Communication and Computing Paradigm for Distributed Mobile Systems

Maluk Mohamed M A
Principal & Professor / CSE
M.A.M. College of Engineering
Tiruchirappalli, Tamilnadu, India 621 105
maluk@mamce.org

Abstract

The advancements in technology have enabled mobile devices to become information and service providers by complementing or replacing static hosts. This motivates the need for merging of mobile and grid technologies, leading to mobile grid paradigm. Computational mobile grid can also be viewed as a seamless integration of cluster of mobile clusters. Hence, as a first step in realizing a mobile grid, Anonymous Remote Mobile Cluster Computing (ARMCC) model is proposed. The main purpose of ARMCC is to utilize the idle computing power of both the static and mobile nodes, to provide parallel programming on a distributed mobile computing environment. The cluster model was extended to a mobile grid paradigm that integrates the computational, data and service grids. However, though the mobile grid is visualized as cluster of mobile clusters, the model of mobile cluster is not directly applicable to mobile grid. In addition to computational power other resources and services offered by the participating nodes are shared in case of mobile grid. Thus, all the participating mobile nodes in the mobile grid are represented using surrogate objects which reside on the static portion of the network. These surrogate objects are realized as a shared distributed object space. The constrained nature of the mobile devices participating in the mobile cluster requires an efficient communication primitive for exactly-once message delivery. Thus a novel communication paradigm, namely, Fault Tolerant Exactly Once Reliable Multicast Protocol (FTEORMP) for distributed mobile system is proposed.

Key words: Computational Mobile Grid, Cluster Model

I. INTRODUCTION

High speed networks and the improved performance of processors in the recent years, has made network of workstations [1] an efficient paradigm for parallel and distributed computing. On the other hand, with rapid evolution of wireless communications, a huge number of communication devices and mobile devices have been developed. These devices provide high computing power and storage capabilities. At the same time, the cost for these high end mobile devices is decreasing and thus becoming affordable for everyone. Thus, computing is no longer restricted to desktop computers, and access to information, resources and data is made available anytime and anywhere [2]. This has led to a new era in computing, namely distributed mobile computing. It involves distributed mobile systems that are defined as a collection of processes where some processes are running on Mobile Hosts (MHs), in addition to some processes running on Static Hosts (SHs). These processes communicate with each other by exchanging messages. This paradigm provides the flexibility for the mobile users to roam around freely while maintaining connectivity with a wired computing infrastructure. Thus enabling mobile users to communicate/compute effortlessly, anywhere, anytime.

This work specifically addresses problems arising out of integrating mobile devices into distributed systems. Applications with distributed systems are not suitable in distributed mobile systems because of bandwidth of wireless mobile networks, mobility and battery capacity of mobile devices and so on. To this end, a novel mobile cluster computing paradigm, namely MOSET is proposed. The mobile cluster was further extended as cluster of clusters to form, a mobile grid. In addition, a specialized exactly-once reliable multicast protocol is also proposed, to provide an efficient communication among the participating mobile nodes, considering the constraints of the MHs.

A. Motivation

Distributed mobile systems typically require special solutions, for a number of reasons. First, traditional network protocols implicitly assume that hosts do not change their physical location over time, which is not true with mobile systems. Second, mobile hosts have severe resource constraints in terms of energy, processing and storage resources. Also, wireless networks are characterized by limited bandwidths and high error rates. Furthermore, mobility introduces new issues at the algorithmic level and in the design of communication and computing paradigms. For example, a mobile host may miss messages simply because of its movements, even with perfectly reliable communication links and computers that never crash [3]. All the above reasons imply that network protocols and distributed algorithms for a mobile computing environment cannot assume that a host maintains a fixed and universally known location in the network at all times. Mobile computing also has important implications for various computing paradigms for distributed and parallel processing and in particular, paradigms aimed at mobile users.

Many new applications for such systems arise with the rapid advancement in wireless and mobile technology. In applications, such as on-field applications, group collaborations among the static and mobile hosts of the logical community are essential for sharing and comparing the collected data. Such interactions involve group communications and computing of various types in order to identify and agree on a situation. The group communication could be achieved using multicast communication primitive.

Further, as the mobile members of the group are resource constrained devices, they need to share the resources of the other members for completing the necessary computational tasks. The mobile hosts are also capable of providing context and location based information which may be essential for may applications. However, these mobile hosts because of there constraints require a special treatment in terms of their communication and computing requirements. Thus the main goal of this work is to consider the effect of host mobility and other key constraints associated with the mobile hosts, to build communication and computing paradigms for distributed mobile systems. These include the paradigms for reliable delivery of multicast messages, mobile cluster and mobile grid.

II. OBJECTIVES AND SCOPE

The objectives of the work are as follows:

- ❖ To design and investigate the feasibility of wide area distributed mobile systems to act as grid computing paradigm, namely mobile grid.
- ❖ The mobile grid is realized as cluster of mobile clusters. Hence the viability of the distributed mobile systems to act as a cluster computing paradigm for parallel programming by seamlessly integrating MHs into traditional cluster model is to be designed and investigated.
- ❖ The above computing paradigms for distributed mobile system, requires an efficient communication primitive to delivery of messages. Hence an efficient multicast protocol for distributed mobile system that can handle the key constraints of the MHs is to be designed and investigated.

The proposed communication protocol and computing models are currently applicable for cellular based distributed mobile systems as shown in figure 1.

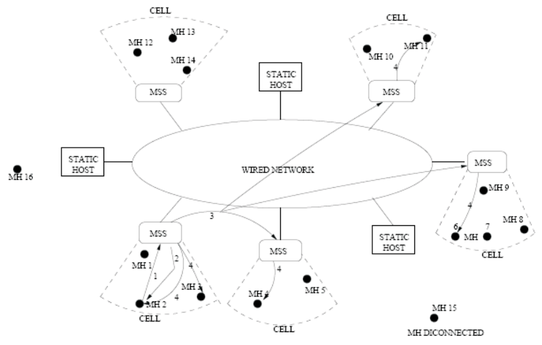

Fig. 1. System Model

III. DESCRIPTION OF THE PROPOSED WORK

A communication and two computing paradigms for distributed mobile systems, named FTEORMP, Moset, and mobile grid are presented.

A. Fault Tolerant Exactly-Once Reliable Multicast Protocol

The FTEORMP primitive is built as a two-tier model. The multicast primitive provides exactly-once message delivery, and supports fault-tolerance. The primitive is demonstrated using a case study and provides theoretical proofs with lemmas to prove the correctness of the proposed paradigm. The key features of the proposed protocol are:

- ❖ Multicast communication is reliable and delivered exactly-once.
- ❖ Tolerates MSS failures.
- ❖ Size of data structures at MHs, size of message headers, and number of messages in the wired network for each multicast do not depend on the size of the multicast group.
- ❖ Viewing the mobile systems as a two-tier model, in contrast to the existing view of mobile systems as a three-tier model.
- ❖ System model considered decentralizes the message traffic in the fixed network.

B. Protocol Overview

When MH wants to multicast a message, it sends a request message, MCASTREQ to its associated MSS. MH continues the request until an acknowledgement, MCASTACK is received. On reception of the acknowledgement, MH starts sending the next message. If the acknowledgement sent by the MSS gets lost, then the MH times out and retransmits the request message. In such case the MSS may receive the request more then once, and to detect duplicate request, a one-bit sequence number, TxSeq is assigned to every request generated from an MH. Similarly each MSS maintains a one-bit sequence number, RxSeq[H] of the next request it expects to receive from each MH H. If RxSeq[H] is not same as the TxSeq of the request received from MH H, then the message is discarded as duplicate.

The MSS is responsible for buffering multicast messages and managing the ordering of messages for delivery to MHs. The initiator of multicast can be either a static or a mobile host. When it is a static host it can directly multicast to static host members and for MH members, the multicast message is passed to the corresponding MSS. If the initiator of the multicast is MH, then the messages along with the multicast group are forwarded to its associated MSS. The message to be multicast carries the sequence number locally generated by the initiator MH, which enables the MSS to handle the message in sequence. The MSS constructs a new message with Tag at the first field to identify the type of message, sequence number and MH identifier as the second and third fields respectively, along with the message at the end to help in ordering of messages in the MSS. The constructed message is then multicast to MSSs which are members of G (Denotes the set of MSSs such that all MHs in the multicast group lie in the domain of some MSS in G). The MSS on receiving the message constructs a new sequence number, with the help of MH generated sequence number and MH identifier, for providing total ordering of the message. The MSSs uses the dynamic channel allocation technique to route the messages to the MHs. The MH uses the sequence numbers to deliver the messages exactly-once in sequence.

C. Fault-Tolerance

Inorder to handle failure of MSS, the states of all the MSSs are replicated in another MSS. A MSS M' is said to be a neighbour of another MSS M", if M' had replicated its information in M" to handle fault-tolerance. When an MH moves from one cell to the other, the MH sends a greeting message to the new MSS Mnew along with the MH identifier, the old MSS identifier Mold and the last message received sequence number. The MSS on receiving the greeting message registers the entry of new MH in its data structure and sends a message to Mold to deregister the information associated with this MH, after checking locally whether the Mnew is the neighbour for Mold. If Mnew is not a neighbour of Mold then request for transferring all the information associated to the MH, before deregistering. The above proposed method helps in retrieving the state of the MSS after it comes back from failure, however MHs within the cell cannot continue their communication until MSS comes back.

In parallel with and separately from the single hop cellular model, another type of model, based on radio to radio packet multihopping, has been emerging to serve a growing number of applications which rely on a fast deployable, wireless infrastructure. When an MH is not able to contact any of the MSSs, which may be due MH entering a out of coverage area or due to MSS failure, in such case an MH may not be able to access any other hosts. Using the radio packet multihopping technique, the MH tries to identify its reachable MH, which may or may not be within the coverage region of a cell. Thus in such scenario the MHs tries to continue communication through the adhoc model among the MHs. Thus the method helps in continuing communication even when an MH enters a out of coverage area, or when the MSS of its cell fails. As the state of the failed MSS is duplicated with its neighbour, the communication can still continue in adhoc mode by contacting the neighbour.

D. Total Ordering of Message

In this section, the overview of the total ordering portion of the protocol is presented. The overview is based on figure 2, with the mobile system consisting of 3 MSSs and 4 MHs, with all the four MHs belonging to the same multicast group. MSS1 buffers the multicast message that has been sent (1) to it from its group member MH1. MSS1 after acknowledging (2) the reception of the message, multicast (3) the message to other MSSs which are within the domain of the multicast group, namely MSS2 and MSS3. If parallely MH5 sends (1) its message to MSS3 for multicasting, then the same steps as explained above for the message sent from MH1 takes place. When the multicast message from both the MHs reaches all the members of G, all the MSSs waits for response from members of G which do not have any multicast message generated from its cell, ie. MSS1 and MSS2 waits for a "no multicast" message. The message to be delivered is reassigned a sequence number with the following logic, on receiving the message from all members of G, the MSS starts delivering the messages ordering them, first by the MH id and then by the sequence number, with the delivery being alternative, namely as 01,51,02,52,03,53,04,etc. The delivery is made alternative to ensure that all the multicast are equally given chance for delivery. Otherwise when the multicast message from an MH is long, then it does n

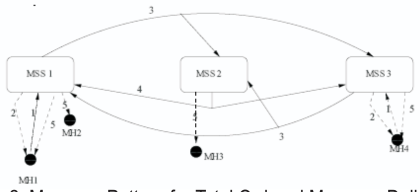

Fig. 2. Message Pattern for Total Ordered Message Delivery. Dotted Arrows Indicate Wireless Communication

E. Simulation Study

To evaluate the performance of our protocol, simulation and comparision with an existing protocol is needed. As existing simulators like NS [4] and GloMoSim [5] does not support our system model, we developed an Object-Oriented discrete event simulator in C++ similar to [6]. The simulation was done considering 4 multicast groups, 32 MSSs and 512 MHs. A detailed study and comparison of the performance has been made in terms of effect of message size, scalability, effect of host mobility and queuing delay.

Role of Queuing Delay:

Our proposed protocol does not have a centralized coordinator unlike other existing protocols, rather it is distributed. Due to this, intuitively, coordinator must be the bottleneck. This could be seen in the table 1, which shows that the queuing delay at the coordinator for the Relm is a major bottleneck. The bottleneck for both the 3-tier protocols seems to be the coordinator. This is mainly due to all multicast getting routed through the centralized coordinator. In addition due to significant mismatch between the bandwidths of the wired and wireless media, the delay in MSS could be noticed. Any host that interfaces two different network media with significantly different bandwidths will be a bottleneck. Similar results can also be seen in [6].

On comparing our protocol with RelM, we infer that the queuing delay constitutes a major portion of the average delay for RelM. In RelM, high coordinator queuing delay is primarily due to handoff messages and high MSS queuing delay is because of unicast in wireless medium. Moreover, due to an acknowledgement-based protocol also, the message traffic in the wired network is very high. Since these bottlenecks are eliminated in our protocol, queuing delay does not contribute significantly to the average end-to-end delay and hence, the load on the wired network is very low even with the total ordering of messages. Further, if the queuing delay contributes to the major portion of the average delay of a protocol, then it does not scale well.

Table 1 : Role of Queuing Delay

Message Size (bytes)	Average FTEOMP Delay (ms)	FTEOMP MSS Queue Delay	Average RelM Delay (ms)	RelM Coord Queue Delay	RelM MSS Queue Delay
256	2.79	0.41	586.85	153.82	201.45
512	4.03	0.79	1541.73	187.76	469.67
768	4.92	0.93	2347.34	349.13	877.34
1024	5.31	1.69	2653.13	301.67	1030.89
1280	8.83	1.99	3174.96	309.01	1326.11
1536	9.36	2.36	3602.04	554.04	1518.89
1792	10.25	2.65	4443.54	549.59	2024.13
2048	11.42	3.72	4877.92	551.32	2273.77

A. Moset

Moset supports parallel computing, by making use of the idle processing power of the static and mobile hosts that form the cluster. To realize such a system for parallel computing, various issues such as asymmetry in connectivity, mobility of hosts, disconnectivity of mobile hosts, architecture and operating system heterogeneities, timeliness issues, load fluctuations on participating hosts, host availability variations and failures in workstations and network connectivities need to be handled. Moset is designed to provide transparency to mobility of hosts, distribution of computing resources and heterogeneity of wired and wireless networks. The model has been verified and validated by implementing a distributed image rendering algorithm over a simulated mobile cluster model.

B. Moset Overview

The basic principle with which the Moset model was designed was to abstract out the heterogeneity of the constituting hosts from the user. The user is made transparent to the hardware, bandwidth, operating system and other heterogeneity existing below the kernel. The user is given freedom to avail any computing power required for his application, without being concerned about whether he is working with a constrained host or not. Moset is designed with clear separation between the administration functionality and the user functionality. It is the function of the administration to install and maintain the system. Once the system is deployed, the user needs only to use the APIs to interact with the system for performing parallel computing. Figure 3 illustrates the basic architectural overview of the proposed Moset model.

In our model, the MSS aggregates the computing resources which are within its cell and present to the distributed system as a set of its own resources. The hosts which are participating in the cluster computing are grouped based on the memory capability of the hosts. The hosts which are participating in the Moset kernel registers their computing entity to the coordinator of the system, based on their capability. The entire data which needs to be processed is multicast to all the participating hosts in the particular group based on the size of the data. The run-time system of the kernel decides on the anonymous host on which the task is to be executed based on its capability.

Unlike the hosts in a traditional distributed system, the mobile hosts cannot maintain high levels of availability or reliability due to wireless connectivity.

Hence, in order to achieve reliable delivery of the data, considering the constraints of the mobile hosts, Moset is built over a exactly-once reliable multicast protocol.

C. Moset Computation Model

The Moset computational model is designed in such a way that it could handle the heterogeneity, fault-tolerance, dynamic load balancing and computing power availability. The dynamic load on the participating systems and the hosts and link failures make the traditional cluster computing model unsuitable for parallel programming on MCC. These issues were effectively handled in [7], however the model does not address the mobile host participation in computation and the issues related to it. The Moset model is aimed to integrate the mobile hosts with the static hosts to form a mobile cluster, and to harness the idle computing power, of static and mobile hosts to utilize them for parallel computing.

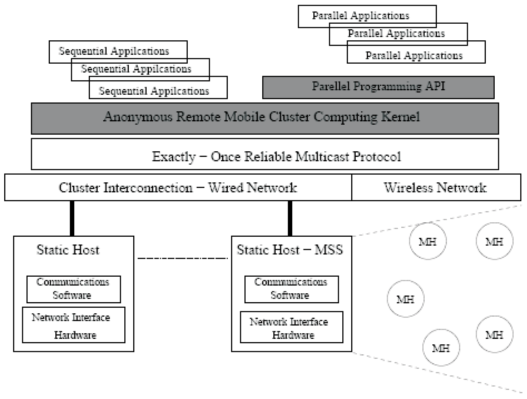

Fig. 3. Moset Architecture

Cluster-Subgroups:

In our model, we use a notion of Cluster-Subgroups (or Subgroups), based on the memory capability of the hosts. A Cluster-Subgroup refers to the characteristics of the task submitted to that subgroup. Each host (static and mobile) based on its capabilities, joins the respective subgroups. For example, Subgroup LOW may refer to those tasks having memory requirements < 10MB. A MODERATE Subgroup may have tasks having memory requirements < 50MB. A HIGH Cluster-Subgroup may have tasks having memory requirements < 100MB. Tasks with memory requirements > 100MB may be in Subgroup VERY HIGH.

Horse Power Factor and Dynamic Load Balancing:

As each host may have different capabilities (such as memory and processing power), it is essential to allocate tasks to the SHs and MHs based on their capabilities and processing power. To incorporate this, each host is allocated an integer called Horse Power Factor (HPF) [7], which is a measure of the computing power of a machine, load on the machine and the network bandwidth of the communication channel. Machines in the network are normalized by a benchmark program to obtain a relative index of the machine, which is a static factor. The dynamic HPF of a machine is obtained using this static relative index, the load on the machine and the communication bandwidth with which the machine is connected to the network. This dynamic factor is normalized as a factor that represents the number of entities that it could compute. When a host has HPF 'h', then 'h' computing entities are allocated to the host. For example, if hosts A and B have h1 and h2 as their respective HPFs, then the time taken by A to compute h1 amount of a task is approximately equal to the time taken by B to compute h2 amount of the same task. Abstracting the heterogeneity in this way makes it viable for parallel processing on unevenly loaded heterogeneous machines.

Dynamic load balancing [8] is done by maintaining two thresholds viz., Upper Threshold (UT) and Lower Threshold (LT) at the MH. These thresholds are shared by all the computing entities within a MH. This can be achieved by creating the computing entities as threads of the MH. When the load on the MH is greater than UT, a computing entity on that MH leaves the group and increments UT and LT on that MH. Both UT and LT are incremented so that all entities do not leave the groups at the same time. Similarly, when the load on the MH becomes lesser than LT, a computing entity on that MH joins the group and decrements both UT and LT. The need for two thresholds UT and LT is to avoid oscillations of frequent join and leave. Further, the load balancing mechanism used is non-preemptive [8] and hence migration of already executing task is not done. This is due to the additional communication overhead involved in process migration.

Parallelism in the Model:

The dataset which is very large is multicast to the subgroup, based on the size of the file. Each computing entity independently splits the task based on its ID and N. For example, in the case of distributed image rendering application, frames having frame number 'f' such that mod(f;N) = ID are rendered by the computing entity with identi¯er ID. When an entity completes its share, it sends the result back to the destination host.

A host is assigned as a coordinator to the cluster and it keeps track of the total number of computing entities (called N) under each Cluster-Subgroup. Further, each computing entity has a unique membership identifier (called ID, ranging from 0 to N ¡ 1) associated with each group subscribed by it. Whenever a MH wants to participate in distributed processing, the daemon at the MH registers a set of computing entities, based on its capabilities. The exact number of computing entities spawned by each daemon will depend upon the capability and the processing power of each MH.

A. Implementation

A Distributed Moset kernel is spread over the hosts that participate in Moset. Moset kernel consists of multiple local coordinators (LC) to coordinate local activities, multiple co-coordinators (CC) to coordinate the global activities within their cell, and one system coordinator (SC) to coordinate the overall global activity of the entire cluster. Each host, either static or mobile that intends to participate in the Moset kernel runs a local coordinator. MH has a client process and a daemon. The client process and daemon run over a reliable multicast protocol. The client processes are used to submit tasks to the MHs (via multicast protocol) for distributed processing. The daemons are the computing entities at the MHs that execute a part of the submitted task concurrently with other daemons.

B. Performance

The performance study of the model was done by implementing the distributed image rendering application over the FTEORMP. Each MH has a client and a daemon. The clients are used to submit tasks to the MHs for distributed processing. The daemons are the computing entities at the MHs that execute a part of the submitted task concurrently with other daemons. Both client and daemon run over FTEORMP and use the socket abstractions of FTEORMP simulator. The daemons run on the SPARC machines and communicate with the respective MHs of the simulator by sockets. The image rendering application developed renders an image obtained by CT scan. The characteristics of a CT scan image [9] are that they contain information from a transverse plane only.

Table 2 : No Load Performance Analysis for Ultra SPARC 333 MHz and SPARC 500 Mhz

	SPARC 333MHz		SPARC 500MHz	
No. of Hosts	Time Taken(sec)	Speedup	Time Taken(sec)	Speedup
1	434		303	
2	211	2.056	149	2.033
3	149	2.912	102	2.971
4	113	3.841	77	3.935
8	61	7.232	42	7.214
16	47	9.234	33	9.181

Table 2 shows the results when daemons were executed on SPARC 333MHz and SPARC 500MHz under no load condition. On a SPARC 333MHz, super-linear speedup is noticed when number of hosts is 2. On a SPARC 500MHz, super-liner speedup is noticed when the number of hosts are 2, 3 and 4. The reason for super-linear speedup is because, during the entire computation of 360 frames, the full volume data (which is reasonably large - 1.37 MB) needs to be in memory. During this time, page faults 7 occur and the computation time increases. The context switch time is very negligible as the experiment was conducted in no load conditions (when the load on the machines was only due to the daemons). When the task is split to two hosts, computation time is only for half the task. Hence, the data needs to be in the memory only for a much lesser time. As the memory residence time reduces, the page fault overhead also decreases. Reduction in these overheads (such as page faults and context switches, if any) and by overlapping computation and communication across hosts, the communication delay seems to be nullified. Thus, super linear speedup is achieved. But, as the number of hosts increases to 8 and 16, the communication delay seems to dominate. This is partly because, multicast simulator runs on a single host and is not distributed. So, a multicast is simulated by multiple unicasts. If multicast is done physically (or a distributed simulator is used), then the communication overheads can be reduced further. Scalability limitation is also due to the problem size. If problem size is increased, with more hosts better speedup could be attained.

Mobile Grid

This section briefs a perspective of the other novel wide area distributed mobile computing paradigm namely, mobile grid, as an integration of mobile computing paradigm into the grid computing paradigm along with service composition technology. It has both advantages of powerful computation capability of grid and ubiquitous accessibility of distributed mobile system. It combines the middleware solution from services composition, resource-sharing solutions of grid computing and the anywhere, anytime resource access of distributed mobile system. The goal is to design a mobile grid middleware model that provides transparency to the users, in addition to handling the key constraints of the participating MHs. The model also addresses critical challenges that arise in the context of composing independent components across multiple service providers. The benefits of this model are information processing capacity increase and resource sharing. This helps even mobile hosts to activate any application, which may need huge computing power to execute a task. The proposed model also provides MH mobility and location as well as management of message delivery.

A. Mobile Grid Architecture

In the proposed mobile grid middleware, object based model is considered for information exchange. This is because the encapsulation of state and its operations into a single object allows a clear separation between the services provided by an object and their implementations. Aspects such as mobility of the representing devices, communication protocols, replication strategies, and distribution and migration of state can be completely hidden behind an objects interface.

The proposed architecture considers each of the participating MHs as unique object, called Surrogate Object (SO) that is maintained in the wired network. In addition, to have a unified design, all other entities are also represented by independent, objects. The SO enables to maintain a cache of the data structures and data associated with its MH, and reduces wireless data transfers. It also provides an ideal placeholder for MH location information, thus solving the location management problem to handle mobility of hosts. Each object has one or more interfaces, with each interface consisting of one or more methods, representing the services offered by the host. By encapsulating the participating hosts, in distributed objects, the grid is transformed from a collection of hosts, offering and consuming services, into an object space of distributed objects. Thus objects are the building blocks of our mobile grid. The proposed mobile grid structured as distributed shared objects (DSO) space [10] as shown in figure 4. The objects communicate with one another using method invocation. Method calls are non-blocking and may be accepted in any order by the called object. Each methods signature describes the parameters and return values (if any) of the method. The complete set of method signatures for an object fully describes that object's interface.

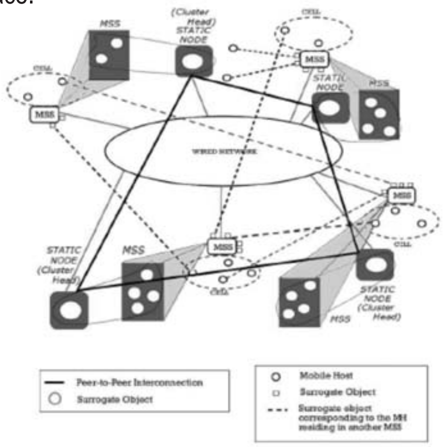
MSS
(Cluster Head) STATIC NODE
CELL
MSS
CELL
MSS
STATIC NODE
MSS
WIRED NETWORK
MSS
STATIC NODE (Cluster Head)
MSS
MSS
CELL
CELL
STATIC NODE (Cluster Head)
MSS
Peer-to-Peer Interconnection
Surrogate Object
Mobile Host
Surrogate Object
Surrogate object corresponding to the MH residing in another MSS

Fig. 4 : Mobile Grid Architecture with Surrogate Objects

The proposed mobile grid is organized as a cluster of clusters. Each cluster is coordinated with a designated host acting as cluster head (CH). The CH handles the cluster management and maintains the required repositories for maintaining the directory of the addresses of the services offered within the cluster. The CH coordinates among the other neighbouring CHs in a peer-to-peer fashion in contrast to the usual hierarchical organization. The MHs using the exporter daemon sends the available services to the MSS, which in turn passes it to the CH. Depending on MSS load conditions, MSS and CH may be configured on same host or in different host.

The proposed model virtualizes all the resources and services offered by the participating hosts as services. When the MH acts as an information service provider, contacting the host directly will lead to consumption of the constrained resources like battery and bandwidth. In this case it would suffice if the corresponding SO is contacted to get the information. Thus the SO model of the mobile grid handles reliability. In addition the SO's can be replicated to avoid the congestion in the network and to improve scalability of the system. With the SO being fully autonomous, users can access services even if the host disconnects because the SO delivers the results upon reconnection.

B. Mobile Grid Computational Model

In the proposed model all the participating hosts will have a DSM runtime object. This object serves as the interface for the client to access the DSM services. When a client request for a service, the DSM runtime object gets a request for creating shared objects, it interacts with the DSM services, namely the lookup and object repository services and returns a replica of the object to the client. The object repository service maintains information about objects and the current accessors for each object. It is responsible for propagating updates to the replicas.

The DSM runtime on each host handles the initial object discovery requests. It looks up the object in its cache, failing which it contacts the lookup service of the cluster in the CH. If the object has not been created before or has been created in a different cluster, the DSM runtime sends a request to the object repository. The object repository creates a new unique oid for the object and gives back a copy of the object. The repository maintains list of current accessors for each object. When an update request message is received from an accessor, it is propagated to all other accessors, the order depending on the application specific consistency criteria.

Horse Power Factor and Metrics for the Services:

As each participating host may have different services capabilities (service includes all resources including memory, processing power, etc.), it is essential to allocate tasks to the SHs and MHs based on their capabilities. The task may involve either a computing task or data which needs memory space for storage, etc. Thus the metrics associated with each service is calculated and advertised in the trading service along with the registration of the services. To incorporate this, each service is allocated integers which refers to the metric, which is dynamic. One such metric is the HPF, which refers to the service that represents the computation capability. The HPF is the same as the one discussed in moset. Abstracting the heterogeneity in this way makes it viable for parallel processing on unevenly loaded heterogeneous machines.

Message Delivery and Node Mobility Management:

Mobile connectivity is highly variable in performance and reliability. The wireless communication channels used by the MHs also have a lower bandwidth than the wired channels. Mobility implies that a MH changes its location. Thus, the location management for the targeted MH becomes an indispensable task of any application that runs on the distributed mobile system. This complicated issue could be handled effectively without affecting the delivery using our proposed model. The sender of a message targeted to an MH need not bother about whether the MH is in motion or out of coverage region. All that needs to be done is to deliver the message to the surrogate object which is residing in the static portion of the network. The surrogate object takes opportune time to deliver the message to its MH considering the availability of the wireless bandwidth and the traffic on the network.

Service Management:

Service Composition refers to the construction of complex services from primitive ones, thus providing rapid and flexible creation of new services. Fixed broker-based or fixed central-entity based service composition techniques cannot be applied as a solution to compose services on the fly as the clients can be mobile. The client may also use a service which may be mobile and connected through wireless network. The architecture facilitates bindings between components (objects providing services) to be discovered or rediscovered at run time. This helps in overall system scalability. Another reason for service discovery is SO mobility, due to the mobility of the host. The details of the SO's are maintained in the CH using two repositories namely trading service and naming service. The trading service returns the object that provides the required service and the naming service gives the object reference of the SO corresponding to the object, using which the location of the object is tracked. When a request is made by a client host for the required service, the SO contacts its cluster head to find out the availability of the service. If the required service is not available within that cluster then peer-to-peer search is done among the cluster heads. Thus the service name space is organized as a Peer-to-peer.

C. Performance Analysis

The performance of the proposed mobile grid model is studied at two levels, namely to study the performance of the new surrogate model and to study the grid model which uses the surrogate object. Inorder to study the surrogate model, query execution was considered and implemented over a simulation model of a cellular network. Figure 5 shows the effect of surrogate object migration on the query latency for various migration frequencies. Travelling salesman problem (TSP) application was considered to study the speedup achieved using the mobile grid model. This experimental model of the mobile grid was a simulation over the actual implementation. The mobile grid was built using the distributed shared object space, namely the virat [10]. Fifty heterogeneous machines from the institute wide network was 10 considered. In order to realize a cellular model, three nodes were designated as MSS, and all the remaining nodes were assumed to be MHs of the cellular network. The MHs were divided into seven groups, and the MHs in each group were made to communicate with any of the other nodes only through there corresponding MSSs. Inorder to realize a mobile scenario the communication from the MH designated nodes were delayed, to correspond, to a wireless bandwidth. The nodes designated as MHs dynamically changed there groups to realize the mobility pattern of the MHs. The out of coverage of the MHs were also simulated dynamically, by making the MHs not to be a member of any of the MSSs.

IV. CONCLUSIONS

Communication paradigm, namely, reliable multicast protocol and computing paradigms, namely, Moset and Mobile grid for distributed mobile systems have been proposed. The focus of this work, in the design of the paradigms was to handle the key constraints of the mobile hosts and its associated issues. The proposed multicast protocol supports exactly-once message delivery and provides totally ordered message delivery with negligible overhead. The multicast protocol also tolerates MSS failures and is scalable.

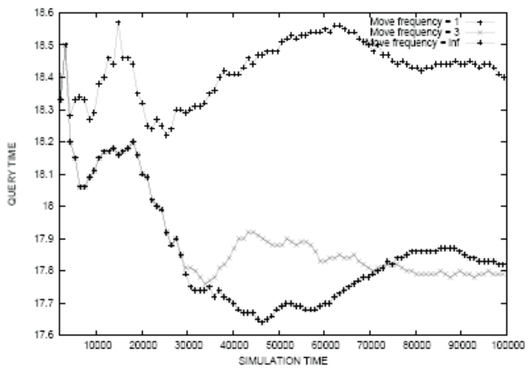

Fig. 5. Effect of SO Migration on the Query Latency

The proposed Moset cluster computing paradigm provides one of the first comprehensive efforts at integrating mobile devices into the cluster and effectively harness the computing power of mobile devices. Further, mobile devices can also utilize large computing power available in the cluster seamlessly. The feasibility of the idea is studied using an image rendering application. The work also includes another computing paradigm, namely the mobile grid, which is done by integrating the mobile devices into the grid to provide a seamless computing environment. The key idea is to integrate the computing, data and service grids. To handle the constraints associated with the participating mobile devices, the mobile grid is designed using object technology, by making the mobile grid middleware a wide area distributed shared object space.

REFERENCES

[1] Anderson T., D. Culler, D. Patterson, and the NOW Team, A case for networks of workstations(now), IEEE Micro, vol. 15, pp. 54-64, Feb 1995.

[2] Chen Y.-F. and C. J. Petrie, Guest editors' introduction: Ubiquitous mobile computing, IEEE Internet Computing, vol. 7, no. 2, pp. 16-17, 2003.

[3] Acharya A. and B. R. Badrinath, A framework for delivering multicast messages in networks with mobile hosts, ACM/Baltzer Journal of Wireless Networks, Special Issue on Routing in Mobile Communication Networks, vol. 1, no. 2, pp. 199-219, 1995.

[4] McCanne S. and S. Floyd, ns-network simulator, 1995.

[5] Zeng X., R. Bagrodia, and M. Gerla, Glomosim: A library for parallel simulation of large-scale wireless networks, Workshop on Parallel and Distributed Simulation, pp. 154-161, 1998.

[6] Anastasi G., A. Bartoli, and F. Spadoni, A reliable multicast protocol for distributed mobile systems: Design and evaluation, IEEE Transactions on Parallel and Distributed Systems, vol. 12, pp. 1009-1022, October 2001.

[7] Joshi R. K. and D. J. Ram, Anonymous remote computing: A paradigm for parallel programming on interconnected workstations, IEEE Transactions on Software Engineering, vol. 25, no. 1, pp. 75-90, 1999.

[8] Singhal M. and N. G. Shivaratri, Advanced Concepts in Operating Systems. McGraw-Hill Inc., 1994.

[9] Watt A. and M. Watt, Advanced Animation and Rendering Techniques: Theory and Practise. Addison-Wesley Publishing Company, 1992.

[10] Srinivas A. V., D. Janakiram, and R. Koti, Virat: An internet scale distributed shared memory system, Tech. Rep. IITM-CSE-DOS-2004-03, DOS Lab, IIT Madras, 2004. Communicated to Concurrency and Computation: Practice and Experience.