic communication patterns and organisational boundaries. Moreover, it is worth noting that the dimension of the buffer may be chosen both in accordance with the application requirements and considering the resources of the devices.

In order to improve the reliability of EMMA, a mechanism for the intelligent replication of queues and topics based on the context information must be developed.

Furthermore, with respect to the efficiency of the platform in terms of the use of resources, I plan to substitute the epidemic protocol used in the current implementation of EMMA with the Context-Aware Routing (CAR) protocol that I developed. CAR is a highly optimised and efficient protocol in terms of number of message replicas present at the same time in the system. I will present this protocol in the next section.

3.2 Context-aware Routing (CAR) Protocol For Message Delivery

As discussed before, in order to improve the efficiency of the platform, I developed a novel routing protocol for the delivery of messages, the Context-Aware Routing (CAR) protocol [16], that integrates synchronous and asynchronous mechanisms for the message delivery. In fact, if the synchronous delivery of the message is not possible since the receiver is not in the same connected portion of the mobile network, instead of replicating the message on all the neighbours, the message is sent to a host (or a set of hosts) characterised by the highest probability of getting in reach of the recipient. In other words, this host (or this set of hosts) acts as message carriers. This process is based on the evaluation and on the prediction of the context information using time series analysis techniques.

As discussed previously, although this problem has been known for some time, only a small number of protocols have been proposed in the field of asynchronous communication for ad hoc networks [21, 2]. The difficulty in producing a protocol for sending asynchronous messages derives from the simple question of determining the best location in which to place each message. Clearly, as a result of the above argument, leaving the message with the sender is not appropriate, since sender and receiver may never meet. An alternate, if inefficient, solution is to spread the messages to all hosts using a form of persistent flooding (i.e., an epidemic-style routing [21]).

In fact, epidemic routing is a reasonable approach where no information can be determined about the movement patterns of nodes in the system. In other words, where there is no basis on which to distinguish the movement pattern of any node from another, and the movement pattern of each node is individually random, the only choices about message placement are to place messages randomly or to place them everywhere, since there is no more intelligent basis for making a decision.

Unsurprisingly, as observed, there is significant problem with scalability for this approach, since the assumptions on which it is based are overly pessimistic. However, in reality, such situations are only likely to occur in simulated environments, where nodes are using simplistic movement models.

The CAR protocol, instead, aims at determining which hosts are most likely successfully to carry the message to the cloud in which the destination resides. Determining which of a number of hosts is likely to be the best requires a multi-dimensional metric: mobility rate, mobility patterns, and amount of battery energy remaining are only three of a possibly large number of factors that go to make up information about the context of each node.

3.2.1 Overview

I now give a concise overview of the protocol. More details can be found in [16]. The CAR algorithm is built on the

assumption that the only information a host has about its position is logical connectivity information. In particular, I assume that a host is not aware of its absolute geographical location nor of the location of those to whom it might deliver the message. Although this information could potentially be useful, and, indeed, I plan to examine its utility in the near future, it is currently unreasonable to assume the existence of GPS for all potential application domains for this technology. Another basic assumption is that the hosts present in the system cooperate to deliver the message. In other words, I do not consider the case of hosts that may refuse to deliver a message or that act in a Byzantine manner.

The delivery process depends on whether or not the recipient is present in the same cloud as the sender. If both are currently in the same connected portion of the network, the message is delivered directly, using the underlying routing protocol to determine a forwarding path. In my current work, I assume that a proactive routing protocol is used (in my simulations I employed DSDV [18]).

If a message cannot be delivered synchronously[2], the best carriers for a message are those that have the highest chance of successful delivery, i.e., the highest delivery probabilities. The message is sent to one or more of these hosts using the underlying synchronous mechanism. Delivery probabilities are synthesized locally from context information such as the rate of change of connectivity of a host (i.e., the likelihood of it meeting other hosts) and its current energy level (i.e., the likelihood of it remaining alive to deliver the message). We define context as the set of attributes that describe the aspects of the system that can be used to optimize the process of message delivery.

Since I assume a proactive routing protocol, every host periodically sends both the information related to the underlying synchronous routing (in DSDV this is the routing tables with distances, next hop host identifier, etc.), and a list containing its delivery probabilities to the other hosts. When a host receives this information, it updates its routing tables. Maintenance of the routing table for synchronous routing simply follows the specification of the algorithm. With respect to the table for asynchronous routing, each host maintains a list of entries, each of which is a tuple that includes the fields (destination, bestHost, deliveryProbability). In the current implementation, each message is placed with only a single carrier rather than with a set, with the consequence that there is only a single list entry for each destination.

The general problem from the point of view of the sender of a message is to find the host with the best delivery probability, as calculated using the predicted values of a range of context attributes. Instead of using the available context information as it is, CAR is optimized by using predicted future values for the context, so to have a more realistic value of the context. The process of prediction and evaluation of the context information can be summarized as follows.

Each host calculates its delivery probabilities. This process is based on the prediction of the future values of the attributes describing the context and on the composition of these estimated values using multi-attribute utility theory [10]. The calculated delivery probabilities are periodically sent to the other hosts in the connected cloud as part of the update of routing information. Each host maintains a logical forwarding table of tuples describing the next logical hop, and its associated delivery probability, for all known destinations. Each host uses local prediction of delivery probabilities between updates of information. The prediction process is used during temporary disconnections and it is carried out until it is possible to guarantee a certain accuracy. Morever, in the case of hosts within reach, the interval between update shipment is based on the evaluation of the accuracy of others' prediction. In other words, hosts send updates only when the predictions others will make about their state become inaccurate. This is done by evaluating the current trend of the sampled values of context information.

More specifically, the CAR approach to this problem is based on Kalman filter prediction techniques [9], originally developed in automatic control systems theory. These are essentially discrete signal processing methods that provides optimal estimates of the current state of a dynamic system described by a state vector. The state is updated using periodic observations of the system if available using a set of prediction recursive equations. In the model used in CAR, the observed values are the current information that can be retrieved from the context.

3.2.2 Evaluation

I am evaluating the CAR protocol using the OMNET++ discrete event simulator [22]. This environment offers broad functionality that facilitates the development and the optimisation of the simulation code. Moreover, I designed a new mobility model to evaluate the performance of our protocol in different mobile scenarios randomly generated using different seeds.

I compared it with flooding and the epidemic protocol. CAR achieves a performance between that of flooding and epidemic routing, as expected. Flooding suffers from the inability to deliver messages to recipients that are in other clouds when the messages are sent. The epidemic protocol offers the best delivery ratio simply because the message is propagated to all hosts, all of which have buffers large enough to hold it. Similarly, as expected, the simulation results show substantial degradation of the performance as buffer size decreases; however, this phenomenon is more evident in the epidemic approach as a result of degree of replication of messages.

Finally, it is worth noting that in the current implementation of CAR, there is only ever a single copy of each message, which represents the worst case for this protocol; a small amount of intelligent replication would help the delivery ratio at the cost of greater overhead.

4. CURRENT RESEARCH DIRECTIONS

I am currently integrating the CAR protocol in EMMA. Other issues that I am analysing include the definition of intelligent replication mechanisms of messages and queues to improve the reliability of the platform.

I also started some evaluation of the middleware using simulations to study scenarios composed of a large number of mobile hosts. In particular, I am investigating the rela-

[2]It is worth noting that the recipient may be in the same cloud but not reachable using synchronous routing, since the routing information is not available (for example because the space in the routing tables is not sufficient to store the information related to all the hosts in the cloud or because the node has just joined the cloud). In these cases I exploit the asynchronous mechanisms.

tionships between the mobility patterns of the hosts and the system performance. For this reason, I am also studying the general problem of testing mobile middleware and the importance of meaningful mobility models for the testing [15]. Many mobility models have been presented for the testing of protocols and algorithms of mobile ad hoc networks [1]. It is interesting and, at the same time, surprising to note that even the best solutions and approaches have only been tested using completely random models such as the Random Way-Point model, without grouping mechanisms. These models cannot be clearly used to test a middleware platform based on the evaluation of the behaviour of mobile hosts.

The first results suggest that the use of CAR guarantees a remarkable efficiency in terms of the use of resources. At the same time, the obtained performance with one single copy of the message are good in terms of delivery ratio in comparison with the epidemic protocol, as described in [16]. However, this is a typical case of trade-off between the use of resources and the consequent reliability of the system. In fact, the epidemic protocol provides the best performance in terms of delivery ratio and average delay, but, at the same time the worst ones in terms of the use of resources. I am currently investigating the definition of message replication mechanisms, considering this trade-off between the number of replicas (i.e, the use of resources) and the system reliability.

An important issue to be explored is the definition of a set of new primitives to extend the semantics of JMS in order to add, for example, the concept of location and groups of mobile users. Likewise, I am going to define a new syntax for the topic subscriptions to include, for instance, the definition of geographical position and group affiliation.

Another possible research development is also the introduction of more mobility oriented communication extensions, for instance the support of geocast (i.e., the ability to send messages to specific geographical areas).

To conclude, the adaptation of JMS represents the initial step towards the definition of a new middleware interface characterised by a set of new primitives and based on the evolution of the architecture defined by the JMS specification.

5. REFERENCES

[1] T. Camp, J. Boleng, and V. Davies. A Survey of Mobility Models for Ad Hoc Network Research. *Wireless Communication and Mobile Computing (WCMC): Special Issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, 2(5):483–502, 2002.

[2] X. Chen and A. L. Murphy. Enabling disconnected transitive communication in mobile ad hoc networks. In *Proceedings of the Workshop on Principles of Mobile Computing (colocated with PODC'01)*, pages 21–23, August 2001.

[3] M. Conti, G. Maselli, G. Turi, and S. Giordano. Cross-layering in Mobile ad Hoc Network Design. *IEEE Computer*, 37(2):48–51, February 2004.

[4] A. Demers, D. Greene, C. Hauser, W. Irish, J. Larson, S. Shenker, H. Sturgis, D. Swinehart, and D. Terry. Epidemic Algorithms for Replicated Database Maintenance. In *Sixth Symposium on Principles of Distributed Computing*, pages 1–12, August 1987.

[5] K. Fall. A delay-tolerant network architecture for challenged internets. In *Proceedings of SIGCOMM'03*, August 2003.

[6] L. Fiege, F. C. Gartner, O. Kasten, and A. Zeidler. Supporting mobility in content-based publish/subscribe middleware. In *Proceedings of Middleware 2003 ACM/IFIP/USENIX International Middleware Conference*, June 2003.

[7] M. Hapner, R. Burridge, R. Sharma, J. Fialli, and K. Stout. *Java Message Service Specification Version 1.1*. Sun Microsystems, Inc., April 2002. http://java.sun.com/products/jms/.

[8] IBM. *WebSphere MQ EveryPlace Version 2.0*, November 2002.

[9] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME Journal of Basic Engineering*, March 1960.

[10] R. L. Keeney and H. Raiffa. *Decisions with Multiple Objectives: Preference and Value Tradeoffs*. Wiley Series in Probability and Mathematical Statistics. John Wiley and Sons, 1976.

[11] A. Lindgren, A. Doria, and O. Schelen. Probabilistic routing in intermittently connected networks. *Mobile Computing and Communications Review*, 7(3), July 2003.

[12] S. Maffeis. *Introducing Wireless JMS*. Softwired AG, www.sofwired-inc.com, 2002.

[13] C. Mascolo, L. Capra, and W. Emmerich. Middleware for mobile computing (a survey). In *Networking 2002 Tutorial Papers*, 2497, pages 20–58. E. Gregori and G. Anastasi and S. Basagni, 2002.

[14] R. Meier and V. Cahill. STEAM: Event-Based Middleware for Wireless Ad Hoc Networks. In *22nd International Conference on Distributed Computing Systems Workshops (ICDCSW '02)*, pages 639–644, June 2002.

[15] M. Musolesi, S. Hailes, and C. Mascolo. An ad hoc mobility model founded on social network theory. In *Proceedings of the 7th ACM/IEEE International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM 2004)*, October 2004.

[16] M. Musolesi, S. Hailes, and C. Mascolo. Adaptive routing for intermittently connected mobile ad hoc networks. Technical report, UCL-CS Research Note, July 2004. Submitted for Publication.

[17] M. Musolesi, C. Mascolo, and S. Hailes. Adapting asynchronous messaging middleware to ad hoc networking. In *Proceedings of 2nd International Workshop on Middleware for Pervasive and Ad-Hoc Computing (co-located with Middleware 2004)*, Toronto, Canada, October 2004. ACM Press.

[18] C. E. Perkins and P. Bhagwat. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. In *Proceedings of thr Conference on Communications Architecture, Protocols and Applications (SIGCOMM 94)*, pages 234–244, August 1994.

[19] C. E. Perkins and E. M. Royer. Ad hoc on-demand distance vector routing. In *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, February 1999.

[20] Sun Microsystems. Java Naming and Directory Interface (JNDI) Documentation Version 1.2. 2003. http://java.sun.com/products/jndi/.

[21] A. Vahdat and D. Becker. Epidemic Routing for Partially Connected Ad Hoc Networks. Technical Report CS-2000-06, Department of Computer Science, Duke University, 2000.

[22] A. Vargas. The OMNeT++ discrete event simulation system. In *Proceedings of the European Simulation Multiconference (ESM'2001)*, Prague, June 2001.

[23] E. Vollset, D. Ingham, and P. Ezhilchelvan. JMS on mobile ad hoc networks. In *Personal Wireless Communications 2003*. Springer-Verlag, September 2003.

[24] E. Yoneki. Pronto: MobileGateway with Publish-Subscribe Paradigm over Wireless Networks. In *4th ACM/IFIP/USENIX International Conference on Middleware (Middleware'03 - Work in Progress)*, number 4(5), IEEE Distributed Systems Online 2003.