OVERVIEW ABOUT THE RESOURCE SCHEDULING IN THE WEB OPERATING SYSTEM (WOS™)

HERWIG UNGER
FB Informatik, Universität Rostock, D-18051 Rostock, Germany
Phone: ++49 381-4983403, Fax: ++49 381-4983366,
Email: hunger@informatik.uni-rostock.de

PETER KROPF
Université Laval, Dépt d'Informatique, Pav. Adrien-Pouliot; Québec, PQ, G1K 7P4, Canada
Phone: +1 418 656 35 86, Fax: +1 418 656 23 24,
Email: kropf@ift.ulaval.ca

Abstract

A number of load sharing/balancing mechanisms were developed in order to improve the execution of a distributed application in a workstation cluster or any other kind of distributed architectures. In fact, it would be possible to execute applications on any machine of a world wide system. However, the notion of 'distributed systems' has gained in importance and attraction with the introduction of the Web. This could result in a significant performance improvement for the user. This paper describes the necessary, newly developed, principle concepts for load sharing in a world wide distributed computing environment as described in [10] *to assure efficient automatic sharing of tasks.*

Keywords: Resource Scheduling, Load Balancing, Distributed Systems, Web

1. CONSIDERATION OF LARGE DISTRIBUTED SYSTEMS

It is already proven that workstation clusters are powerful instruments to allow the execution of processor intensive applications [8]. These systems have a good price/performance relation and can avoid high costs for the installation of parallel supercomputers. Modern Load Sharing and Load Balancing systems (for an overview see [3] or [9]) can enhance this relation once more.
The Internet seems to be a very similair, only extremly large distributed architecture - which shall be opened to a widely use by an Web Operating System [10][13]. Because of the size there is the same need to apply powerful resource trading mechanisms to use the full capacity of this system. Considering the circumstandes in workstation cluster systems and problems discussed in related works it become clear that resource scheduling and security of resource accesses might be the key problems for an effective work of a WOS systems [1][4][5][12][13].

But there is a major difference between a common distributed system like a workstation cluster and a Web oriented system: it is the distance between the machines and therefore the low speed

of communication. While even a FastEthernet at 100Mbps may limit the possible performance in a system, large distances between two machines and therefore reduced communication speeds make it impossible to collect and to compute data in real time as it is done in classical load sharing systems.
If there are no hard real-time restrictions, machines far away from the users' computer can be an acceptable place for processing any service for him [10]. Nebertheless, there are a number of new problems (see also Fig. 1) becoming more important, which could be neglected in a workstation cluster system. In particular, we identify the following ones:

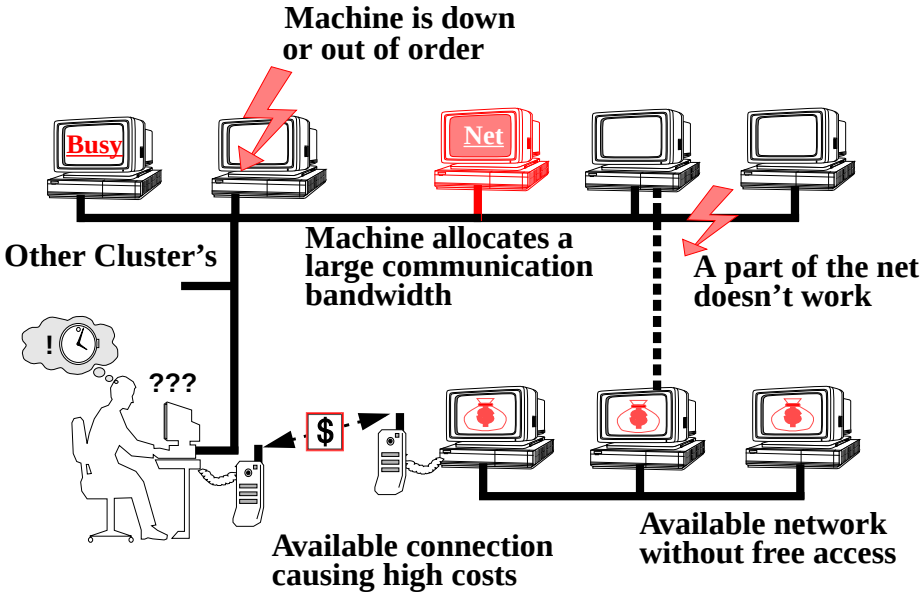

Figure 1: Exemplary Architecture showing some new Problems for a Resource Scheduling in a WOS

- The structure of the system, especially of distant components can not be considered to be a fixed. Furthermore, it is truly heterogeneous in all aspects: structure of processing nodes, messages, used communication units, storage capacities and speeds, etc.
- Because of large communication times, global on-line resource prediction is almost impossible. There is no global manager which can get and hold exact information about the current state of the whole system.
  Furthermore, the generated overhead would further increase the load of the already slow network.
- Another general problem with a central data collection in the system is caused also by the transmission times, since the state of the system may already have changed during the transmission. The transmitted data will thus already be outdated upon arrival and therefore be invalid for any operations at the destination.
- Off-line learning and adaptation methods cannot be used because too many parameters had to be collected. This could cause again a large overhead.
- In large networks with complicated structures failures or breakdowns of the system (the network as well as the computing nodes) are not improbably. Specifically, failures at any intersection also result in a system fault. Therefore, suitable reactions of the system in case of a failure must be implemented as a part of the load sharing system [10].
- Although most applications may be executed on any node without time limits, there might exist restrictions regarding the completion time of an application

- To meet the requirements of the user mobile or private hardware (communication and processing resources) might be used, for which the user has to pay for. Considering this fact, an optimization problem must be solved to get a suitable time/costs ratio.

From the above considerations it is clear that *new* approaches have to be implemented in order to provide each application the most suitable hardware for its execution. As already mentioned in [10], hereby criterias for the optimal execution may differ for each task.

A number of modern, innovative systems like client-server architectures, CORBA [17], 'task brokers' in the HP9000-System [18] or some warehouses concepts [10] already support such a distributed execution of services on different hosts of a distributed architecture. Nevertheless, all these systems need any kind of a *centralized* service (the server or the broker, see Fig. 1) to organize the work and to assign service requests of any user to a suitable host. Therefore the access and management of the server modules may cause larger response and waiting times even in larger systems. Furthermore, special hosts must be distinguish as server nodes.

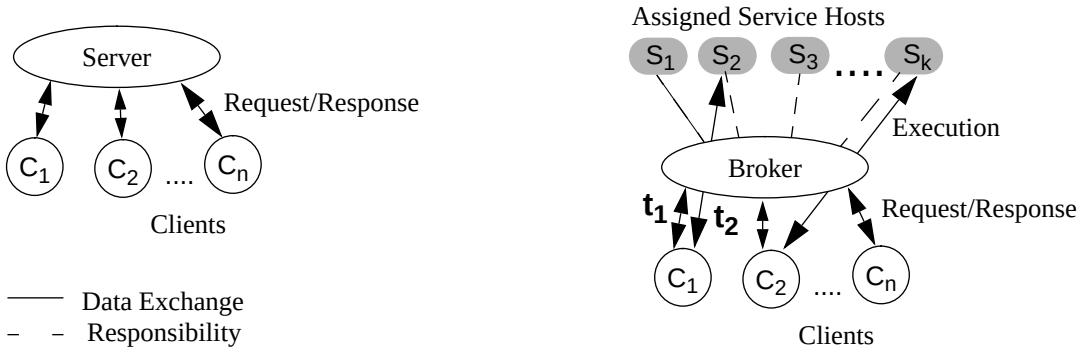

Figure 2: Execution with Clients and Server or Broker

Since no centralized information about the state of a machine can be kept anywhere in the huge world wide web system, each server must contact a potential service host directly before it can start a job there. If more than one server offer the same service, an interconnection between them might be necessary. So the efficiency of a centralized server in a WOS must be doubt once more.
This contribution intend to introduce a more effective, completely decentralized approach for doing so. It mentioned in this section that also the required software / services must be considered in the same manner as all the hardware resources. The reason is that on one hand a correctly working software can only be expected on the machine where it is installed properly, on the other hand the availability of software is also limited by a number of licenses. Therefore, allocating resources also means in the following part to check and to allocate all necessary software resources for a service.

2. MAIN PRINCIPLES

With respect to the requirements mentioned above a suitable load management shall be developed. The heterogeneous nature of the Web, its diversity and dynamics imply that existing systems, system routines and mechanisms have to be used as much as possible. Instead of developing completely new, but standard operating system procedures, we thus rather provide mechanisms for adequate configuration of system and application software and user requests.

An essential precondition for the work of the WOS is that a number of users is registered in the WOSNet (see [2]) and that the administratorsd of the machine have allowed the remote use of hard- and software for other WOS participants. Sharing resources means in our context to define and to declare for public access:

- the used hardware platform and the operating system
- the resources, which will be available for public access in a formalized way
- the programs, which will be shared with other users at defined times of the day or the week with a short service description in a standard form
- a possible, required payment for the use

Now the execution of a (remote) WOS service requires that

- the necessary software is installed in the right version on a given machine
- the administrator of the machine allows a remote user to execute the necessary program(s)
- that machine has been detected to be able to execute this service for the remote user
- actually, enough free resources can be found and allocated on that machine to execute the job
- the job can be finished in the required time / with the required parameters

For the development of suitable management units the consideration of the necessary overhead plays an important role, because network connections cannot be considered to be fast. Therefore searches over the WOSNet and resource trading actions shall be limited to a minimum.

Because of the above said and the knowledge of the user habits while working on a workstation within a local cluster it sems to be convinient to allocate for each user a so called standard user space. This user space should contains a set of different resources on different machines allowing the user to satisfy 80% of his service requests.
Therefore at first a characteristic of typically used processes and programs of the respective users must be created, stored and considered [11][16]. The use of a small, fixed set of user classes as suggested in [16] is not proper for doing so, because of the resource needs of the users may differ to much. For doing so we suggest the application of statistic methods to characterize the processes typically started by the respective user, including all their subprocesses. This also include to keep the host information on which hosts the service was preferably executed. On the base of these informations

- a user standard resource space can be allocated when the user login
- a steady adoptation of the size of the user space can be supported and
- a placement of any application within the reserved space can be done by using standard, well-known load balancing methods (e.g. from [6], [7] or [14]) with a centralized decision unit on the user‘s machine.

Furthermore, each process returns information about the allocated resources of a finished process to the user‘s machine to support an update the named statistics about the average maximal resource needs of the processes and of the user, respectively.
From the above said it is clear that a special resource trading demon must run on each machine, because only the respective machine has the authority to deal with its resources. Subcentralized managers/warehouses cannot fullfil this task because of long data transmission times, they are only able to spread informations where resources could be allocated (see Fig. 3).

The developed specification and implementation shall be discussed in the following chapter.

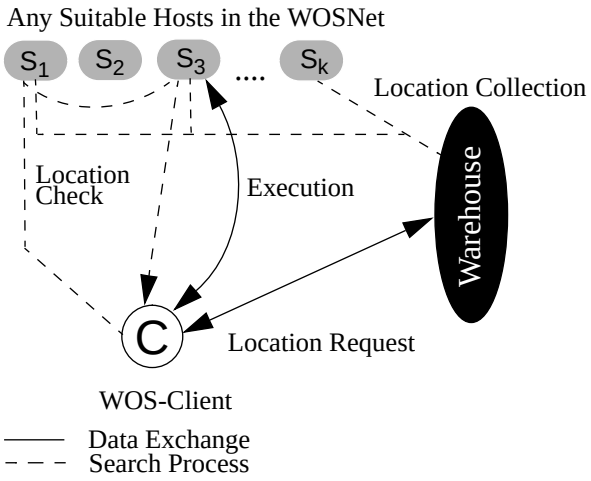

Figure 3: The Warehouseconcept within the WOSNet

3. IMPLEMENTATION

Section 2 figures out that each computer in the WOSNet has two tasks: to be a WOS-client which requests the service execution for the user sitting infront of the machine and to be a WOS-server which accept or reject remote service calls. Therefore we suggest for the effective work of the system including the resource scheduling the structure shown in Fig. 4.

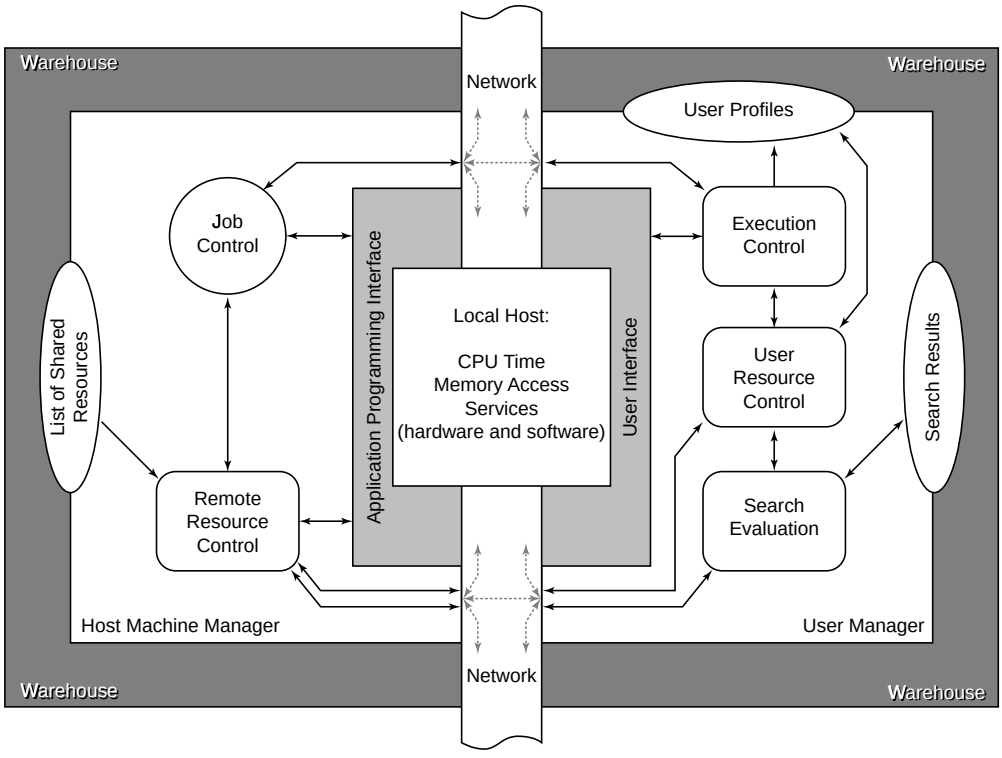

Figure 4: Structure of a Node in the WOSNet

The left side of this figure shows the server the right one the client part of each WOS node. The functionality is the following. Each user command is processed by the client side execution control unit. It decides, whether a process must be loaclly executed (for instance a *ps*-command) or not. While local executable jobs will directly proceed to the user interface of the respective operating system, all other service requests are sent to the user resource control unit. Here a decision will be made, if the requested service can be fulfilled in the prereserved standard user space. If it is so, with a common load sharing mechanism will be determined on which machine this will be done.
Otherwise a search over the WOSNet must be started as described more detailed in [15]. This is the task of the Search Evaluation unit which also evaluate the results of that request. For that search former search results from the local warehouse can be give an satisfactor answer or a WOSNet wide search must be organized. The results will be used from the user resource control unit to contact potential hosts. A succesful service execution require, that at least one remote resource control unit of a machine must obtain from its local warehouse a valid list entry that the service can be executed on that machine for the requesting user and must be able to allocate the necessary resources for that job. Than the job can be started there under the responsibility of the remote job control unit which also contacts the user execution control to transmit the results. Therefore we suggest the use of a Net File System, like it will be used in WebOS [5].
Nevertheless, all search and execution information will be used to update the respective user profile and search warehouse entries.

4. CONCLUSION

We have present an overview about the WOS-Project Resource Scheduling approach. Hereby we figured out that partially known load balancing methods can be used within an allocated standrad resource space and described a mechanism for the resource trading within the WOSNet.
After the implementation of the WOSP and WOSRP protocols the implementation is going on in collaboration with the development of the saerch and resource trading methods. Most difficulties arise from the fact that a partial set of files must be available for the execution of some services and all data transmission might need to much time or generate a too high overhead within the WOSNet as well as security aspects while accessing remote file systems.

REFERENCES

[1] A.D. Alexandrov, M. Ibel, K.E. Schauser, C.J. Scheimann, SuperWeb: Research Issues in Java-Based Global Computing, *Workshop on Java for Computational Science and Engineering Workshop, Syracuse University,* 1996.

[2] G. Babin, Requirements for the Implementation of WOS-Protocols, *Proceedings of the Workshop for Distributed Computing on the Web, Rostock,* 1998

[3] M.A. Baker, G.C. Fox, H.W. Yau, A Review of Commercial and Research Cluster Management Software, *Technical Report, Syracuse University New York,* 1996

[4] J.E. Baldeschwieler, R.D. Blumofe, E.A. Brewer, ATLAS: An Infrastructure for Global Computing, *7th ACM SIGOPS European Workshop on System Support for Worldwide Applications,* 1996

[5] D. Culler et al, WebOS: Operating Services for Wide Area Applications, *Technical Report, http://now.cs.berekeley.edu/WebOS,* 1997

[6] W. Becker, Dynamische, adaptive Lastbalancierung für große heterogen konkurrierende Anwendungen, *PhD-Thesis, University of Stuttgart*, 1995

[7] K.P. Bubendorfer, Resource Based Policies for Load Distribution, *PhD-Thesis, University of Wellington,* 1996

[8] G. Hipper, D. Tavangarian, A new Architecture for Efficient Parallel Computing in Workstation Clusters, *Int. Conference HPCN Challenges in Telecomp and Telecom: Parallel Simulation of Complex Systems and Large-Scale Applications, Delft, Netherlands*, 1996

[9] J.A. Kaplan, M.L. Nelson, A Comparision of Queueing, Cluster and Distributed Computing Systems, *NASA Technical Memorandum 109025, Langley Research Center*, 1993

[10] S.B. Lamine, J. Plaice, P. Kropf, Problems of Computing in the WEB, *Proc. of the Int. Conference on HPC'97, Atlanta*, 1997

[11] P. Mehra, Automated Learning of Load Balancing Strategies for a Distributed Computer System, *Technical Report, University of Illinois*, 1993

[12] A. Reinefeld u.a.: The MOL-Project: An Open Extensible Metacomputer, P*roc. of the IPPS'97*, http://www.uni-paderborn.de/fachbereich/AG/monien/PUBLICATIONS/list_publica_1997.html

[13] F . Reynolds: *Evolving an operating system for the Web.-* IEEE Computer, 29(9). 1996. pp. 90--92, 1996.

[14] H. Unger, T. Böhme, A Decentral, Adaptive Load Sharing Approach for workstation Clusters using Fuzzy Logic, *Proceedings of the Workshop „Anwendungsbezogene Lastbalancierung ALV'98"*, München, 1998

[15] H. Unger, T. Böhme, Search in the Web - a User Adaptive Approach, *Proceedings of the Workshop for Distributed Computing on the Web, Rostock,* 1998

[16] G. Wilhelms, Dynamische adaptive Lastverteilung für PVM mittels unscharfer Benutzerprofile, *PhD-Thesis, Univ. Augsburg*, 1994

[17] n.n., The CORBA Architekture and Specification, *Technical Report, http://www.omg.org/corba/corbiiop.htm,* 1997

[18] n.n., HP Task Broker for HP9000 Servers and Workstations, *http://hpcc997.external.hp.com/wsg/ssa/task.html#task*, 1995