Architecture for Group Communication in Mobile Systems

Ravi Prakash*
Computer Science Program
University of Texas at Dallas
Richardson, TX 75083-0688, U.S.A.
www.utdallas.edu/~ravip

Roberto Baldoni
Dipartimento di Informatica e Sistemistica
Universita' di Roma "La Sapienza"
Via Salaria 113, I-00198 Roma, Italy.
www.dis.uniroma1.it/~baldoni

Abstract

In mobile computing systems the network configuration changes due to node mobility. This paper identifies the issues a group communication service has to take into account in order to handle node mobility. These include the need to identify the location of a node, and the ability to cope with inaccuracies in the determination of a group membership. A multi-level architecture for group communication in mobile systems is presented. This architecture contains a synchronous proximity layer protocol to determine the set of mobile nodes in the proximity of a given node in the network. This information is used by a three-round group membership protocol for construction of groups used by mobile applications. As an example, the architecture is specialized to solve the channel allocation problem.

1. Introduction

Several distributed applications require messages to be sent to a group of nodes. The set of destinations may change with time. Membership services may run on top of the network and maintain dynamically changing communication groups [1]. The application program, running atop the membership service, only has to specify the name of the group to which a message is to be delivered [2, 3]. The membership service maintains information about the nodes that belong to the group and delivers the message to them. Thus, the application is shielded from changes in groups. This promotes modular design of distributed applications [2].

It has traditionally been assumed that, in the context of a given application, a process belongs to a single group at any instant of time [4]. Additions to a group take place on the recovery of previously failed processes. Deletions from a group take place on the failure of previously operational processes. This system model for group membership is suitable for distributed applications running on a set of nodes connected by a reliable, usually fixed, wireline network in which link failures and network partitions are rare. The Transis project has dealt with membership services and group communication in environments where the network itself may get partitioned due to node and link failures, and nodes may operate for extended periods of time in a disconnected mode [5].

However, in the context of mobile environments, we need to consider additional aspects of group communication and this paper is an attempt in that direction.

First, group membership is affected not only by the state of processes (operational or crashed) and links (connected or disconnected), but also by the location of the mobile nodes. For example, a police dispatch service may wish to coordinate the actions of all non-busy troop cars within a kilometer of a crime site. We will describe a multi-level architecture for mobile systems in which there is a *proximity layer* between the *group membership layer* and the underlying mobile network.

Second, given a node p, it is the job of the proximity layer to determine the nodes in the vicinity of node p. Extending the troop car example, the proximity layer identifies all the cars within the specified distance. For this purpose proximity layer messages may have to be *location stamped* as well as *time stamped*.[1] Note that not all nodes in the vicinity of p, as determined by the proxmity layer, may become members of the group.

Third, we need to address the fact that a node may simultaneously belong to *multiple* groups. For example, a particular troop car may be within one kilometer of two different crime locations.

Fourth, mobility of nodes can inject some inaccuracy in determining group membership. This inaccuracy, if not avoided or restricted, could lead to a violation of the safety requirement of a mobile application.

Some of the mobility related issues for group mem-

*This work was supported in part by the National Science Foundation grant number CCR-9796331.

[1] Location stamping a message would involve the message carrying location information of the sender at the time of sending the message, just as time stamping corresponds to the message carrying the sender's clock value at the time of message send.

bership depend on the model of the mobile system. We shall consider three models: (i) cellular, (ii) virtual cellular, and (iii) the ad-hoc network models.

The paper is organized as follows: Section 2 describes the three models of mobile systems. Section 3 presents an overview of the group communication architecture. Section 4 describes the proximity layer protocol. Section 5 discusses group communication and membership service issues specific to each of the three network models. It also presents a protocol for constructing groups in mobile systems. Section 6 enumerates some of the properties of membership services in mobile systems. Section 7 presents the conclusions.

2. System Model

We briefly describe three mobile network models in increasing order of flexibility and decreasing order of fixed wireline content.

2.1. Cellular Network Model

A cellular network covers a certain area that is divided into possibly overlapping cells. Each cell has a fixed base station (BS). The base stations are connected to each other by a wireline network. Mobile nodes (also referred to as mobile hosts or MHs) can move from one cell to another. An MH always communicates with other nodes in the system through the base station of the cell in which it is present. In order to do so, the MH needs to establish a wireless link with its base station. If the communication partner is also present in the same cell, the base station forwards the message to the partner along another wireless link. If the partner is present in another cell, the base station forwards the message along the wireline backbone to the base station of the partner's cell. The backbone network is also connected to the telephone network.

2.2. Virtual Cellular Network Model

A virtual cellular network (VCN) is similar to the cellular network, except for one major difference: the base stations of a VCN are also mobile. The inter-base station links are also wireless. The mobile base stations continue to coordinate the activities of the MHs in their vicinity. However, unlike cellular networks, the graph of mobile base stations changes with time. As a result, several distributed algorithms that work for cellular networks cease to perform correctly. VCNs may be useful for tactical networks where mobile base stations may be installed on tanks, trucks, etc. Individual soldiers would carry the mobile hosts.

2.3. Ad-hoc Network Model

In an ad-hoc network all nodes are alike and all are mobile. There are no base stations to coordinate the activities of subsets of nodes. Therefore, all the nodes have to collectively make decisions. Due to mobility, a node's neighbourhood changes with time. As the mobility of nodes may not be predictable, changes in network topology over time are arbitrary. All communication is over wireless links. Ad-hoc networks are suitable for tactical missions, emergency response operations, electronic classroom networks, etc.

2.4. Network Connectivity

In a cellular network we assume that the wireline backbone network never gets disconnected and all base stations stay operational. As long as two MHs are operational and located in the coverage area, they can reach each other. From a graph theoretic point of view, the base stations are the internal nodes of a graph and the MHs are the leaves. Over time, the only changes to the graph are the leaf node to internal node interconnections. The subgraph comprising the internal nodes remains unchanged.

In the VCN model we assume that the mobile base stations are always reachable from each other along paths that consist exclusively of mobile base stations.[2] Each MH is connected to some mobile base station. However, the subgraph consisting of internal nodes (mobile base stations) changes with time.

In an ad-hoc network two mobile nodes share a link if they are within wireless range of each other. We assume that a single or multiple wireless hop path exists between every pair of mobile nodes. However, these paths may change over time.

In all the three network models described above, there is a possibility that a mobile node may move out of range of all the other mobile nodes, thus becoming disconnected. Even when the network stays connected and no nodes fail, mobile systems pose interesting problems because the network configuration may change. So, in subsequent discussions we address the group membership issues assuming that the network stays connected and there are no failures. As wireless signals have a limited range, the presence of a link implies that the separation between the nodes sharing a link is upper bound by a distance value.[3]

3. The Protocol Stack

Entrusting the group membership protocol to also determine network reconfigurations (due to node mobility) will complicate the protocol. Therefore, we propose the architecture shown in Figure 1.

The *proximity layer* protocol monitors changes in network configuration using the medium access control

[2] We are developing solutions to coordinate movement of mobile base stations while maintaining connectivity [6].

[3] We assume that a node always transmits at the same power level. If nodes were allowed to change the power level, the upper bound on link length would be determined by the maximum power at which a node can ever transmit. Such a loose upper bound would be of little use in several situations.

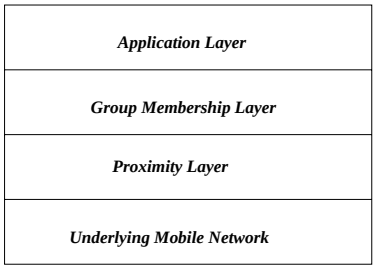

Figure 1. Protocol stack for group communication in mobile systems.

(MAC) sublayer communication primitives over the underlying mobile network. Given a node p and a distance parameter D, the proximity layer determines all nodes within distance D of node p. The proximity layer delivers this information at regular intervals to the group membership layer which can use the information to determine group memberships. Group membership is built by a protocol which runs *only* upon application demand, and not necessarily at regular intervals.

4. The Proximity Layer

The proximity layer consists of a protocol that uses services of the MAC sublayer to find all nodes at a certain distance d from the mobile host.

The MAC sublayer provides point-to point communication and beacons. We assume that each mobile node can successfully send a beacon once every t time units. The value of t will depend on: (i) the upper bound on the number of mobile nodes that can be present in the interference range of a node, and (ii) the MAC sublayer protocol used to send beacons. A beacon has a limited range and serves as an *"I am alive"* message. Neighboring nodes that are within its range can detect the presence of the mobile node even when it is not involved in any application level message exchange.

We define a connected wireless network as follows:

1. For every node x there exists a node y such that $distance(x, y) \leq d$, where d is the range of the beacon.
2. $distance(p, q) \leq d \Rightarrow connected(p, q)$
3. If $connected(p, q)$ and $connected(q, r)$ then $connected(p, r)$ for any p, q, r.
4. If for all p, q in the set of nodes, $connected(p, q)$ then the network is said to be connected.

The D-proximity Test: The proximity layer protocol is run once every t' time units, where $t' > t$. We assume that communication at the proximity layer is synchronous and the one-way message communication delay is bounded by t_p, *i.e.*, if a node is contending with other nodes in its interference region to send a message over a shared wireless channel it takes the node no more than t_p time units to successfully send the message. ALOHA protocols cannot guarantee such a delay bound. So, collision-free protocols like the bit-map protocol or binary countdown, or limited contention protocols like adaptive tree walk [7] should be used.

The goal is to find all nodes within distance D from a given node p. We assume that during a given *D-proximity* test the separation between nodes may change due to mobility, but the connectivity graph remains unchanged with a very high probability. This is because during the small time that a *D-proximity* test takes to run the distance that nodes can cover is small. Each instance of a *D-proximity* test can be uniquely identified by the following tuple: *(initiator_id, timestamp, location stamp, D)*. Here, *timestamp* and *location stamp* refer to the values of the initiator's local clock and location, respectively, at the time of starting the *D-proximity* test. [4] Let a node p initiate the *D-proximity* test. Initially, the *D-proximity* set of p only contains p.

If $D \leq d$ (the wireless range of a node), the proximity layer protocol at node p transmits a probe message and waits for replies for a period of $2t_p + t_r$ which equals the round-trip proximity layer message delay. Here, t_r is the upper bound on the processing time required by a node before it can propagate or respond to a *D-proximity* test message. All nodes within distance d hear the probe and send their location stamped reply. Note that no replies will be received after $2t_p + t_r$. For each reply, node p computes its distance from the node sending the reply (using the location stamp of the reply). If the distance is no more than D, the node sending the reply is added to the *D-proximity* of p. Alternatively, node p may just listen for location stamped beacons for t time units. Every node within distance d is bound to emit a beacon during this period. The *D-proximity* set can be determined without having to send any extra message.

If $D > d$, node p cannot directly reach all the nodes in its *D-proximity* with a single round of messages. Therefore, the following multi-round synchronous algorithm is executed. The algorithm entails flood-based formation of a breadth first search (BFS) tree rooted at the initiator, and subsequent convergecast.

1. Initially, all nodes are unmarked, their parent pointer is NULL and their *child* set is empty. Node p, the initiator, sends a flood message to all nodes within distance d. The message contains the 4-tuple uniquely identifying the *D-proximity* test.
2. When a node receives its first flood message for a particular *D-proximity* test, the node gets marked

[4] In 1994 the Federal Communications Commission (FCC Docket No. 94-102) and the Telecommunication Industry Association set the goal of locating a cellular telephone to within 125 meters sixty seven percent of the time by October 2001. Technology is already in development to be able to do so. This endeavour coupled with the availability of global positioning system (GPS) technology will enable location stamping.

and sets its parent pointer to the sender of this message. This node also propagates the flood by transmitting a message containing its own identity and its parent's identity in addition to the 4-tuple *D-proximity* test identifier. If multiple flood messages with the same 4-tuple arrive in the same round at an unmarked node, one of them is arbitrarily selected while the others are ignored.

3. Having propagated the flood, a marked node q listens for flood messages in the next round. All of q's children will be propagating the flood message in this round. If q hears a message containing its own identity in the *parent* field it adds the identity of the message sender to its child set. Thus, one round after getting marked as a node in the BFS tree, a node knows all its children. If no child is detected, the node realizes that it is a leaf in the BFS tree and initiates a convergecast towards the initiator of the *D-proximity* test.
4. During the convergecast, if a leaf node determines that its distance from the initiator p is less than or equal to D it sends a singleton set with its own identity to its parent. Otherwise, the leaf node sends an empty set to its parent. As the flood messages always carry the location stamp of the initiator p, computing distance from p is possible.
5. Once an internal node q in the BFS tree has received convergecast messages from all its children, it determines its own distance from node p. If the distance is greater than D, q sends the union of the sets received from its children to its parent. Otherwise, q also sends its own identity along with the union of sets received from the children.
6. Once p, the initiator, receives a convergecast message from all its children, the union of the sets received in the messages, along with itself, is the set of nodes in its *D-proximity*.

4.1. Accuracy of Proximity Test

During the execution of the *D-proximity* test the separation between node p and other nodes may change after those nodes send their convergecast message to their respective parents. Therefore, there is a possibility of some inaccuracy in the set returned by the *D-proximity* test. Let the diameter of the network be upper bound by $diam$. Then, the duration of the convergecast phase is bound by $diam \times (t_p + t_r)$. Let the maximum admissible speed of a mobile node be s. Note that the maximum admissible speed depends on the underlying wireless network system, *e.g.* DECT, GSM, PCS, ETACS. For example, in a DECT microcellular system s is less than 40 km/hours (approximately $11m/s$). For GSM and PCS systems s is bounded by maximum highway traffic speeds (about $50m/s$).

Therefore, from the time q sent its convergecast message to the termination of the *D-proximity* test, the maximum change in separation between nodes p and q is equal to $2s \times diam \times (t_p + t_r)$, and the maximum change in separation during the entire *D-proximity* test is twice that. Therefore, nodes that were within distance $D - 4s \times diam \times (t_p + t_r)$ of p at the start of the test are definitely included in the set. Nodes that were somewhere between $D - 4s \times diam \times (t_p + t_r)$ and D units away from p at the start of the test and were reported in the proximity set may have moved out of *D-proximity*. Nodes that were not reported in the proximity set and were somewhere between D and $D + 4s \times diam \times (t_p + t_r)$ units away from p at the start of the test may have moved into *D-proximity*. If we consider a metropolitan area network and node speeds upper bound by highway traffic speeds the error is not significant due to the following reasons: the diameter of the network is of the order of tens of links and t_p and t_r are fairly small values, typically few milliseconds.

Note that distance checking is done during the convergecast phase, and not during the flood phase. If, during the flood phase a node were to determine that its distance from p was greater than D and stopped propagating the flood message all nodes in the *D-proximity* of the initiator may not be detected. This error will be in addition to the small inaccuracy described in the previous paragraph. For example, let us consider Figure 2 representing the connectivity of an ad-hoc network. Let the distance between adjacent nodes be $d - \delta$, where δ is much smaller than d. Let $D = 3d$. Therefore nodes that are in the *D-proximity* of node p are p itself, *1, 2, 3, 11*, and *12*. However, if node 4 had stopped propagating the flood because its distance from p is greater than D, nodes 11 and 12 would never get detected.

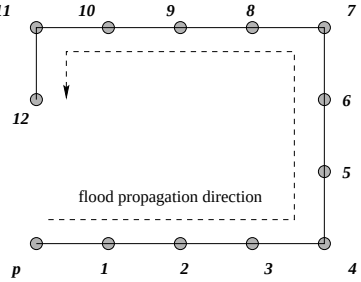

Figure 2. An ad-hoc network configuration.

5. Group Membership Layer

For group membership purposes a cellular network is similar to a wireline network if we assume that the base stations can act as proxies for the MHs in their cells. Thus, the mobility of MHs can be *hidden*. For example, let us consider channel allocation for communication between an MH and its base station. If an MH were to handle channel allocation it would need to poll all the MHs in its cell and neighbouring cells. This set of nodes would constitute its group for channel allocation purposes. Due to the mobility of MHs, the group would change with time even in the absence of node failure and disconnection. However, the job

can be simplified as follows. The task of channel allocation is delegated to the base station. The base station already knows the channels it is using to communicate with other MHs in its cell. The base station needs to communicate with the neighbouring base stations to gather information about channel usage in their cells. As this set of neighbouring base stations never changes, group membership remains unchanged when there are no base station failures or wireline link disruptions. Thus, mobility of MHs has no impact on group communication for channel allocation. A distributed dynamic channel allocation algorithm that follows this strategy has already been proposed in [8].

However, in a virtual cellular network a mobile base station's set of neighbouring counterparts changes with time even when there is no failure. Similarly, in an ad-hoc network the set of MHs in the interference range of a given MH changes over time. Hence, the set of nodes that need to be probed is dynamic.

As communication between nodes participating in group communication is not the same as the point-to-point communication at the proximity layer, messages may have finite but unpredictable delays. So, we model group membership layer communication as *timed asynchronous* [4].

5.1. Group Construction Protocol

Let us restrict our attention to group formation in virtual cellular networks and ad-hoc networks. A group formation initiated by a node p should return information about all the nodes that are within a distance D of the initiator and exhibit some desired characteristics. Note that this is not the same as the *D-proximity* test described earlier because every node within distance D may not be a member of the group.

We describe a three round protocol that is based on the three round protocol proposed by Cristian and Schmuck [9]. As example applications, we specify how a localized mutual exclusion or channel allocation algorithm can be superimposed on it. We assume that the initiator p knows *a priori* of a superset of nodes S, in the *D-proximity* of p, which contains the group g that p is trying to form. This information is provided by the latest run of the proximity layer protocol. The group construction protocol proceeds as follows:

1. Node p tentatively joins group g and multicasts a location stamped REQUEST message to the nodes in set S. Then p waits for responses. Note that a message may require multiple wireless hops on the underlying network to reach its destination. So, latency may be non-deterministic and significantly greater than t_p, the point-to-point proximity layer delay. Hence, the assumption about timed asynchronous communication at the group membership layer.
2. Node $q \in S$ does the following on receiving the location stamped message from p:
   (a) If q is within the pre-specified distance of p (equal to the interference range in the context of channel allocation), determined by comparing q's knowledge of its location with the location stamp of the REQUEST, and *some correctness condition* to be discussed later is satisfied, q sends an ACK to p along with the set of channels it is using at that time for communication ($busy_q$). Node q also tentatively joins the group g.
   (b) If q is not within the pre-specified distance of p, q sends a NACK to p.
3. On receiving an ACK or a NACK from every node in set S, p sends a JOIN(g) message to all nodes from which ACKs were received and adds them to group g. Node p also joins the group. On receiving the JOIN(g) message a node commits to joining group g. The union of *busy* sets of the elements of g denotes the set of channels being used within interference range. So, p selects a channel that does not belong to any *busy* set.[5]
4. During the execution of the first three steps of the group membership protocol, the underlying proximity layer protocol at p may return a set $S' \neq S$ in the *D-proximity* of p. In such a situation:
   (a) If node $r \in S' - S$, node p sends a REQUEST to r and awaits its response.
   (b) If $r \in S - S'$ the ACK received from r is discarded, JOIN(g) is not sent to r, and the *busy* set received from r is ignored. If the JOIN(g) message was already sent to r before knowing about S', only the *busy* set received from r is discarded, but r continues to be in the group g.

In a sense, Step 1 of the algorithm keeps running until the end of Step 3. The channel allocation algorithm terminates because there are a finite number of nodes in the network.

The *correctness condition* mentioned in *Step 2(a)* of the protocol depends on the algorithm being used to ensure mutual exclusion or interference free access to channels. If an ACK is sent immediately after receiving a REQUEST, two neighbouring nodes may simultaneously enter the critical section or may start using the same channel causing interference, respectively. Therefore, as in the Ricart-Agrawala algorithm [10], ACKs may be deferred based on the timestamps of the REQUESTs. This is the policy employed in [8].

Therefore, if mutual exclusion or channel allocation messages are to be piggybacked on the group construction messages the time to construct a group will be influenced by delays involved in application-specific message communication. If group construction is to be employed to facilitate multiple applications, besides mutual exclusion or channel allocation, such a delay may

[5] In the case of a localized mutual exclusion, p's desire to enter the critical section triggers the group formation and permission from other processes in the vicinity can be piggybacked on ACKs.

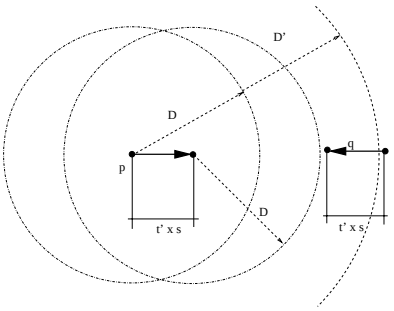

Figure 3. Margin of safety of the non-interference area of process p.

not be acceptable. In such situations, first group construction will be performed: ACKs can be sent without any delay and they do not carry the *busy* sets. Then mutual exclusion or channel allocation messages will be exchanged among nodes in the group.

Accuracy of Group Construction Protocol: As in the D-proximity test, there could be some inaccuracy in determining group membership due to the mobility of nodes. This could lead to a violation of the safety requirement of the application. For example, two channels could be allocated to two communication sessions that are within each other's interference range.

Let t' be the interval between two successive deliveries of the D-proximity set of p. The maximum distance covered by p in an interval is $t' \times s$ where s is the maximum admissible speed of p. Let the group construction protocol use a D-proximity set S and terminate δ time units before the next D-proximity set S' is delivered, where δ is a very small value. Then, p could be using a channel which is already allocated to process q, where $q \in S'$ and $q \notin S$. This problem arises as the channel allocation algorithm runs on an old group membership.

To avoid this interference we could build a $(D+D')-$ *proximity* set and then try to achieve non-interference in this group. Here D' is the margin of safety. To calculate D' let us consider the worst case scenario depicted in Figure 3. During the execution of the group construction protocol process p covers a maximum distance of $t' \times s$. Let us consider a process q that proceeds at the maximum speed s towards p, covering a distance of $t' \times s$. In order to be safe the following inequality must hold: $2 \times t' \times s + D < D + D'$. Hence, D' must be at least $2 \times t' \times s$ as shown in Figure 3. Note that, if we consider t' to be of the order of seconds, even in the context of DECT *microcellular* system, D' is about 20 meters which is a quite small value compared to the value of D which is of the order of a few hundred meters.

Each process that was out of the $(D+D')$-proximity set at time t will be also out of the non-interference area of p with radius D at time $t'' < t + t'$, as shown in Figure 3.

6. Discussion

6.1. Multiple Group Memberships of Nodes

It has been traditionally assumed that a node belongs to a single group at any point of time, with respect to an application. However, such is not the case in mobile systems. For a given distributed application a node may belong to multiple groups simultaneously. Consider part of a network of base stations of a cellular network shown in Figure 4.

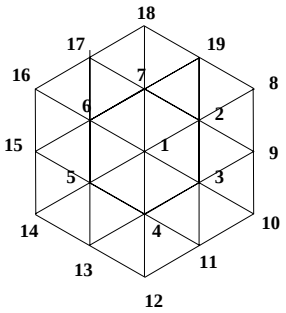

Figure 4. Base Station Connectivity in Cellular Network

When base station 1 needs to allocate a channel for communication with an MH in its cell, this base station has to communicate with base stations 2 - 7, which are in its co-channel interference range, to determine the channels being used. Therefore, base stations 1 - 7 constitute a group. Similarly, when base station 2 needs to allocate a channel, its group consists of base stations 1 - 3, 7 - 9, and 19. Thus, any base station is a member of multiple groups. Multiple group membership is not a significant issue in cellular networks because in a no failure condition all these groups remain unchanged and can be computed *a priori*. However, in the case of VCN and ad-hoc networks the following need to be determined:

1. What is the number of groups to which a node belongs at a given point of time?
2. What is the membership of each group to which a node belongs?

The impact of location on group membership is due to the *spatial reuse characteristics* of resources in a mobile environment. A wireless channel is a shared resource. Allocating a channel to support a communication session while avoiding interference is akin to providing mutually exclusive access to the shared resource. However, the same channel can be used simultaneously to support multiple communication sessions provided the sessions are separated by a certain distance. This is analogous to having multiple instances of the same resource present in the system with their availability limited to only parts of the network. Therefore, group

communication for such applications involves multicasting to a subset of active nodes, as opposed to broadcasting to all active nodes [8].

6.2. Agreement on a Linear History of Groups

A node p is said to be in the *locality domain* of another node q at time t if node p belongs to one of the communication groups of q at time t. We propose the following partial order on the history of groups to which a process p belongs: Let g_p be a group to which p belongs during the time interval $[t_1, t_2]$ and g'_p be a group to which p belongs during the time period $[t_3, t_4]$. We define the precedence relation on group history, $\rightarrow$, as follows:

1. $g_p \rightarrow g'_p$ if $t_2 < t_3$
2. if $[t_1, t_2]$ and $[t_3, t_4]$ overlap then $g_p \parallel g'_p$
3. if $g_p \rightarrow g'_p$ and $g'_p \rightarrow g''_p$, then $g_p \rightarrow g''_p$

This relation is similar to the *happened before* relation proposed by Lamport [11], except that we now consider time intervals instead of time instants. The group transitions for a node constitute a lattice. In such a lattice there is a directed edge from g_p to g'_p if: (i) $g_p \rightarrow g'_p$, and (ii) there does not exist a g''_p such that $g_p \rightarrow g''_p \rightarrow g'_p$. Then a group communication service should satisfy the following property:

- Let nodes p and q join group g at local times t^p and t^q, respectively. Also, let p and q later join group g' at local times $t^{p'}$ and $t^{q'}$, respectively. Let p and q be in each other's locality domain during the intervals $[t^p, t^{p'}]$ and $[t^q, t^{q'}]$, respectively. Then, there exist sequences of group transitions $(g_p^1, g_p^2, \ldots, g_p^n)$ and $(g_q^1, g_q^2, \ldots, g_q^n)$ in the lattices of p and q, respectively, such that:
  1. $g_p^i \rightarrow g_p^{i+1}$ and $g_q^i \rightarrow g_q^{i+1}$, $1 \leq i < n$,
  2. $membership(g_p^i) = membership(g_q^i)$, $1 \leq i \leq n$, and
  3. $membership(g_p^1) = membership(g)$, and $membership(g_p^n) = membership(g')$.

A group communication service that satisfies this property can ensure that there exists a subset of messages delivered to node p in the interval $[t^p, t^{p'}]$ that is also delivered to q in the interval $[t^q, t^{q'}]$ in exactly the same sequence. This agreement on the history of message deliveries is possible if group communication is totally ordered and atomic. If these messages carry state update information then two nodes that are in each other's locality domain always receive those updates of shared information in exactly the same order and can maintain mutually consistent copies. This property can be exploited to support fault tolerant redundant systems.

6.3. Similarities with Weak Virtual Synchrony

The assumption, in the group construction protocol, that the membership of set S is known *a priori* exhibits similarities with the approach adopted by Friedman and van Renesse [2] in describing *weak virtual synchrony (WVS). Strong virtual synchrony (SVS)* requires that a message be delivered within the view in which it is sent [2]. However, in order to implement SVS, during view changes, processes have to temporarily stop sending messages until a new view is installed. The duration of such stoppage is a function of the delay of the underlying network.

In order to avoid the stoppage of message sends, Friedman and van Renesse [2] proposed the notion of WVS. In WVS there is no need to stop sending messages when the view change protocol is being run. In the beginning of a view change a *suggested view*, which is a composition of the old view and every process that wishes to join, is introduced to the processes. The first suggested view during a new view construction is always a superset of the new view. Every subsequent suggested view and the new view is a subset of its predecessor suggested view.

This is similar to the operation of the proposed group construction protocol. The protocol initially starts with a set S of neighbouring nodes. When the proximity layer *suggests* another neighbour group S' ($S \neq S'$) during the execution of the group construction protocol, REQUESTs are sent to all processes in $S \cup S'$. Therefore, $S \cup S'$ is equivalent to the first suggested view. Subsequently, nodes in $S - S'$ are dropped from the *suggested view*/group when their ACKs are discarded. As in WVS, there is no need to abort the already running group construction protocol and start afresh when S' is received.

6.4. Final Remarks and Future Work

In the proposed architecture a process can build a group by selecting processes from its proximity set according to some criterion. Hence, there could exist some physically separated/disjoint groups, like *islands*, that provide the same service. If this service needs to share a common state across all the disjoint groups, then a question arises: how to exchange state update information among disjoint groups to maintain state consistency? An answer could be to build a sublayer on top of the group membership layer, giving the illusion to the user of interacting with a unique group (formed by the union of distinct islands) providing that service. This sublayer would take care of the exchange of state update information among distinct groups.

Extending the notion of *islands*, such a sub-layer would construct *bridges* between islands. In order to construct such bridges, pairs of nodes p and q that belong in each other's locality domain and agree on a linear history of groups may be used. If p belongs to one of the disjoint groups and q to another disjoint group, and p and q agree on a linear history of groups

for a time interval, they can be used to exchange state information between the disjoint groups. Such a *bridge* has a single span: *(p,q)*. However, multi-span bridges may also be constructed between two disjoint groups g_1 and g_2 using a sequence of nodes $n_1, n_2, \ldots, n_N$, for a period T such that:

1. $n_1 \in g_1$, $n_N \in g_2$;
2. for all i: $1 \leq i < N$: n_i and n_{i+1} are in each other's locality domain and agree on a linear history of groups for a period of time t_i;
3. T is the time interval that is common to all t_i, $1 \leq i < N$.

Note that due to the mobility of nodes during the lifetime of a service, different bridges (composed of different sequences of mobile nodes) may exist between a pair of disjoint groups that provide the same service. There may also be intervals during which there may not be any bridge between two disjoint groups and their states may diverge. In so, as soon as a bridge is subsequently formed, the two disjoint groups should resynchronize to reach a mutually consistent state. The idea of providing a sublayer to integrate disjoint groups providing similar service needs further investigation and will be the focus of our future research.

The multi-layer protocol architecture described in this paper can be enhanced by supporting alternatives to the proximity layer between the group membership layer and the underlying network. Such an alternative layer could provide information to the group membership layer on the basis of some other criterion like type of service or some cost/QoS factor. This enhanced architecture will be along the lines of the HORUS architecture which permits several independent options at each layer such that each option at a layer is compatible with several options of the layers above and below. Depending on the operating environment and desired functionality, appropriate layer options from each layer can be picked up and bundled together to form the protocol stack [3].

7. Conclusion

Membership services and group communication are useful in distributed systems as they enable modular design of the system and support reliable communication. In this paper we discussed membership services and group communication in the context of mobile systems. We described how group membership can change due to mobility of nodes. This is in addition to membership changes due to failure and recovery in static distributed systems.

We presented an architecture for group membership in mobile systems. With the example of channel allocation, we motivated the need for *location stamping* messages in mobile systems and described how a node may concurrently belong to multiple groups. We presented a synchronous proximity layer protocol to determine mobile nodes in the proximity of a given node. We also presented a variation of an existing three round protocol for the construction of groups in mobile systems.

Finally, we showed how the mobility of nodes injects inaccuracy in the determination of group membership leading to a possible violation of the safety requirement of an underlying mobile application. This problem is typical of mobile distributed systems. We proposed a method to build a margin of safety in a group membership determination that can be easily embedded in the architecture.

References

[1] R. Baldoni, R. Friedman, and R. van Renesse, "The Hierarchical Daisy Chain Architecture for Causal Delivery," in *Proceedings of the* 17^{th} *IEEE International Conference on Distributed Computing Systems*. 1997, pp. 570–577, IEEE Press.

[2] R. Friedman and R. van Renesse, "Strong and Weak Virtual Synchrony in Horus," Tech. Rep. TR95-1537, Department of Comuter Science, Cornell University, August 1995.

[3] R. van Renesse, K.P. Birman, and S. Maffeis, "Horus, a flexible Group Communication System," *Communications of the ACM*, April 1996.

[4] F. Cristian, "Synchronous and Asynchronous Group Communication," *Communications of the ACM*, vol. 39, no. 4, pp. 88–97, April 1996.

[5] D. Dolev, D. Malki, and R. Strong, "A Framework for Partitionable Membership Service," Tech. Rep. CS95-4, Institute of Computer Science, The Hebrew University of Jerusalem, 1995.

[6] C. Shields, Jr., V. Jain, S. Ntafos, R. Prakash, and S. Venkatesan, "Fault-Tolerant Mobility Planning for Rapidly Deployable Wireless Networks," in *Proceedings of the IEEE Workshop on Fault-Tolerant Parallel and Distributed Systems*. April 1998, LNCS, Springer-Verlag.

[7] A. Tanenbaum, *Computer Networks(*3^{rd} *Edition)*, Prentice Hall, Upper Saddle River, N.J., 1996.

[8] R. Prakash, N. Shivaratri, and M. Singhal, "Distributed Dynamic Channel Allocation for Mobile Computing," in *Proceedings of the* 14^{th} *ACM Symposium on Principles of Distributed Computing*, Ottawa, Canada, August 1995, pp. 47–56.

[9] F. Cristian and F. Schmuck, "Agreeing on Processor Group Membership in Timed Asynchronous Distributed Systems," Tech. Rep. CSE95-428, University of California, San Diego, 1995.

[10] G. Ricart and A. K. Agrawala, "An Optimal Algorithm for Mutual Exclusion in Computer Networks," *Communications of the ACM*, vol. 24, no. 1, pp. 9–17, January 1981.

[11] L. Lamport, "Time, Clocks and the Ordering of Events in a Distributed System," *Communications of the ACM*, vol. 21(7), pp. 558–565, July 1978.