Supporting Mobile Multimedia Applications in MAPGrid

Yun Huang, and Nalini Venkatasubramanians
Donald Bren School of Information and Computer Sciences, University of California, Irvine
Irvine, CA, USA
yunh@ics.uci.edu, nalini@ics.uci.edu

ABSTRACT

MAPGrid (Mobile APplications on Grids) system enables the delivery of services to mobile users by exploiting heterogenous and intermittently available grid resources. In this paper, we present techniques of cost-effective resource discovery and dynamic resource reprovisioning on grid proxies when context of system environment (e.g. application, network, resource, mobile device) change in infrastructure based wireless networks (e.g. cellular or WLAN). Especially, we focus on how to adapt to dynamic changes of user mobility patterns. We show that the MAPGrid system provides an infrastructure support for enabling and effectively enhancing mobile applications.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

Performance

Keywords

Mobile computing, grid computing

1. INTRODUCTION

Mobile multimedia applications are becoming ubiquitous and are projected to be the dominant applications in next generation mobile systems. These applications have distinctive performance and processing requirements which tend to make them extremely resource hungry. They also have diverse Quality of Service (QoS) requirements that determine the utility of the (perceived) information to the end-user. Due to resource constraints in mobile devices and dynamic conditions in wireless networks, achieving sustained QoS between the service provider (server) and mobile device is difficult. This has brought about an enhanced interest in infrastructure support for mobile multimedia applications. A recent trend is to utilize resources in the path of service, i.e. network proxies to provide localized computation and storage to enable distributed and ubiquitous service availability. The proxy approach attempts to use available resources in the wired networks within close proximity to the mobile device to support strategies such as proxy caching [22], proxy-based transcoding [20] or task-offloading [21] that can alleviate stringent resource needs of mobile multimedia applications. However, these intermediate resources must be discovered and deployed effectively. We have been developing a mobile grid infrastructure called MAPGrid (Mobile APplications on Grids), where grid resources are used as proxy servers to enable mobile multimedia applications [2].

We argue that our proxy-based resource scheduling and data placement approaches take into account of the intermittently availability of grid proxies [10, 11] that other proxy-based solutions fail to consider. Research efforts of grid computing technologies have focused on developing and implementing computation task scheduling [23] and data replication [24] techniques for computational-intensive and data-intensive scientific applications. Directly utilizing such techniques to implement the above proposed MAPGrid system will not lead to optimal performance [13, 15], because they do not account for mobility of users and hence are unable to support mobile multimedia applications.

In this paper, we discuss static and dynamic aspects of the resource allocation problem, i.e. selecting optimal grid proxies to service mobile requests. We present how dynamic adaptation can sustain QoS guarantees under changing network, device and system conditions in a cost effective manner. We argue that context (application, network, resource, device) plays a crucial role in effective resource discovery for mobile applications. We illustrate how to integrate the dynamic changes non-intrusively into a wide-area mobile infrastructure for enhancing mobile multimedia applications.

In this paper, QoS-aware mobile applications are supported by a generalized architecture as follows. A context repository keeps information of system environments (e.g. network resources, availability of grid proxies). A request containing QoS parameters is initiated at a mobile device. The broker utilizes the resource availability information stored in the context repository (maintained by an adaptive context collection module) to decide an optimal allocation of network, server or grid proxy resources to service this mobile request. Grid proxies can be volunteer servers (*VS*s) from the network. Significant changes in the availability of the allocated *VS*s will trigger the resource reprovisioning process to adapt the resource allocation accordingly.

IWCMC'07, August 12–16, 2007, Honolulu, Hawaii, USA.

When the request terminates, the resources are reclaimed along the connection. The objective of our developed techniques for MAPGrid system is to support diverse QoS requirements of mobile applications while improving overall system performance in terms of user QoS experience, client admission ratio, grid throughput and grid utilization.

2. GRID RESOURCE DISCOVERY AND RESOURCE REPROVISIONING FOR MOBILE APPLICATIONS

In the MAPGrid system,a mobile user initiates a multimedia request, *R<VID,T,itinerary(opt)>*, where *VID* identifies the requested video object, *T* represents the whole service period, and the *itinerary* contains user's mobility information (NULL, if no mobility information is available). Given mobile requests and information of grid resource availabilities, the static resource discovery attempts to increase the overall acceptance of requests in the system by selecting optimal grid resources for each mobile request, while satisfying users' QoS requirements. QoS needs can be expressed as user-perceived quality needs (e.g. video quality) that are translated into lower level application/system parameters. The QoS requirements e.g. required network transmission bandwidth, will be determined by streaming a certain video streaming object. QoS statements may specify constraints on timing, availability, security and resource utilization at various levels of abstraction. For instance, timing based QoS requirements can be specified using abstract properties such as correct/timely data delivery and uninterrupted service. These properties can be translated to concrete application parameters such as jitter, end-to-end delay, synchronization skew and/or concrete resource requirements such as network and disk bandwidth and buffer requirements [26]. Resources required to support these multidimensional notions of QoS in mobile applications can be in form of computation (CPU), storage, bandwidth, memory or services that must continue to be available as the user moves in the mobile infrastructure.

2.1 Mobility-driven Grid Resource Reprovisioning

In our previous work, we proposed a phased approach [14] that divides the whole service period T into non-overlapping chunks (possibly of different sizes), each of which is mapped to an appropriate grid proxy (VS) e.g. the one that is geographically close and lightly loaded. A graph theoretic technique is developed [12] for selecting an optimal set of grid resources to service each chunk. Decisions are made by considering all the following factors: (a) intermittent availability of *VS*s, (b) currently allocated workloads and predicted future workloads on *VS*s, and (c) user's distance to *VS*s. The problem of discovering intermittently available *VS*s is solved as a maximum flow problem. A feasible maximum flow solution that meets resource constraints corresponds to a possible scheduling solution; the basic solution has been adapted to develop a family of policies that cater to various application QoS needs. Video objects are also divided into equal-sized segments. Corresponding video segments are downloaded onto selected grid proxies. The selected *VS*s process the request by transcoding the video segment and transmits the video stream via wireless links to mobile clients. We have shown [14] that the above approach effectively increases client admission ratio, improves grid resource utilization and reduces network transmission cost.

However, it is difficult to accurately discover and provision resources using static methods for the entire duration of a service, especially when there is no prior knowledge of how a user moves during service time. In general, randomness of client mobility will not affect the service completion ratio once the request has been scheduled, since the scheduled *VS*'s are connected to the access point (base station) of the mobile client via a wired network. However, due to changes in client itinerary, the pre-assigned *VS* (originally within close proximity to the mobile host) may no longer be optimal; increased path lengths can introduce additional delays, jitter and network traffic. To adapt to dynamic mobility changes, we devise a distance-based approach to enable mobility-based rescheduling that determines a new reassignment between a *VS* and the mobile client *MC* during its service time. The mobility-based rescheduling process determines whether a reconfiguration is needed and when to select a new *VS* for the request. We define a benefit factor:

$$benefit = \frac{dist(newAvailableVS, currentMClocation)}{dist(preselectedVS, currentMClocation)}. \quad (1)$$

When the benefit factor value exceeds a predefined threshold, a *VS* rescheduling is triggered. For example, a threshold value of 0.1 implies that if the distance between the MC and a new available *VS* is smaller than 10% of the distance between the MC and the pre-selected *VS*, then a new rescheduling process can be triggered. The frequency of rescheduling and the selection of a threshold value are affected by many factors, such as: locations of available *VS*s, clients' mobility patterns, etc. In section 3, we will present our studies on how different threshold values affect frequency of rescheduling events. We aim to make rescheduling decisions that help relocate closer grid proxy resources to service the mobile request when its itinerary changes, while avoiding frequent changes of service connections (reducing the number of possible rescheduling) in future movements.

However, dynamic rescheduling requests on new proxies may require those proxies to replicate/cache requested data of mobile clients. This may cause unexpected overload situations that deteriorate system performance. An overload is defined as a situation when there is insufficient proxy cache for a new incoming user. In the next section, we will address how to reduce overloads by applying prediction techniques.

2.2 Overload-driven Mobility-aware Caching on Grid Proxies

In [25], we proposed proactive caching techniques that determine how to allocate proxy cache periodically to avoid proxy overloads in the next period. In [15], we also developed an intelligent data placement technique for placing mobile objects on intermittently available grid proxies. We presented a two-tier architecture, where the upper tier captures grid related features, e.g. intermittent availability of grid proxies and the lower tier captures features associated with the mobile environment, e.g. data request patterns of mobile users. By exploiting the knowledge of client mobility patterns, device energy profiles and grid resource availability, we determine the localized computational and storage resources within the grid for enhancing overall user experiences for mobile applications [11,13]. The above tech-

niques are executed periodically based on prediction of system workload for the next period of time. However, significant workload variations can be caused by dynamic changes to user mobility and request access patterns. Therefore, when a mobile client's request is rescheduled to a grid proxy and a certain amount of cache space is requested, the decision on how much cache space of the proxy can be issued to this request should be based on a number of factors: e.g. current available cache space of this proxy, the number of total serviced mobile hosts by this proxy, and the predicted number of overloads calculated at the end of last period.

In order to achieve fairness, we differentiate four types of conditions and deal with them separately as the follows. Case 1: When at the end of last period, it was predicted that there would be no overloads in the current period of time, on-demand dynamic cache adaptation will give as much cache space of the proxy as possible to this new request. Case 2: If a certain number of overloads are predicted in this period, but there is enough cache space available for a new mobile client's request, then the proxy will still give the requested cache space to this mobile client. On the other hand, if there is no enough cache space for this new request, then to be fair, on-demand dynamic cache adaptation will try to give this new request the average cache size that would be given to all other mobile clients. This is calculated as *CAVE(m)*, the average cache size for m mobile clients who are currently accessing the proxy cache. Comparing *CAVE(m)* and current available free cache space (Cava) will lead two other situations, case 3 and case 4. Case 3: when the current available proxy cache is larger than *CAVE(m)*, each request will get cache space depending on the number of future predicted overloads in that time period. A new average cache space value is calculated. If there have already been a larger number of overloads happened than that was predicted, we will not take the overloads predicted in the last period into considerations. Case 4: when current available proxy cache is smaller than *CAVE(m)*, we regard this request as a new overload, we still try to give the amount of cache space calculated by case 3, but we may need to release some cache space to accommodate all requests. The procedure of releasing cache space is based on the increasing order of data access frequency and the decreasing order of data object sizes. In section 3, we will show that an integration of the above dynamic caching approach and our previously proposed proactive caching can significantly reduce proxy overloads.

3. PERFORMANCE EVALUATION

In this section, we present and evaluate our simulation results. System performance of resource discovery and reprovisioning strategies are studied under various *VS* configurations and host mobility patterns. We simulate a cellular system with 100 cells. The basic video server configuration used in this simulation includes 50 grid proxies (*VS*s) distributed evenly in an area of cellular network; each with a storage of 100 GB and network transfer bandwidth of 100Mbps. We set the duration of each video that ranges from 30 minutes to 2 hours; network transmission bandwidth ranges from 500 kbps to 2 Mbps. The shortest segment of a video object can run for 10 minutes. We characterize incoming mobile requests to the broker by using a Zipfian distribution [16]. This request pattern determines the number of requests that will be issued to different data objects by users over time.

We model availability of grid proxies via notion of time maps, which represents the available resources of the proxies at different time intervals. Specifically, we use the service time map for *VS*s to keep the information of when the *VS* will be available during the span of a day. We study three approaches to model the time map of each server. (1) Uniform availability: all *VS*s are available (or unavailable) for an equal amount of time; and the *VS*s are divided into groups, such that within each group, the time distribution covers the entire 24-hour day. In our simulations, we use the uniform availability with duration set to 6 hours as the basic time map strategy. We also executed the entire set of experiments with a variety of continuity or total availability for time maps. (2) For random availability, we randomly choose the number of hours when a server is available for each day. The time at which a server is available is also randomly chosen on a per-hour basis; hence the available times may be continuous, discontinuous or partially continuous. (3) Total availability: all *VS*s are available all the time.

Our studies focus on three host mobility patterns shown in Fig. 1: (a) Random movement: the mobile node will travel at any speed, heading in any direction. (b) Constant velocity with intermediate halt: a linear straight moving pattern with one long stop where the node remains stationary. (c) Clustered movement: there exist more than two cell clusters between which the mobile client moves back and forth; the clusters are geographically distant.

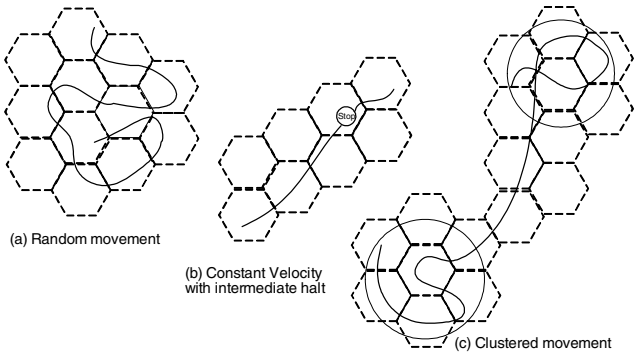

Figure 1: Three main moving patterns

To characterize random mobility of individual nodes, we use the incremental mobility model [6], where mobile hosts are distributed randomly and move freely in a closed coverage area. The movement of the mobile host is represented by its velocity vector $\overrightarrow{v} = (v, \theta)$, where v is the speed and θ is its direction. The location of the mobile host (x, y) and its velocity are updated periodically every δt time units as follows: $\theta(t + \delta t) = \theta(t) + \delta\theta$ and $v(t + \delta t) = \min[\max(v(t) + \delta v, 0), V_{\max}]$. V_{max} is the maximal mobile velocity; δv is the velocity change which is uniformly distributed within $(-A_{max} \cdot \delta t, A_{max} \cdot \delta t)$, A_{max} is the maximum acceleration or deceleration. $\delta\theta$ is the change in the mobile client's direction and uniformly distributed in $(-\alpha \cdot \delta t, \alpha \cdot \delta t)$, where α is the maximal angular change of the mobile host's direction per unit time. By adding constraints on the parameters of the incremental mobility model, we generate other mobility patterns. Also, we set A_{max} as 85 mph (135kph), so that the speed varies between a car driving velocity and a steady entity (0kph), and vary the size of the cells from 100m to 20km [17]. Given space limitation, for each of the

following study, we only show results for one of the above three mobility models. In fact, results under the other two mobility models can draw the same conclusions.

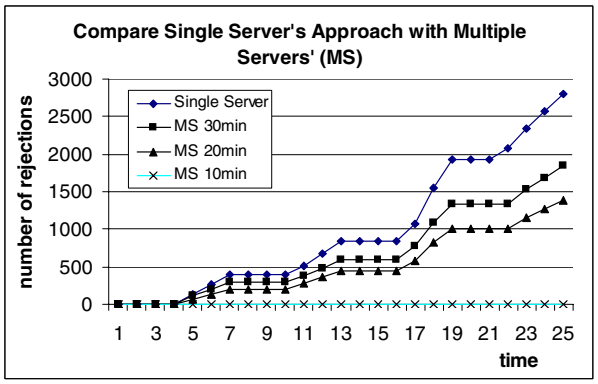

Figure 2: Performance of selecting multiple grid proxies for mobile services

Robustness to changes in mobility: We compare the traditional single server approach (where only one proxy services the whole request duration) and multiple servers' approach (when multiple grid proxies are selected to service a mobile request). As shown in Fig. 2 (under random movement), the multiple servers' approach significantly increases acceptance ratio in the system. As shown in Table 1, it also reduces the distance between the selected *VS* and mobile client when sizes of smallest data partitions are reduced.

Table 1: The average distance between the selected *VS* and the mobile host when service is partitioned in different sizes

The size of partitioned segment (in minutes)	The average distance between the selected *VS* and the mobile client
30	1.121927
20	1.090445
10	1.020738

We study the effect of distance-based mobility tracking process. As Fig. 3 illustrates, the number of rescheduling processes will differ significantly under different threshold values. This result suggests that if we want to achieve less than two average rescheduling events for a single request, the threshold hold value should be less than 0.3 (30%).

Impact of the number of mobile hosts: In this experiment, we evaluate the sensitivity of caching adaptations to the number of mobile hosts. Fig. 4 (under constant velocity with intermediate halt model) shows that when the traffic gets heavier, cache adaptations will be more effective when reducing the number of overloads. For instance, when there are 400 mobile clients in this simulation environment, cache adaptation policies can reduce 30% of the overloads compared to the on-demand LRU caching technique. But when there are 1500 mobile clients, our caching approach can reduce 70% of the overloads.

Impact of proxy cache size: Fig. 5 shows how proxy cache capacity affects caching performance. Obviously, given the same system traffic loads (the same number of mobile hosts), increasing proxy cache sizes can significantly reduce

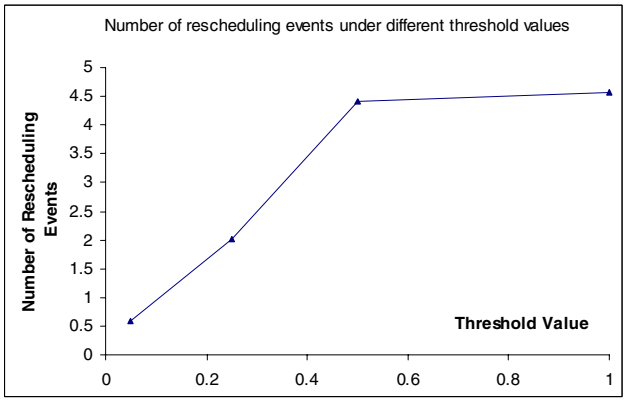

Figure 3: The number of rescheduling events under different threshold values

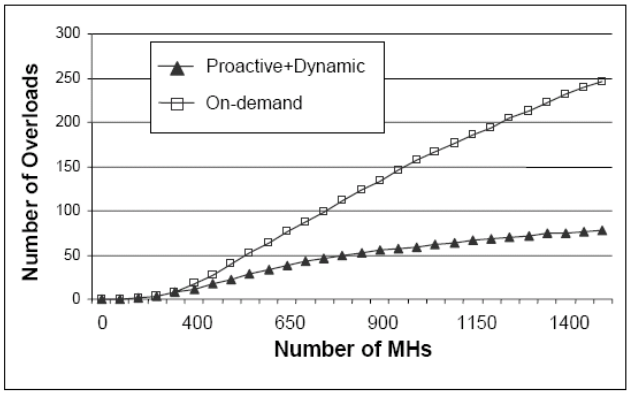

Figure 4: Performance of proxy caching under different number of mobile hosts

system overloads with either caching policy. However, when each proxy has limited cache capacity (e.g. less than 40MB) the combined caching policy (proactive and dynamic together) successfully prevent leaps in system overloads.

Impact of grid availability patterns: Fig. 6 presents the impact of various time map models on our proposed resource allocation approach. It compares the Faststartup (a type of multiple-server approach) and Single *VS* policy proposed in [14] when in random movement mobility pattern. We observe that Faststartup policy has a much higher number of completed requests as compared to a Single *VS*. This is especially true when the total *VS* availabilities (time for which each *VS* is up) is small; similar results are observed when *VS* time continuities (period for which a *VS* is continuously available) is small.

We also study the impact of cell sizes and host velocities on different policies. Given space limitation, we are not able to show all results. In summary, high speeds facilitate easy partitioning of a service period. Our proposed adaptations are not sensitive to variations of cell size and host velocity. They are resilient to dynamic changes in mobility patterns and can significantly improve request completion ratios under unpredictable system conditions.

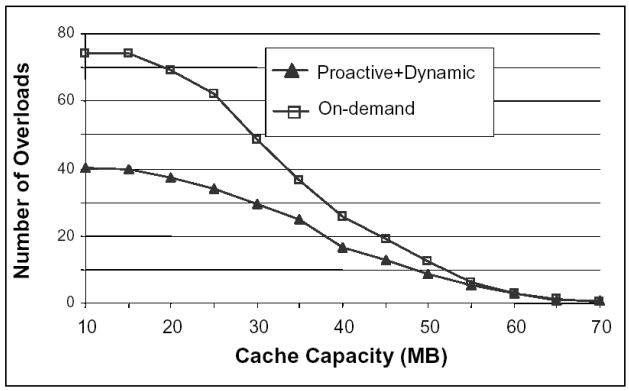

Figure 5: Performance of proxy caching under different cache capacities

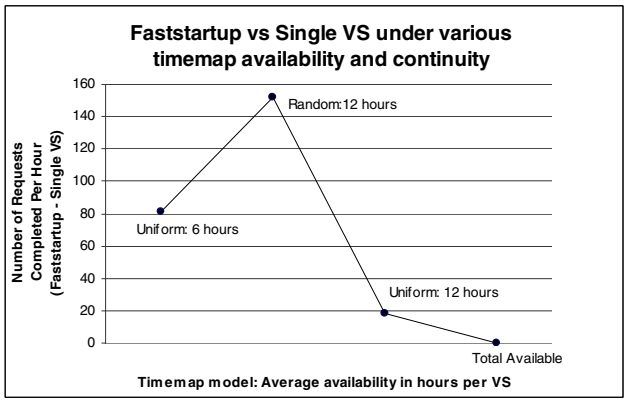

Figure 6: Impact of grid availability patterns

4. RELATED WORK AND FUTURE PLANS

The InviNet project [19] presents an architecture to address issues of mobile access to grids. More projects are seeking solutions for enabling grid-based mobile applications or resource sharing among mobile hosts in grid infrastructure, e.g. K*Grid [7], AKOGRIMO [27], MoGrid [4], Mobile OGSI.NET [3] and the wireless grid project [18]. GridLab [5] project develops *Grid Application Toolkits* (GAT) that allows access from J2ME-enabled mobile devices to grid services. Compared to other mobile/wireless grid projects, we focus on developing efficient resource management solutions for resource discovery and data placement by leveraging context information of mobile hosts, e.g. mobility, energy profiles and intermittently availability of grid resources.

Our MAPGrid project addresses how to provide effective Quality of Service (QoS) for mobile applications by leveraging intermittently available grid resources. We have applied techniques from graph theory, neural nets, etc. to deal with quality-aware discovery of grid resources for QoS-based mobile applications (targeting streaming multimedia applications). We develop an intelligent mobile data placement mechanism on grid proxies that effectively balances tradeoffs between replication cost and data access cost by applying knowledge of grid availability and data request patterns of mobile users [15]. We also proposed an integrated solution that adapts to dynamic changes in device energy consumption and unpredictable grid resource availability without compromising application QoS [8]. In this paper, we focus on addressing dynamic grid resource reprovisioning for mobile applications when user mobility pattern changes.

The initial prototype of MAPGrid assumes a WLAN infrastructure. We built the prototype system over a network of volunteer desktop machines located at different buildings at UCI. In [9], we address practical issues of implementing MAPGrid system. We introduce a comprehensive and versatile system design, where fine grained and well-defined meta-level services enable flexibility in easy plug-in of different policies. The underlying implementation of MAPGrid subsumes a multithreading model for multitasking the various models for resource discovery and data placement services, which scales well in performance. Our design and implementation incorporate concerns for synchronization, flexibility and easy plugging-in of these policies. The mobile client uses either a Sharp Zaurus handheld device that runs on Linux or a windows laptop. To allow for easy extensibility and re-use, the prototype MAPGrid implementation uses the Globus Toolkit, along with standardized messaging interfaces such as XML for metadata exchange.

We designed a mobile application for testing our system performance. We implemented a video stream application tested on the laptop computer, where video clips are displayed by JMF (Java Media Framework) at laptop mobile clients. When mobile user moves among wireless regions, data service connection will be changed to closer proxy machine as scheduled by the broker, i.e. downloading file from nearby proxies during the service period. Our download process has a reduced switch-over time compared to that of without changing proxy connection. Fig. 7 shows the GUI for a mobile client to select his preferred service type [11].

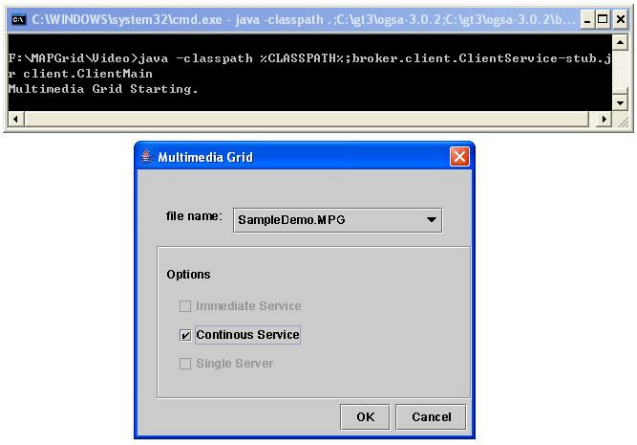

Figure 7: The MAPGrid video application

We also implemented other grid-based mobile applications. For example, requests from mobile handheld users, as shown in Fig. 8, are scheduled onto nearby proxies. Based on one user's location, nearby grid proxy customizes a local map to a proper size with this user's and other users' locations. Fig. 9 shows a XML snippet for defining the cropping map service of the proxy. We are currently integrating proxy-based techniques developed by DYNAMO project [1] into MAPGrid. We believe our MAPGrid system will lead to promising applications for mobile grid environments.

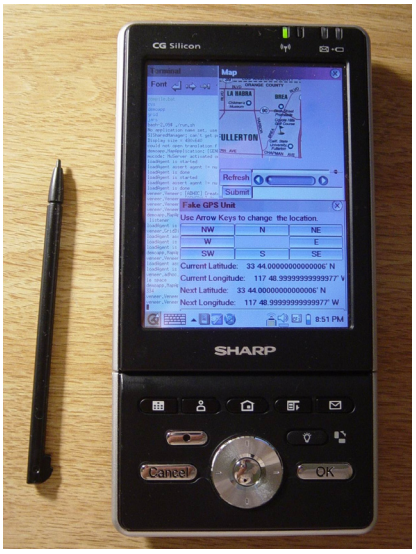

Figure 8: A customized map application on mobile handheld devices

```
<types>
<xsd:schema targetNamespace="http://mapgrid.ics.uci.edu/namespaces/0.2/core/gwsdl/VSService"
    attributeFormDefault="qualified"
    elementFormDefault="qualified"
    xmlns="http://www.w3.org/2001/XMLSchema">

    <xsd:element name = "cropImage">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name = "imageData" type = "xsd:base64Binary" />
                <xsd:element name = "mapLong1" type = "xsd:double" />
                <xsd:element name = "mapLat1" type = "xsd:double" />
                <xsd:element name = "mapLong2" type = "xsd:double" />
                <xsd:element name = "mapLat2" type = "xsd:double" />
                <xsd:element name = "clientLong" type = "xsd:double" />
                <xsd:element name = "clientLat" type = "xsd:double" />
                <xsd:element name = "width" type = "xsd:int" />
                <xsd:element name = "height" type = "xsd:int" />
                <xsd:element name = "quality" type = "xsd:int" />
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>

    <xsd:element name = "cropImageResponse">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name = "mapData" type = "mapdata:MapData" />
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>

</xsd:schema>
</types>

<message name="CropImageInputMessage">
    <part name="parameters" element="tns:cropImage"/>
</message>
<message name="CropImageOutputMessage">
    <part name="parameters" element="tns:cropImageResponse"/>
</message>

<gwsdl:portType name="VSServicePortType" extends="ogsi:GridService">
    <operation name="cropImage">
        <input message="tns:CropImageInputMessage"/>
        <output message="tns:CropImageOutputMessage"/>
        <fault name="Fault" message="ogsi:FaultMessage"/>
    </operation>
</gwsdl:portType>
```

Figure 9: XML snippet for defining the Cropping Map Service of the MAPGrid Proxy (using Globus Toolkits 3.2)

5. ACKNOWLEDGEMENTS

MAPGrid project is sponsored by Office of Naval Research (ONR) as part of the DoD Multidisciplinary University Research Initiative (MURI) Project CONTESSA.

6. REFERENCES

[1] The dynamo project, http://dynamo.ics.uci.edu.
[2] Mapgrid project, http://mapgrid.ics.uci.edu.
[3] D. Chu and M. Humphrey. Mobile ogsi.net: grid computing on mobile devices. In *Grid Computing, Fifth IEEE/ACM International Workshop*, 2004.
[4] L. dos S. Lima, A. T. A. Gomes, A. Ziviani, M. Endler, L. F. G. Soares, and B. Schulze. Peer-to-peer resource discovery in mobile grids. In *MGC '05: Proceedings of the 3rd international workshop on Middleware for grid computing*, pages 1–6, New York, NY, USA, 2005. ACM Press.
[5] P. Grabowski, B. Lewandowski, and M. Russell. Access from j2me-enabled mobile devices to grid services. In *Mobility Conference in Singapore*, 2004.
[6] Z. Haas. A new routing protocol for the reconfigurable wireless networks, 1997.
[7] K. P. http://gridcenter.or.kr/.
[8] Y. Huang, S. Mohapatra, and N. Venkatasubramanian. An energy-efficient middleware for supporting multimedia services in mobile grid environments. In *ITCC '05: Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05) - Volume II*, pages 220–225, Washington, DC, USA, 2005. IEEE Computer Society.
[9] Y. Huang and N. Venkatasubramanian. Design and prototype implementation of mapgrid: A grid system for mobile applications. submitted for publication.
[10] Y. Huang and N. Venkatasubramanian. Data placement in intermittently available environments. In *HiPC*, pages 367–376, 2002.
[11] Y. Huang and N. Venkatasubramanian. Qos-based resource discovery in intermittently available environments. In *HPDC*, 2002.
[12] Y. Huang and N. Venkatasubramanian. Qos-based resource discovery in intermittently available environments. Jul 2002.
[13] Y. Huang and N. Venkatasubramanian. Supporting mobile multimedia services with intermittently available grid resources. In *HiPC*, 2003.
[14] Y. Huang and N. Venkatasubramanian. Supporting mobile multimedia services with intermittently available grid resources. In *Proc. of HiPC*, 2003.
[15] Y. Huang, N. Venkatasubramanian, and Y. Wang. Mapgrid: A new architecture for empowering mobile data placement in grid environments. In *CCGrid Workshop on Context-Awareness and Mobility in Grid Computing*, 2007.
[16] M. Karlsson and C. Karamanolis. Choosing replica placement heuristics for wide-area systems. In *ICDCS '04: Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04)*, 2004.
[17] E. Lawrey. Ofdm verses cdma, 1997.
[18] L. McKnight, M. Gaynor, J. Hwang, J. Park, H. Chang, A. Gupta, B. Plattner, J. Howison, P. Aravamudham, O. Uzuner, and B. rong Chen. Wireless grid issues. June 2003.
[19] M. Migliardi, M. Maheswaran, B. Maniymaran, P. Card, and F. Azzedin. Mobile interfaces to computational, data, and service grid systems. *SIGMOBILE Mob. Comput. Commun. Rev.*, 6(4):71–73, 2002.
[20] S. Mohapatra, R. Cornea, N. Dutt, A. Nicolau, and N.Venkatasubramanian. Integrated power management for video streaming to mobile handheld devices. In *ACM Multimedia*, 2003.
[21] S. Mohapatra and N. Venkatasubramanian. "PARM: Power-Aware Reconfigurable Middleware". In *ICDCS-23*, 2003.
[22] A. Singh, A. Trivedi, K. Ramamritham, and P. Shenoy. Ptc: Proxies that transcode and cache in heterogeneous web client environments. *World Wide Web*, 7(1):7–28, 2004.
[23] A. Takefusa, S. Matsuoka, O. Tatebe, and Y. Morita. Performance analysis of scheduling and replication algorithms on grid datafarm architecture for high-energy physics applications. In *HPDC*, 2003.
[24] A. Takefusa, O. Tatebe, S. Matsuoka, and Y. Morita. Performance analysis of scheduling and replication algorithms on grid datafarm architecture for high-energy physics applications. In *HPDC*, pages 34–47, 2003.
[25] H. Topcu-Altintas, Y. Huang, and N. Venkatasubramanian. Overload-driven mobility-aware cache management in wireless environments. In *Mobile and Ubiquitous Systems: Networking & Services 2006 Third Annual International Conference*, 2006.
[26] N. Venkatasubramanian, C. Talcott, and G. A. Agha. A formal model for reasoning about adaptive qos-enabled middleware. *ACM Trans. Softw. Eng. Methodol.*, 13(1), 2004.
[27] S. Wesner, T. Dimitrakos, and K. Jeffrey. Akogrimo - the grid goes mobile. October 2004.