

Design space exploration for providing QoS within the HARMONY framework

Citation for published version (APA):

Lele, A. M., Nandy, S. K., & Epema, D. H. J. (2000). Design space exploration for providing QoS within the HARMONY framework. In Proceedings of the IEEE International Conference on Multi-Media and Expo (ICME 2000, New York NY, USA, July 30-August 2, 2000. Part I) (pp. 521-524). Institute of Electrical and Electronics Engineers. DOI: 10.1109/ICME.2000.869653

DOI:

[10.1109/ICME.2000.869653](https://doi.org/10.1109/ICME.2000.869653)

Document status and date:

Published: 01/01/2000

Document Version:

Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

[Link to publication](#)

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

www.tue.nl/taverne

Take down policy

If you believe that this document breaches copyright please contact us at:

openaccess@tue.nl

providing details and we will investigate your claim.

DESIGN SPACE EXPLORATION FOR PROVIDING QoS WITHIN THE HARMONY FRAMEWORK

A.M. Lele and S.K. Nandy

Supercomputer Education and Research Center
Indian Institute of Science
Bangalore - 560012 INDIA
e-mail: {abhijit,nandy}@serc.iisc.ernet.in

D.H.J. Epema

Dept. of Computer Science
Delft University of Technology
Delft, The Netherlands
e-mail: epema@cs.tudelft.nl

ABSTRACT

The HARMONY architecture [2] provides mechanisms for management of network and compute resources in a mobile computing environment. In HARMONY network resources are reserved based on the *Entropy Model* [7], and compute guarantees are provided by off-loading tasks from the mobile units to compute servers in the backbone network. A load-balancing scheme redistributes loads across all compute servers so that these are equally loaded. This paper carries out a design space exploration of the HARMONY architecture to determine parameter bounds within which quality of service can be provided.

1. INTRODUCTION

It is envisioned that future *Mobile Computing Environments (MCE)* supporting multimedia applications will have wearable front-end computing devices and various I/O servers, database servers and compute servers (CS) on a backbone network. Users of the Mobile Units (MU) in such an MCE want to run ever more demanding applications, but they don't want to have the MUs increase in size and weight due to more battery power. A solution to this problem is to off-load parts of applications from the MUs to the CSs in the backbone. This off-loading introduces the need for stringent Quality of Service (QoS) requirements in the CSs, in addition to the QoS requirements for the wireless network connections. With a closer integration of communication and computation in multimedia applications, their end-to-end performance increasingly relies on the combination of network and compute QoS guarantees. A mobile multimedia system under investigation in which we are considering off-loading applications to the backbone, is the *Ubiquitous Communications* project at the Delft University of Technology, The Netherlands [1]. The HARMONY architecture proposed in [2] provides QoS in an MCE and helps maintain harmony between communication and compute resources. In this paper we carry out a design space exploration of the

various architectural components in HARMONY to identify a feasible region in which QoS is guaranteed.

Previous work on providing compute guarantees has concentrated on over-provisioning compute capacity [3]. In [4] a scheme of mapping applications from the MUs to the CSs has been investigated from the perspective of reducing power consumption. A good survey of existing QoS architectures appears in [5]. However, none of the cited architectures in [5] guarantee end-to-end network and compute QoS, whereas the HARMONY architecture proposed in [2] has mechanisms to provide end-to-end network and compute QoS.

2. SYSTEM MODEL

The system model that we consider is an MCE in which network and compute resources can be reserved. In this section we provide details of the mechanisms to achieve this.

2.1. Mobile Computing Environment

We consider a cellular MCE with hexagonal cells comprising a two-level hierarchical network as shown in Figure 1. The first level of the hierarchy is the wireless network between the MUs and the *Base Stations (BS)*. At the second level of the hierarchy, a wireline backbone network interconnects the BSs and the CSs. In this paper it is assumed that the backbone network is an *Asynchronous Transfer Mode (ATM)* network with a peak transmission rate of 645Mbps [6]. A group of four cells forms a *macro-cell* as shown by bold lines in Figure 1. For instance, cells c_5, c_6, c_{15}, c_{16} together constitute a macro-cell. A CS is associated with every macro-cell. All CSs have the same compute capacity C , which is the MIPS (*Million Instructions Per Second*) rating of the associated processor. We say that a cell c' is a *First Order Neighbor (FON)* of cell c if they have a boundary in common. A cell c' is a *Second Order Neighbor (SON)* of cell c , if there exists a cell c'' which is FON of c and c' , and if c

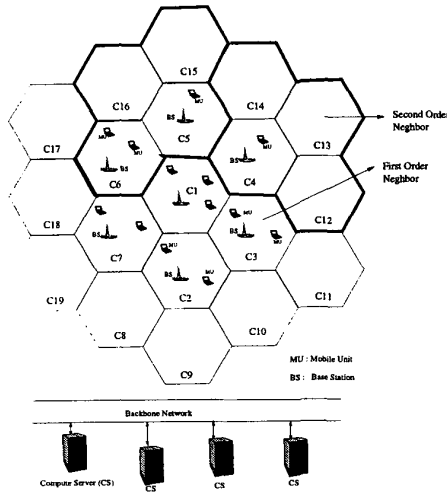


Figure 1: Mobile Computing Environment

and c' are not FONs of each other. Clearly, each cell has 6 FONs and 12 SONs.

2.2. Network Resource Reservation

Providing network guarantees to an MU involves estimating the network traffic generated by the MU, and reserving resources based on this estimation. Network guarantees are provided based on the *Entropy* model proposed in [7] over ATM networks. A synthetic traffic profile with the same statistical properties as those of traffic in a wireless environment is generated by modulating traffic in wireline domain with *white gaussian noise*. This synthetic traffic is modeled using the *Entropy* model with 21 states of unequal extents in a Markov chain [7]. Each state in the Markov chain represents a bandwidth regime. The variation in bandwidth required for the duration of the call is obtained by traversing the most probable path in the Markov chain.

When an MU originates a call in cell c , it wants to establish a network QoS contract Q_n comprising its required bandwidth B , its required buffer space S in the BS, and its maximum network latency T_{delay} . As MUs are mobile, we may want to reserve these resources not only in cell c , but also in c 's FONs, in its SONs, and perhaps in additional cells. Therefore, we allow the MU to subscribe to the following Network QoS classes, where ρ_p and ρ_s ($0 \leq \rho_s \leq \rho_p \leq 1$) are the *Primary* and *Secondary Resource Usage Factors* and are indicative of the network resources reserved in FONs and SONs. This helps maintain QoS as MUs move from one cell to another.

1. **Mobility Independent Guaranteed** : Q_n is guaranteed in all cells of the MCE.

2. **Mobility Independent Predictive** : Q_n is reserved in cell c , $\rho_p \times Q_n$ is reserved in the FONs of c , and $\rho_s \times Q_n$ is reserved in all remaining cells.
3. **Mobility Dependent Guaranteed** : Q_n is reserved in cell c , $\rho_p \times Q_n$ is reserved in the FONs of c , and $\rho_s \times Q_n$ is reserved in the SONs of c .
4. **Mobility Dependent Predictive** : Q_n is reserved in cell c and $\rho_p \times Q_n$ is reserved in the FONs of c .

If any of the reservations required for the Network QoS class to which the MU wants to subscribe cannot be made when the call is initiated, the call is blocked. After the call is allowed into the MCE, every time the MU crosses a cell boundary, new resources need to be reserved and old ones relinquished according to its Network QoS Class. If QoS resources cannot be acquired in the new cell and its associated FONs and SONs the call is blocked.

2.3. Compute Resource Reservation

When an MU originates a call, it wants to establish a compute QoS contract Q_c consisting of the required compute capacity of the application C_{app} and the required compute latency t_{comp} . We also assume that a call specifies its call duration T_{call} . Similarly MUs subscribing to one of the four Network QoS classes, can subscribe to one of two *Compute QoS Classes*, viz. the *Guaranteed Available* and the *Predictive Available* class. In the *Guaranteed Available* class, Q_c is guaranteed to the MU throughout the duration of the call T_{call} . In the *Predictive Available* class, no guarantees are provided at all, and only best effort compute power is delivered. The *Predictive Available* class can be supported by reserving a portion \hat{C} of the total compute capacity C of every CS, thus reducing the total compute capacity for the *Guaranteed Available* class to $C - \hat{C}$. In this paper we do not consider MUs subscribing to the *Predictive Available* class, and hence, $\hat{C} = 0$.

We categorize the spectrum of compute requirements of tasks into some fixed number of *Compute Levels*, each of which is entitled to a fair share of the total compute capacity in the CSs. A load-balancing scheme that redistributes tasks across CSs within compute levels ensures fairness in the entire MCE. Let N denote the number of compute levels in each CS. The total compute capacity C_Q associated with any compute level is $C_Q = \frac{C}{N}$. Compute level i is defined by the maximal capacity C_i , with $C_i < C_{i+1}$ for $i = 1, \dots, N - 1$. A task that specifies a capacity of C_{app} belongs to compute level i if

$$C_{i-1} < \left\lceil C_{app} \times \frac{1}{t_{comp}} \right\rceil \leq C_i \quad i = 1, \dots, N \quad (1)$$

(We don't consider tasks with $\left\lceil C_{app} \times \frac{1}{t_{comp}} \right\rceil > C_N$.) The number of slots M_i in the task queue of compute level

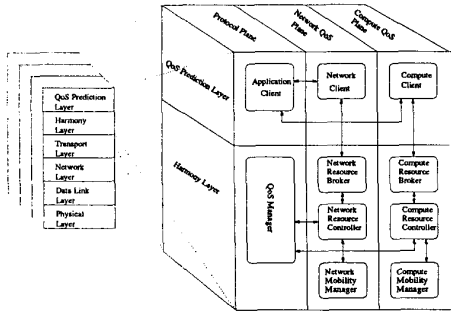


Figure 2: Block Diagram of the Harmony Architecture

i in each CS is $M_i = \lfloor \frac{C_i}{C} \rfloor$, $i = 1, \dots, N$. The service time T_i to serve a slot in compute level i is $T_i = \frac{C_i}{C}$, $i = 1, \dots, N$. When a call belonging to compute level i is initiated in cell c , it is admitted into the MCE, and its associated compute task is put on the CS associated with the macro-cell to which c belongs, provided that the current queue length at compute level i of this CS is smaller than M_i . Otherwise, the call is rejected.

3. THE HARMONY ARCHITECTURE

HARMONY is a multilayered architecture as shown in Figure 2 in which each layer and plane is associated with a specific task. In the QoS prediction layer, the *Application Client* provides the interface with the application and gets Q_n and Q_c from the application and forward it to the *Network Client*. The *Network Client* implements the *Entropy Model*. The compute capacity is forwarded to the *Compute Client* in the *Compute QoS Plane*.

In the *Harmony* layer, the *Network Resource Broker* implements the call admission control for network resources. The network resource policing is done by the *Network resource controller*. The *Compute Resource Broker* implements the call admission policies for compute resources. The *QoS-aware* load balancing is implemented in the *Compute Resource Controller*. The *Compute Resource Controller* also maintains a database of free compute capacity available per CS in the MCE.

The *Harmony* layer supports a QoS-aware load-balancing mechanism to maximize resource utilization. The *Compute Mobility Manager (CMM)* maps/relocate computations across CSs in a way transparent to the application. Since the domain of application is known in advance, we assume that there exists appropriate application libraries in all CSs. Multimedia tasks executed in the CS are iterative in nature and works on stream data types. Hence task migration from one CS to another amounts to executing the same application from the library (of the associated CS) and binding the

input data to it along with appropriate state information. In addition, the *Network Mobility Manager (NMM)* in the *Harmony* layer is used to manage handoffs.

3.1. QoS-aware Load-balancing

The QoS-aware load-balancing implemented in the *Harmony* layer redistributes tasks in the queues corresponding to the different compute levels across all CSs so that all CSs are uniformly loaded. Let $\alpha_{l_i}, \alpha_{h_i}$ be the *Load Scaling Factors* corresponding to compute level i , where $0 \leq \alpha_{l_i} \leq \alpha_{h_i} \leq 1$. We define the *High Water Mark (HWM_i)* and the *Low Water Mark (LWM_i)* for every compute level i as $\lceil \alpha_{h_i} \times M_i \rceil$ and $\lceil \alpha_{l_i} \times M_i \rceil$, respectively. In addition, we define the *System Average SA_i* for compute level i by $SA_i = \lfloor \frac{HWM_i + LWM_i}{2} \rfloor$. The *Processor Load (PL)* for compute level i of a CS is defined as High (H) if $l_i > HWM_i$, Low (L) if $l_i < LWM_i$, and Medium (M) if $LWM_i \leq l_i \leq HWM_i$. The processor state at time instant t is determined based on the processor loads at times t and $t - 1$ as follows. The processor is in state High Transmitter (T+) if $PL(t - 1) = H$ and $PL(t) = H$, Transmitter (T) if $PL(t - 1) = M$ or L and $PL(t) = H$, High Receiver (R+) if $PL(t - 1) = L$ and $PL(t) = L$, and Receiver (R) if $PL(t - 1) = H$ or M and $PL(t) = L$. The QoS-aware load-balancing scheme helps maintain equal loads on all CSs by migrating tasks from processors in state T+ and T to processors in state R+ and R. Tasks from CSs in state T+ are migrated to CSs in state R+ until either one of them reaches a queue length of SA_i . If any of the CSs in state T+ remain, tasks from such CSs are migrated to CSs in state R. On the other hand, if CSs in state R remain, we migrate tasks from CSs in state T.

4. DESIGN SPACE EXPLORATION

The *Harmony* layer of the HARMONY architecture that implements the various schemes for network and compute reservation and load-balancing is implemented as a C code. An MU is admitted into the MCE only when the requested Q_n and Q_c can be guaranteed. Load-balancing is invoked on a call admission, call termination and handoff. We simulate an MCE with 64 cells, with a cell diameter of one kilometer and 16 CSs, each CS with a compute capacity of 4096 MIPS. The *Call Durations* are independent and uniformly distributed in the range of 0 to 5 minutes. The *Speed* and *Direction* of an MU is uniformly distributed from 0 to 20 Kmph and 0 to 360 degrees respectively. These assumptions are based on the design space of the *Ubiquitous Communications* project [1]. Simulations were carried out for more than 10000 calls per cell. The design space of HARMONY comprises compute levels N , primary and secondary resource usage factors ρ_p, ρ_s , and call arrival rate λ . The design space is explored by assigning different probabilities to

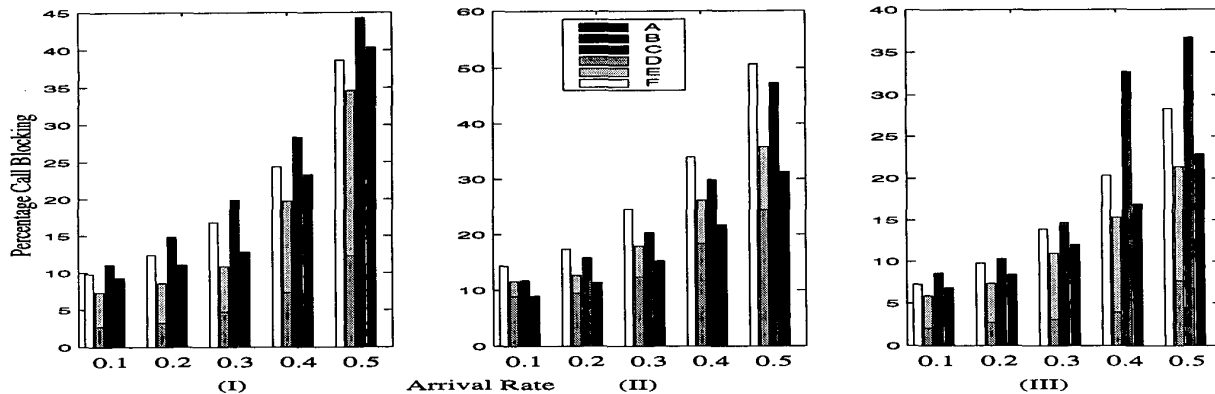


Figure 3: Call Blocking Probability for (I): $P_{ind} = P_{dep} = 0.5, P_{gua} = P_{pred} = 0.5$, (II): $P_{ind} = P_{dep} = 0.5, P_{gua} = 0.7, P_{pred} = 0.3$, (III): $P_{ind} = P_{dep} = 0.5, P_{gua} = 0.3, P_{pred} = 0.7$, The shaded bars are labeled as A : contribution to PCB due to lack of network resources for $\rho_p = \rho_s = 0.8$, B : contribution to PCB due to lack of compute resources, C : PCB for $\rho_p = \rho_s = 0.8$, D : contribution to PCB due to lack of network resources for $\rho_p = \rho_s = 0.6$, E : contribution to PCB due to lack of compute resources, and F : PCB for $\rho_p = \rho_s = 0.6$.

the various traffic classes. The metrics used to explore the design space are *Percentage Call Blocking (PCB)* and *Network Utilization (NU)*. Let P_{dep} and P_{ind} denote the probability of mobility independent and dependent network QoS traffic class and P_{gua} and P_{pred} be the probability of guaranteed and predictive traffic class. We restrict the number of compute levels to six per CS since a further increase did not improve performance. Figure 3 summarizes the results for PCB. Percentage of calls dropped due to lack of network resources reduces to 3.87% from 19.11% as ρ_p and ρ_s are increased from 0 to 0.6. The effect of load-balancing manifests as an improvement in PCB. For $\lambda = 0.1$, load-balancing reduces the PCB to 6.48% from 11.81%. The effect of load-balancing is more pronounced for higher values of λ . Increasing ρ_p and ρ_s results in more resources being reserved in FONs and SONs, and hence less number of calls are dropped due to lack of network resources, but would result in higher under utilization of network resources. We observe that as ρ_p and ρ_s increase from 0.6 to 0.8, PCB increases by 3.06% and NU reduces by 7.8%. The PCB increases by 3.43% and reduces by 4.31% when guaranteed and predictive class dominate respectively as compared to the case when both are equiprobable. The corresponding NU reduces by 17% and increases by 5.1%. When calls subscribing to predictive class dominate, network resources reserved by any MU is available to other MUs subscribing to predictive network QoS class. Similar plots for NU (not shown here for lack of space) combined with those of PCB (Figure 3) defines the feasible region for call admission.

5. CONCLUSIONS

This paper investigates the design space of the HARMONY architecture for providing network and compute guarantees.

Network guarantees are provided based on the ENTROPY model and compute guarantees are provided by mapping and reserving compute capacity for mobile originated tasks to compute servers in the backbone. The effect of load-balancing is seen as an improvement in call blocking probability. Network resources is a bottle neck when MUs subscribing to guaranteed or mobility independent class of traffic dominate and compute resources is a bottleneck when predictive and mobility dependent class of traffic dominate. The increase in primary and secondary resource usage factor results in a higher call blocking probability but reduces the number of calls blocked due to handoff.

6. REFERENCES

- [1] Ubiquitous Communications Project, "Delft University of Technology, The Netherlands", <http://www.ubicom.tudelft.nl>
- [2] A.M. Lele and S.K. Nandy, "Harmony - A Framework for Providing Quality of Service in Wireless Mobile Computing Environment" *Proceedings of the 4th International Conference on High Performance Computing*, Calcutta, India, December-1999.
- [3] K. Nahrstedt, "To Overprovision or To Share via QoS-aware Resource Management", *Proc. of IEEE International Symposium on High Performance Distributed Computing*, 1999, pp. 205-212
- [4] M. Othman and S. Heiles "Power Conservation Strategy for Mobile Computers Using Load Sharing" *Mobile Computing and Communications Review*, January 1998, pp. 203-216.
- [5] C. Aurrecochea, A.T. Campbell and Linda Hauw, "Survey of QoS Architectures", *Tech. Report, Center for Telecommunication Research, Columbia University*, (<http://comet.columbia.edu/publications/conference.html>)
- [6] ATM Forum, <http://www.atmforum.org>
- [7] A.M. Lele and S.K. Nandy, "Can QoS Guarantees be Supported for Live Video Over ATM Networks", *Proc. IEEE Conference on Global Communications*, Nov. 1998.