USING GROUP MANAGEMENT TO TAME MOBILE AD HOC NETWORKS

Malika Boulkenafed[1], Daniele Sacchetti[1], Valerie Issarny[1]

[1]INRIA-*Rocquencourt Domaine de Voluceau,*
Rocquencourt, BP 105,
78153 Le Chesnay Cedex,
FRANCE
{Malika.Boulkenafed, Daniele.Sacchetti, Valerie.Issarny@inria}@inria.fr

Abstract Mobile ad hoc networks (MANET) offer a convenient basis towards pervasive computing, due to inherent support for anytime, anywhere network access for mobile users. However, the development of applications over MANET still raises numerous challenges. One such challenge relates to accommodating the high dynamics of the network's topology. Group management appears as a promising paradigm to ease the development of distributed applications over dynamic, mobile networks. Specifically, group management takes care of assembling mobile nodes that together allow to meet target functional and non-functional properties, and of further making transparent failures due to the mobility of nodes. Various solutions towards group management over MANET have been investigated over the last couple of years, each targeting specific applications. Building upon such an effort, this paper introduces the design and implementation of a group service for MANET, which is generic with respect to the various attributes of relevance. Generic group management allows supporting various applications, as illustrated through groups dedicated to mobile collaborative data sharing.

Keywords: Ad hoc networking, group management, mobile environments, resource sharing.

Introduction

Mobile ad hoc networks (MANET) pave the way for pervasive computing due to inherent support for anytime, anywhere network access for mobile users. Nonetheless, the highly dynamic nature of mobile ad hoc networks poses tremendous challenges for the development of applications since the application's context keeps changing over time. One approach to master this complexity lies in the management of groups over MANET, i.e., applications execute on top of groups that manage the dynamic execution context, including mobility-related failures. There has been extensive research on group man-

agement and related group communication services in the context of fixed networks, with special emphasis on providing availability properties [16, 7]. However, proposed solutions cannot be applied directly to mobile wireless networks due to the network's highly dynamic topology [2]. This has led to adapt the management of group membership to the specifics of MANET.

Group membership is primarily defined according to the functional property to be achieved by the group, e.g., collaborative editing, sharing a computational load, increasing performance, providing fault tolerant service. In general, a member may leave a group because it failed, explicitly requested to leave, or is expelled by other members. Similarly, a member may join a group because it explicitly requests it or recovers from failure. A group membership protocol must manage such dynamic changes in a coherent way, i.e., all members of the group must have a consistent view of the group's membership despite failures [7]. The highly dynamic topology of MANET introduces additional complexity in the management of group membership because connections may be transient and partitioned networks may never be rejoined together. However, group membership for MANET may still be defined as for fixed networks, i.e., according to the functional property to be realized. In this case, it is considered that the MANET allows restoring lost connections using the underlying routing protocol. Solutions then lie in the adaptation of the group communication service to the dynamic topology of the network [11]. Such an approach does not adapt the system's functions to the specifics of the network but rather adapt the implementation of traditional distributed system functions. However, it is advantageous to revise the definition of group membership so as to integrate the connectivity dimension in addition to the functional one. This allows dealing with quality of service requirements by bounding communication latency [8] and/or supporting location-aware applications. Specifically, connectivity-constrained group membership enables managing a dynamic (sub-)network that is configured according to both connectivity constraints and the functional property to be implemented, while hiding mobility-induced link failures to applications [17]. Connectivity constraints may vary from 1-hop to multi-hop connectivity, where unbounded multi-hop connectivity corresponds to the aforementioned connectivity-unaware group membership addressed in [11]. Connectivity constraints may then be fixed according to the network's connectivity (i.e., number of hops) as in [17] or the respective geographical position of the group's members as in [14]. The definition of group membership may further be extended with integrity constraints (e.g., security constraints, size) [15].

In this paper, we provide a characterization of the attributes of group membership in ad hoc networks (Section 1), and then introduce the design of a group management service that is generic with respect to the elicited attributes (Section 2). We further assess the proposed service based on its theoritical

complexity, its implementation in a middleware aimed at mobile computing (Section 3), and its usage for enabling pervasive computing scenarios relating to mobile collaboration and data management (Section 4). Finally, we conclude with a summary of our contribution and our future work (Section 5).

1. Attributes of Group Membership

We recall that groups are first defined with respect to a given functional property. We denote such a property by f. Without loss of generality, we assume that the property is offered by any node, as opposed to being an aggregation of some functions provided by grouped nodes. We use the Boolean function $support(x, f)$ to denote that node x offers function f. Note that f may characterize various features supported by nodes, and resembles to resources considered in resource discovery protocols [3]. Finally, we denote by G^f, a group realizing function f.

Network Model. We consider a WI-FI-based ad hoc network consisting of a set N of n nodes, and assume that every node x of N has a unique identifier $Id(x)$. However, we do not fix the routing protocol that is used. We further introduce the following functions to reason about the connectivity of nodes, for $x \in N$ and a time period T that is such that the network does not change over T.

- $Proximity(T, x, p)$ returns the geographical distance in meters between the location of x and geographical position p, during T.
- $Dist(T, x, y)$ returns the geographical distance in meters between the respective locations of x and $y \in N$, during T.
- $Connectivity(T, x)$ returns the set of all nodes of N with which x can communicate using the underlying network protocols, during T; note that due to the asymmetric nature of wireless networks in general, $y \in Connectivity(T, x)$ does not imply $x \in Connectivity(T, y)$.
- $DualConnectivity(T, x)$ returns the set of all nodes y of N such that $y \in Connectivity(T, x)$ and $x \in Connectivity(T, y)$.
- $Hops(T, x, y)$ returns the number of hops for communication between x and y for any y belonging to $Connectivity(T, x)$.

Location. We now define functions characterizing group membership with respect to constraints set on the relative location of member nodes.

- *Location-unaware* groups as, e.g., addressed in [11, 12], are defined solely with respect to the functional properties offered by the group members. Hence, $LocationUnaware(G^f)$ always holds.

- *Proximity-based* groups as, e.g., addressed in [14, 15], set that group members should be in a given geographical area, whose location may be fixed a priori or set relative to the position of group members. Let pos denote a referenced geographical position and $dist$ let denote the maximal geographical distance that is allowed, we get:

$$Geo\_Prox(G^f, T, pos, dist) \Leftrightarrow \forall x \in G^f, Prox(T, x, pos) < dist$$
$$Relative\_Prox(G^f, T, dist) \Leftrightarrow \forall (x, y) \in G^{f2}, Dist(T, x, y) < dist$$

- *Bounded* groups are defined with respect to the number of hops separating node members as, e.g., addressed in [17, 18, 6], are defined in a similar way based on the maximal number of hops, noted $hops$, between nodes:

$$Bounded(G^f, T, hops) \Leftrightarrow \forall (x, y) \in G^{f2}, Hops(T, x, y) \leq hops$$

Note that the above functions are not exclusive of each other and may be combined for the definition of a given group.

Openness. Group membership may be restricted to authorized nodes. We model such a constraint using the notion of *security domain*: a security domain S^f sets nodes of N that trust each other towards realizing function f. Practically, a security domain is managed by a trusted third party to which nodes may authenticate and register themselves; nodes then get a signed certificate that they may use to authenticate themselves with other nodes belonging to S^f. Secure group communication may further be enforced through the implementation of group key agreement within the group [4]. We get:

$$Closed(G^f, S^f) \Leftrightarrow \forall x \in G^f \Rightarrow x \in S^f$$

Connectivity. Group membership may require full, partial or even loose connectivity among nodes. In general, connectivity constraints may be combined with any of the aforementioned location-related constraints and may apply to both open and closed groups. Loose connectivity consists of relying on the connectivity enabled by the underlying network over time and thus does not impose any specific constraint. A fully connected group is further characterized by:

$$Connected(G^f, T) \Leftrightarrow \forall (x, y) \in G^{f2}, y \in DualConnectivity(T, x)$$

Partial connectivity is defined according to the client and server roles of nodes with respect to function f. We use the function $client(x, f)$, resp. $server(x, f)$, to denote that x is client of f, resp. server of f. We get:

$$Partial(G^f, T) \Leftrightarrow \forall (x, y) \in G^{f2}, server(x, f) \Rightarrow y \in DualConnectivity(T, x)$$

Note that $Connected(G^f) \Rightarrow Partial(G^f)$. Also, symmetric communication links may not be required between client and server nodes depending on the

interaction patterns required by the application. However, we consider dual connectivity only, as this is the most common case for applications. The definition of partial connectivity with uni-directional reachability is further direct to infer. Finally, we enforce full connectivity among server nodes.

QoS Awareness. Group membership may further be constrained for the sake of enhanced quality of service, i.e., members of the group must meet a number of Quality of Service (QoS) attributes. Various QoS attributes may be considered. In particular, the following attributes appear to be the most dominant in the context of MANET [10]: reliability, security, performance and transactional behavior that relate to service-level attributes, and CPU load, memory, bandwidth and battery that relate to resource-level attributes. Then, a QoS-aware group may restrict group membership to nodes meeting the QoS attributes that are fixed for the group among the above. In addition, group membership may be constrained so as to limit the probability of a node leaving a group. For instance, exploiting the movement of nodes has been suggested as an additional criterion for integration within a group [17]. In general, disconnection of a node from a group may be due to the node's mobility and/or the node's resource scarcity. The former may be anticipated based on information on the node's movement, and the latter may be anticipated based on information about resource-level QoS attributes of the node [6].

2. Group Service Design

This section details the design of a group management service that is generic with respect to the above membership attributes. We further consider the following requirements for the service design: (i) The group management service must minimize resource consumption on mobile nodes, and in particular energy, requiring minimizing message exchanges. (ii) Group management cannot accommodate a centralized solution where a single node is responsible for managing the group, since the node may leave the group at any time. It is thus necessary to provide a decentralized solution. (iii) The group management service must mask the highly dynamic topology of the network to the application, requiring updating group membership accordingly. We distinguish three functions in group management:

- Discovering group members, i.e., discovering mobile nodes that are eligible for membership according to relevant membership attributes, i.e., location, openness and QoS-related constraints.
- Initializing the group, i.e., exchanging meta-data relevant to the group's functionality and further checking for global membership constraints, i.e., connectivity and QoS-related attributes.
- Managing the group's dynamics, i.e., updating group membership according to the dynamics of the network's topology.

2.1 Group Members Discovery

Each node maintains the list of all groups, $\{G^{f1}, \ldots G^{fn}\}$, to which it may belong. For each group, the node periodically runs a discovery process to locate peer nodes with which it may join. The discovery process consists of broadcasting a discovery message, *Disc*, towards nodes accessible in 1 hop, and dually handling *Disc* messages that are received. A *Disc* message embeds at least the name of the group and the Id of the sender. The discovery process is further customized according to the QoS-related, openness, and location attributes set for the group.

Prior to issuing a *Disc* message, every node checks compliance with respect to relevant local QoS-related attributes. In addition, in the case where membership is restricted to authorized nodes (i.e., $Closed(G^f, S^f)$ must hold), related discovery messages embed the signed certificate of the sender node, and transmitted data (but the group's identifier) are encrypted using the sender's private key given that the sender's public key is part of the sender's certificate.

Subsequent handling of discovery messages depends on the location constraints set for the group. If the group G^f is location-unaware, peer nodes that are discovered through receipt of relevant *Disc* messages are added to the local list of peer nodes, P^f. In addition, any *Disc* message that is received, for the given period, is broadcasted so that the message is eventually (assuming every two nodes of N are connected, possibly in n hops) received by nodes that are not accessible in 1 hop from the sender. If the group is proximity-based, *Disc* messages embed the geographical position (e.g., using embedded GPS function) of the sender and their handling is constrained by the enforced geographical and/or relative proximity. In the case of geographical proximity (i.e., $Geo\_Prox(G^f, T, pos, dist)$ must be enforced), a *Disc* message is sent only if the sender node is located in the targeted region. In the same way, nodes discovered through received *Disc* messages are included in the local list of peer nodes and further forwarded, only if their position meets the geographical constraint set for the group. Relative proximity (i.e., $Relative\_Prox(G^f, T, dist)$ must be enforced) is handled similarly; only *Disc* messages received from nodes whose position meets the enforced geographical constraint are processed by the receiver. Bounded groups (i.e., $Bounded(G^f, T, hops)$ must be enforced) are handled as for proximity-based groups; the only difference is that proximity is defined with respect to the number of hops instead of geographical position. The number of hops can be determined using the TTL (Time To Live) counter.

2.2 Group Initialization

The above decentralized process allows every node to discover its peers according to local QoS-related, openness and location constraints set for the

group. However, setting the group requires further constraining group membership according to connectivity and global QoS-related constraints.

Note that decentralized management of location-unaware groups is only compliant with loose connectivity unless members of the group are fixed a priori and hence known by members prior to the discovery process. This restriction allows to bound the time taken by the discovery process. Precisely, the discovery phase is bounded according to the location constraints combined with the time taken for message exchange with the underlying network.

Once the discovery phase is terminated, the initialization phase establishes group membership so as to enforce required connectivity. This process is managed by a single node, called *leader*. The leader is a peer node whose *Id* is the greatest among all group members. Hence, based on its local list of peer nodes, P_n^f, for group G^f, every node n knows the group leader. Centralization of the initialization process via the leader allows minimizing the process' cost in terms of exchanged messages and thus of energy consumption. In addition, as detailed in the next section, the group leader is periodically changed within the group, so that the associated load is fairly distributed among nodes and the disconnection of the leader does not affect group management.

The role of the leader is first to check connectivity constraints. Thus, every node n sends its local list of peer nodes P_n^f to *its* leader l (i.e., $l \in P_n^f \; and \; \forall x \in P_n^f : Id(x) < Id(l)$) using the $Join$ message. Due to the partial connectivity inherent to wireless network, a node may be elected as leader by some nodes while not electing itself as leader. Thus, every node, even if it did not elect itself as a leader, handles incoming $Join$ messages received within a given time period Δ that is set according to connectivity constraints combined with the time taken for message exchanges.

Consider first the case of full connectivity (i.e., enforcing $Connected(G^f, T)$). Let R^f denote the list of nodes from which l received a $Join$ message over time period Δ, $G_l^f = P_l^f \bigcap R^f$, $I_l^f = \bigcap_{(j \in G_l^f)} P_j^f$ and $U_l^f = \bigcup_{(j \in G_l^f)} P_j^f$. In particular, I_l^f is the set of mobile nodes that discovered each other and meet the QoS-related, openness and location constraints of the given group. Also, note that $G_l^f \subseteq U_l^f$. Group membership is then established through comparison of the values of G_l^f, I_l^f and U_l^f. We distinguish three cases:

- If $G_l^f = I_l^f = U_l^f$, then all the peers of G_l^f have identical view on group membership. The leader then validates G_l^f as group membership, which is notified by sending the related $Group$ message to all nodes of G_l^f.
- If $(G_l^f = I_l^f \; and \; I_l^f \neq U_l^f)$ or $(G_l^f \neq I_l^f \; and \; I_l^f = U_l^f)$ then nodes belonging to U_l^f but not to G_l^f do not meet location constraints with respect to the leader. In this case, the leader validates G_l^f as the group's mem-

bership and the nodes that are excluded are notified using an exclusion message.

- If ($G_l^f = U_l^f$ *and* $I_l^f \neq U_l^f$) or ($G_l^f \neq U_l^f$ *and* $I_l^f \neq U_l^f$ *and* $G_l^f \neq I_l^f$) then nodes belonging to I_l^f meet location constraints with respect to nodes belonging to G_l^f but some nodes belonging to G_l^f do not meet such constraints. It is then up to the leader to fix group membership (e.g., validates I_l^f as group membership).

Finally, in the case where a node receives a group membership message, while it already joined another group, it is up to the node to either ignore the message or change group.

Partial connectivity (i.e., enforcing $Partial(G^f, T)$) requires ensuring that every client node is connected to every server node. We further require all the server nodes to be fully connected. The client or server role is made known to peers during the discovery process. The group membership is then established as above, except that the leader is elected among server nodes only, and that group membership is set with respect to partial connectivity requirement.

QoS-related constraints on group membership may also lead to exclude peers from the group in the case global attributes cannot be met. Currently, we set the constraint as a maximal number on group members, provided that dedicated QoS management should be implemented within the group according to the function f that is provided (e.g., see Section 4 for an example).

In the case where the group is closed, the initialization process is further complemented with a group key agreement (GKA) protocol so as to establish a shared secret among the group's members. The secret will then be used to encrypt any message subsequently exchanged within the group. There exist various such protocols in the literature for Internet-based systems. The interested reader is referred to [4] for an overview and analysis of GKA protocols aimed at groups of resource-constrained nodes that require minimizing resource consumption.

Initialization of the group additionally depends on the specific functionality of the group, possibly leading to exchange additional data among group members. Basically, relevant data are piggybacked in the *Join* message sent to the leader, which combine and forward them to group members when issuing the message validating group membership.

2.3 Managing the Group Dynamics

Group management over MANET requires to take into account the highly dynamic topology of the network, i.e., mobility-induced changes in group membership, in a way that is transparent to applications. Changes are detected during the discovery and initialization phases. However, the related processes

cannot be run continuously due to the resource consumption that they induce. We thus propose to make periodic the process of group maintenance, where the period is dynamically adapted according to the rate of changes within the group, as initially proposed in [5] and outlined below.

The period is initially set to a given value T and is then dynamically adapted according to the past behavior of the embedding group (which may be a single node in the case of a singleton group). Let t be the current time, T' and T be the last two periods, and $\Gamma = \frac{C^{prev}}{C^{last}}$ with C^{last} being the number of changes over the last period (i.e., over $[(t-T), t]$) and C^{prev} being the number of changes over $[(t-(T'+T)), (t-T)]$. Then, if Γ is greater (resp. smaller) than one, the value of T should be increased (resp. decreased) for the next period because the group has been changing less (resp. more) frequently over the last period than it changed over the previous period. The new value of T then becomes equal to: $T \times \Gamma$.

Upon expiration of period T, each node belonging to the group runs its local discovery process (§2.1) in order to detect possible changes in group membership. Due to the periodic discovery process, nodes (and even distinct groups) that may join together according to constraints set for group membership may not run the discovery process at the same time. It is then up to the groups' leaders[1] to synchronize their respective discovery process in order to join together.

The node elected as leader within a group changes periodically in accordance with period T, due to the leader's possible mobility but also to distribute the load for group management among group members. We recall that the election of the group leader is decentralized (§2.2). Precisely, the leader is elected according to the following algorithm. Assuming node n belongs to group G^f, n keeps the list L_n^f of the nodes that were elected as leaders within the group, provided that $L_n^f = \phi$ if n belongs to a singleton group. Then, n includes L_n^f in its discovery message *Disc*. After the completion of the discovery process, every node n computes L^f as the union of L_n^f with the sets L_x^f embedded in the discovery messages received from all the peer nodes x. Then, n elects as leader the node that has the maximum *Id* from $P_l^f - L^f$, which is added to L_n^f. In the case where $P_l^f - L^f = \phi$, L_n^f is set to ϕ. As a result, all nodes of a group G^f elects the same leader in a decentralized way.

Mobility of nodes (i.e., joining or leaving the group) does not affect the election of the leader since the decentralized nature of the algorithm makes it independent of the group's dynamic. In addition, the leader role is significant only during the initialization phase. Hence, the leave of the leader affects group management only if it occurs during the initialization phase, leading

[1] In the case where the group is a singleton, the single member node is the group's leader.

Table 1. Cost of group membership in terms of exchanged messages.

Node	Management Phase	Sent	Received	Discarded
Creation of a group by joining n singletons				
Leader	Discovery	1	n-1	0
	Initialization	1	n-1	0
Peer	Discovery	1	n-1	0
	Initialization	1	1	n-2
p nodes join the group				
Leader	Discovery	1	n+p-1	0
	Initialization	1	n+p-1	0
Peer	Discovery	1	n+p-1	0
	Initialization	1	1	n+p-2

to effective update of group membership at the next period. In addition, our algorithm guarantees that a single leader exists during a period.

3. Assessment

The following first provides an assessment of our group management in terms of theoritical complexity, and in terms of group dynamics impact on applications, and then discusses the implementation of the group management service on the top of the WSAMI middleware aimed at mobile distributed computing.

3.1 Theoretical Complexity

Our group service is designed so as to minimize resource consumption on nodes. In particular, the number of exchanged messages to manage group membership is minimized through the introduction of group leader. Table 1 gives the number of exchanged messages for group management, considering specifically the creation of a group of n nodes and the update of a group with p joining nodes. Precisely, we give the number of messages that are sent[2], received and discarded, given management over an one hop ad hoc network and considering only nodes of the group. The theoretical complexity of group management is in $O(n)$ with n being the number of group members. The node designated as leader sends/receives more messages than the other group members. Hence, resource consumption is larger on the leader node. In particular, induced energy consumption for the leader node is 32.5% higher than the one for peer nodes [6], which is why it is crucial to periodically change the node acting as leader.

[2]For message broadcast, we set the number of emission to 1 and the one of reception to n.

3.2 Group Dynamics Impact

Changes occurring within a group are detected every period T. Then, group membership G^f viewed by nodes may be inconsistent with actual group membership if the value of period T is greater than the period during which the network's topology does not change. This may possibly affect the application's correctness, provided that we consider that applications are designed to execute over dynamic groups. Note that we further consider that only location-based and connectivity constraints (i.e., membership constraints parameterized by time period in Section 1) can be violated. While it may be the case that the certificate of a group member may expire during period T in the case of a closed group, this case is avoided by integrating within a group, only nodes that have a certificate that is valid for the duration of whole period T. In the same way, QoS-related attributes are provided with respect to the period T.

Various cases may be considered with respect to inconsistent group membership. If the composition of G^f (as viewed by group members) still meets the group's (location-based and connectivity) constraints with respect to actual group membership then this does not impact the application's correctness; it simply means that the application misses nodes that could join the group. If the composition of G^f no longer meets the group's constraints with respect to actual group membership, then we consider two cases:(i) Required connectivity may still be established although violating location-based constraints (e.g., connectivity now requires 3 hops while the group should be limited to nodes at a distance less than 2 hops). (ii) Required connectivity cannot be established. Although the former case leads to an application that is possibly not correct with respect to some non-functional property for some period less than T (e.g., the group is bounded to fixed maximum response time), it does not affect the functional correctness of the application. On the other hand, the latter case possibly leads to message loss, and hence may affect application's correctness. Various solutions can be considered, e.g., using a gossip mechanism as in [11], for reliable group communications, leading to the former situation. However, there is no guarantee about when the message will be delivered, which may happen subsequent to the end of the current period. We thus choose a simpler solution that is to report an exception to the application.

3.3 Implementation

We are implementing a prototype of our group management service within the WSAMI[3] middleware aimed at supporting pervasive computing/ambient intelligence applications over hybrid networks, which is being developed as

[3] Web Services for Ambient Intelligence.

part of the European IST OZONE project[4]. WSAMI builds upon the Web services architecture, hence allowing to benefit from the pervasiveness of the Web for mobile applications[5]. The base WSAMI middleware comprises the WSAMI core broker, enabling interaction among mobile Web services, and the ND service for naming, discovery and lookup that allows retrieving services according to the user's situation [9].

The WSAMI core broker provides communication functionalities to distributed services, and offers a development and deployment environment to service developers. The WSAMI development and deployment environment is based on the WSDL[6] and WSAMI languages. WSDL documents define service interfaces and instances, as specified by the Web Services Architecture. WSAMI documents define service composition (i.e., required services) and quality of service requirements (i.e., non-functional requirements more specifically related to security and performance). The WSAMI ND service serves discovering instances of Web services implementing a given WSAMI interface provided the URI of the corresponding document, and offers the following functionalities: (i) the management of a repository of local service instances; (ii) the location of remote service instances, which is based on the Service Location Protocol[7] (SLP); and, (iii) the handling of connector customization for enforcing quality of service. The WSAMI customizer document associated to the service is used to discover local and remote customizer service instances.

The Group Service is then implemented on top of the ND service, which is extended to discover the peer services that meet the constraints set for a given group. Precisely, the ND service is extended with the function $DiscPeer$, which takes as input: the URI of the WSAMI document defining the function provided by the group, and additional parameters relevant to the group's constraints (i.e., certificate if closed group, and any related location and QoS data). The Group Service is deployed on any node taking part in group management, and offers a $RegGroup$ (resp. $UnRegGroup$) function for registering (resp. canceling) participation to a given group. The $RegGroup$ function takes as input the functional specification of the group (i.e., WSAMI specification) provided that a corresponding service instance is locally deployed, together with membership constraints associated with the group. In addition, any group member must support the function associated with the leader, i.e., group initialization. Group management is then initiated following call to the $Activation$ function of the group service for the given group, further leading

[4] `http://www.extra.research.philips.com/euprojects/ozone/`
[5] `http://www.w3.org/2002/ws/`
[6] `http://www.w3.org/TR/wsdl`
[7] http://www.srvloc.org/

to group creation and periodic maintenance, as detailed in the previous section, until the $Desactivate$ function is called.

4. Collaborative Data Sharing

This section discusses the implementation of a specific group, providing example of group management supporting pervasive computing scenarios. This example relates to mobile data management, where group management in MANET may be exploited to implement a shared data structure that aggregates the content made available on peer nodes, e.g., [17, 6, 13].

Group management is much suited to collaborative data sharing in the mobile environment; nodes that have access to common data may join together so as get access to related data located on peer nodes. This further supports mobile collaborative applications [1], where collaboration may be either synchronous or asynchronous depending on the degree of concurrent updates. Supporting mobile collaborative data sharing further subsumes the definition of adequate replication and coherency management protocols. In general, optimistic coherency management, where data are updated independently and updates propagated when connectivity allows, is the most appropriate for the mobile environment where disconnections are frequent. However, strong coherency management is much suited to synchronous collaborative applications and may be supported by mobile groups whose membership is constrained by the proximity of nodes (e.g., group bounded by 1 hop-connectivity used for P2P meeting-based applications) [5]. Regarding replication management, data that are accessed by a node should be replicated on that node due to possible disconnection, and preventive data replicas may further be created so as to anticipate the disconnection of a node that holds data of interest and hence increase data availability [6]. Based on the above, we define a group, called G^{Collab}, dedicated to synchronous, collaborative data sharing, which is based on the proposal of [5]. In this context, the security domain defines nodes that are granted access to a given shared data structure and is managed by the server that stores a reference copy of the data. Then, peer nodes in the communication range of each other may join within a group, instance of G^{Collab}, which supports synchronous collaboration through the implementation of strong coherency management. Asynchronous collaboration is further supported at the level of the overall security domain (i.e., distinct groups) through the implementation of optimistic coherency management.

Group design. Membership constraints (as detailed in Section 1) associated with the G^{Collab} group are then:

- $Bounded(G^{Collab}, T, 1)$, i.e., members of a group instance should be at a distance of one hop, since we primarily target meeting-based applications.

- $Closed(G^{Collab}, S^D)$, i.e., members of a group instance should be granted access to the shared data, as identified by membership to the given security domain S^D.

- $Connected(G^{Collab}, T)$, i.e., nodes within a group instance should be fully connected so as to allow P2P-based data sharing among all nodes.

The $Collab$ function relates to providing access to data that are locally stored on all peer nodes belonging to the security domain S^D. More specifically, we consider that nodes share XML data as in the XMIDDLE middleware [13]. Hence, each peer node offers the DOM-based $Collab$ service, which provides access to the local (most likely partial) copy of the XML tree that is shared by all members of S^D. Any mobile node belonging to S^D then stores locally a partial copy of the tree, according to previous access to the tree performed on the node.

Management of the G^{Collab} group allows for mobile nodes that are members of an instance of G^{Collab} to access parts of the shared tree that are either not locally stored or are not locally up-to-date, but are stored on peer nodes of the group instance. Precisely, group initialization (see §2.2) is customized so that peer nodes get aware of the overall (possibly partial) copy of the tree that is stored within the group. After peer nodes are grouped, the local XML trees get annotated with information about replicas available in the group. Note that a subtree that is not locally available is not replicated at initialization time; only information about available replicas in the group is stored. Upon access to data of the XML tree on a node, if the data is stored locally, the data is checked for coherency according to the strong coherency protocol that is implemented at the group level and then possibly updated, prior to grant access to the node. Otherwise, a local copy is obtained from the peer node that has the most recent data version, still according to the strong coherency protocol that is implemented.

Group implementation. Implementation of the above group management for synchronous, collaborative data sharing using our group service lies in the implementation of the DOM-like Web service $Collab$ and its deployment on every mobile node that is willing to participate in collaborative data sharing. Each such node must further obtain a certificate associated with the XML tree it is granted access to. As in [5], we assume that a reference copy of any shared XML tree is stored on some highly-available, secure server from which a certificate can be obtained. Note that a node may be granted access to more than one XML tree; this is distinguished by providing different names for groups, i.e., a group is identified by the supporting function ($Collab$ in our example) and unique name (e.g., name of the tree/project). The interface of the $Collab$ service is similar to the one of DOM enriched with operations dedicated to

group management according to the aforementioned membership constraints. Implementation of the service further inherits from the *Group* class, which specializes with functions dedicated to synchronous, collaborative data sharing (i.e., replication and coherency management as detailed in [5], which is adapted here to XML tree sharing). Access to a shared XML tree on a mobile node then relies on accessing the tree using the local *Collab* service instance, which transparently handles collaborative data sharing through related group management. Note that in the case where the node cannot join a group, the node is a member of a singleton group, accessing only data that are locally stored.

5. Conclusion

Group management appears as a key middleware functionality for assisting the development of applications over MANET. Group management takes care of managing a dynamic sub-network on top of which the application executes towards implementing given functional and non functional properties. Group management over MANET has actually given rise to various studies over the last couple of years, each concentrating on specific applications. However, a distinctive set of key attributes may be identified for MANET-based groups, which may further be exploited to design a generic group service that is to be customized by applications.

This paper has presented the design of such a generic group management service. In a first step, we have introduced key attributes for group management over MANET, in particular based on applications published in the literature. Those attributes amount to setting membership constraints in relation with the location, connectivity, authentication and supported QoS of group members. We then have introduced a group service that is generic with respect to membership constraints, and realizes three basic functions: discovery of group members, initialization of the group, and management of the group's dynamics. Implementation of the generic group service has further been addressed in the context of the WSAMI middleware aimed at mobile distributed computing, which is based on the Web Services Architecture. Finally, we have presented an instance of group management that builds on our generic group service and allows supporting ambient intelligence scenarios for instance related to mobile collaborative work.

Acknowledgments

This work was partially funded by IST-2000-30026 OZONE project (`http://www.extra.research.philips.com/euprojects/ozone/`) and ACI COrSS (`http://www.irit.fr/CORSS/`).

References

[1] L. Bartram and M. Blackstock. Designing portable collaborative networks. *ACM Queue*, 1(3), May 2003.

[2] C. Basile, M-O. Killijian, and D. Powell. A survey of dependability issues in mobile wireless networks. Technical report, LAAS CNRS, France, February 2003.

[3] C. Bettstetter and C. Renner. A comparison of service discovery protocols and implementation of the Service Location Protocol. In *Proceedings of 6th EUNICE Open European Summer School: Innovative Internet Applications*, 2000.

[4] R. Bhaskar. Group key agreement in ad hoc networks. Technical Report 4832, INRIA-Rocquencourt, France, 2003.

[5] M. Boulkenafed and V. Issarny. AdHocFS: Sharing files in WLANs. In *Proceedings of the 2nd IEEE International Symposium on Network Computing and Applications*, April 2003.

[6] M. Boulkenafed and V. Issarny. Middleware service for mobile ad hoc data sharing, enhancing data availability. In *Proceedings of the 4th ACM/IFIP/USENIX International Middleware Conference*, June 2003.

[7] G. Chockler, I. Keidar, and R. Vitenberg. Group communication specifications: A comprehensive study. *ACM Computing Surveys*, 33(4), December 2001.

[8] B. Hugues and V. Cahill. Towards real-time event-based communication in mobile ad hoc wireless networks. In *Proceedings of the 2nd International Workshop on Real-time LANs in the Internet Age*, July 2003.

[9] V. Issarny, D. Sacchetti, F. Tartanoglu, F. Sailhan, R. Chibout, N. Levy, and A. Talamona. Developing Ambient Intelligence Systems: A solution based on Web services. *Journal of Automated Software Engineering*, 2004, To appear.

[10] J. Liu and V. Issarny. Qos-aware service location in mobile ad hoc networks. In *Proceedings of the 5th IEEE International Conference on Mobile Data Management*, January 2004.

[11] J. Luo, P. Eugster, and J-P. Hubaux. PILOT: Probabilistic lightweight group communication system for mobile ad hoc networks. Technical Report IC/2003/35, EPFL, May 2003.

[12] J. Luo, J-P. Hubaux, and P. Eugster. PAN: Providing reliable storage in mobile ad hoc networks with probabilistic quorum systems. In *Proceedings of the 4th ACM SIGMOBILE Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2003.

[13] C. Mascolo, L. Capra, S. Zachariadis, and W. Emmerich. XMIDDLE: A data-sharing middleware for mobile computing. *Personal and Wireless Communications Journal*, 21(1), April 2002.

[14] R. Meier, M-O. Killijian, R. Cunningham, and V. Cahill. Towards proximity group communication. In *Proceedings of the 1st Workshop on Middleware for Mobile Computing (in Conjunction with Middleware'2001*, 2001.

[15] A. Meissner and S. B. Musunoori. Group integrity management support for mobile ad hoc communities. In *Proceedings of the 1st Workshop on Middleware for Pervasive and Ad Hoc Computing Computing (in Conjunction with Middleware'2003)*, 2003.

[16] D. Powell. Group communication. *CACM*, 39(4), April 1996.

[17] G-C. Roman, Q. Huang, and A. Hazemi. Consistent group membership in ad hoc networks. In *Proceedings of ICSE'2001*, 2001.

[18] S. Sha, K. Chen, and K. Nahrstedt. Dynamic bandwidth management for single-hop ad hoc wireless networks. In *Proceedings of the IEEE International Conference on Pervasive Computing (PerCom)*, 2003.