Gilbert Babin. "Requirements for the Implementation of WOSTM Protocols." in Distributed Computing on the Web Workshop (DCW'98). Rostock, Germany. June 1998. pp. 129-133.

REQUIREMENTS FOR THE IMPLEMENTATION OF WOS™ PROTOCOLS

GILBERT BABIN
Département d'informatique, Université Laval, Québec, Canada G1K 7P4,
babin@ift.ulaval.ca

Keywords: Distributed Systems, Protocols, World Wide Web, WOS™.

1. INTRODUCTION

To reap the potential of the Web, mechanisms are required to locate and use available, suitable resources. In [1], a two-level protocol was introduced for that purpose. A first protocol allows the selection of an appropriate version of WOS resources, the WOS Request Protocol (WOSRP). Another protocol, the WOS Protocol (WOSP), allows to locate and use distributed resources, such as services, machines, etc., in a fault-tolerant manner over the Web.

This extended abstract presents some of the basic requirements for the implementation of these protocols, using TCP/UDP as transportation means.

2. OVERVIEW OF WOS PROTOCOLS

This section briefly presents previous results about WOS Protocols.

2.1. WOS Request Protocol

Figures 1 and 2 illustrate how a WOS client (a WOS node in its client mode) may use WOSRP to obtain information about other WOS servers (a WOS node in its server mode). At first, a WOS client will broadcast a request to all machines in its immediate vicinity (the local network where the machine is located). Any WOS server being able to provide a positive answer will respond. In this case, more detailed requests may be submitted to those WOS servers. Otherwise, the WOS client broadcasts a request to all the machines at the next network level, and so on (see Fig. 1). Note that the network levels correspond to physical network levels : local network, network of local networks, network of networks of local networks, etc. All the responses received are used to populate the WOS client warehouse, which is at first empty when the WOS node enters WOSNet (the network of all WOS nodes) (see Fig. 2). Registration to WOSNet

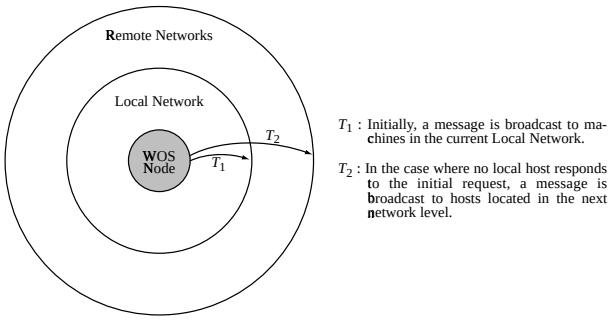

Figure 1: WOSRP Information Retrieval Strategy

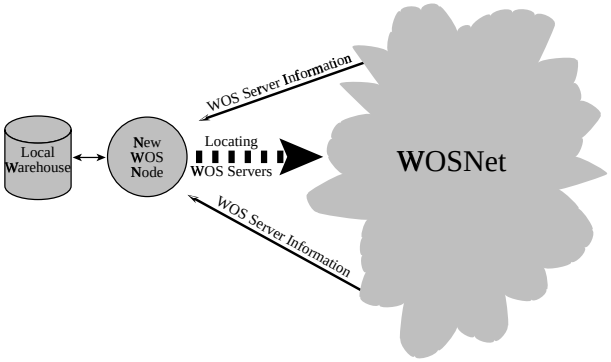

Figure 2: WOS Server Initialization Mechanism

is implicit. The only requirement necessary is that the WOS subsystem has been previously installed (daemon, plug-in, etc.).

WOSRP uses a “pull” philosophy, where a WOS node requests information from other nodes in its vicinity. These messages may be lost without any disruption of service. Furthermore, WOS nodes may decide to propagate these messages to other sites. Eventually, replies may be returned to the node which made the original request.

2.2. WOS Protocol

WOSP handles the following interactions between WOS nodes :

1. Setup commands to change the execution parameters of a WOS node.
2. Execution commands allowing a WOS client to use resources from another node.
3. Query commands used by a WOS client to interrogate another WOS node’s warehouse.

Communications are connectionless. A node sends a request to another node. That node processes the commands and sends the results back to the requesting node.

2.3. Fault tolerance

In [3], we have studied approaches to efficiently locate resources in WOSNet. The approach retained searches for resources on WOS servers using multiple sequential search chains. For the search to be effective, each machine in every chain must be reached, until either all the WOS servers in each chain have been visited or a WOS server able to supply the resources required has been found.

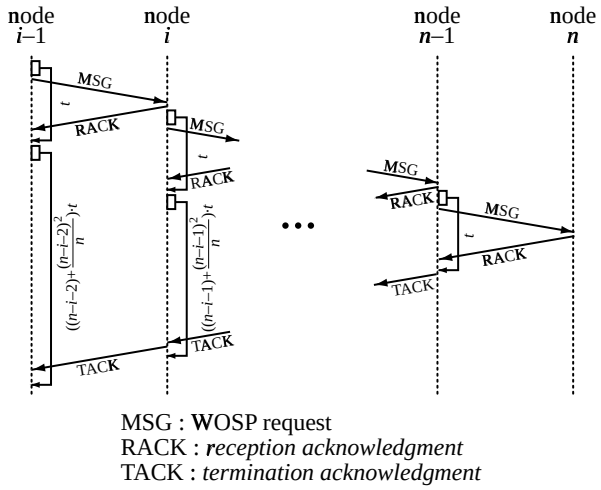

Figure 3: Locating Resources in WOSNet

To overcome potential communication breakdowns, Babin et al. [1] suggest the use of two types of acknowledgment messages. The first acknowledgment, the *reception acknowledgment* (RACK), confirms proper reception of the message by the next machine in the chain. The second acknowledgment, the *termination acknowledgment* (TACK), indicates that either all the WOS servers in the chain were visited or a suitable candidate was found, as shown in Figure 3.

3. REQUIREMENTS FOR WOSRP AND WOSP IMPLEMENTATIONS

Although the general use of WOSRP and WOSP is fairly well specified, there remain a number of tasks to be accomplished before completing their implementation. Some of these tasks necessitate further research. Here is a list of these tasks :

- *Development of a WOSP version naming scheme*. Presently, WOSRP specifies the maximum length of a version name. However, a suitable naming scheme (naming space, naming conventions, name assignment, distributed name management, etc.) still needs to be specified. In [1], a name management approach similar to DNS is suggested. In order to support a larger number of version names, a mapping between version names and version identifier (similar to the hostname and IP address mapping) will be required.

This way, WOSRP may be used for up to 2^{3840} ($\approx 10^{1156}$) distinct WOSP versions, which is more than enough.

- *Definition of a resource version management approach*. In addition to WOSP versions, other resources in WOSNet are also versioned. Version space management tools must also be developed, based on the work of Plaice [2].

- *Implementation of a warehouse management module*. Warehouse information must be structured. An initial data model has to be specified. This model should allow for different types of resources known and managed by the WOS node to be represented, each resource potentially having different characteristics. Furthermore, versions of these resources must be managed. This include WOSP versions. Interfaces to the warehouse have to be defined to insert, update and delete information about known resources. In addition to warehouse management tools, information retrieval interfaces must be specified. These interfaces must be flexible and easily maintainable.

These modules are required for a WOS node to fully accomplish all its tasks properly. However, simpler versions may be developed in order to test the communication mechanisms between WOS nodes. The development of WOSRP may be achieved by performing the following tasks :

- *Implementing the information retrieval strategy as described in [1] (see Fig. 1).*

- *Establishing of a WOSP connection with a requesting node.*

In the case of WOSP, tasks remaining include :

- *Implementing the resource localization strategy as described in [3] and [1].*

- *Specifying and implementing response messages to WOSP queries.*

The previous tasks assure that peer-to-peer communications are properly handled at the WOSRP and WOSP level. These communications use the services provided by lower level protocol layers (TCP or UDP, IP, etc.). However, this is not sufficient. The WOSRP and WOSP are developped to be used by WOS managers/modules (see Fig. 4). Therefore, the services required by these modules will guide the development of suitable service access points.

ACKNOWLEDGMENTS

This research was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC grant #OGP0155899).

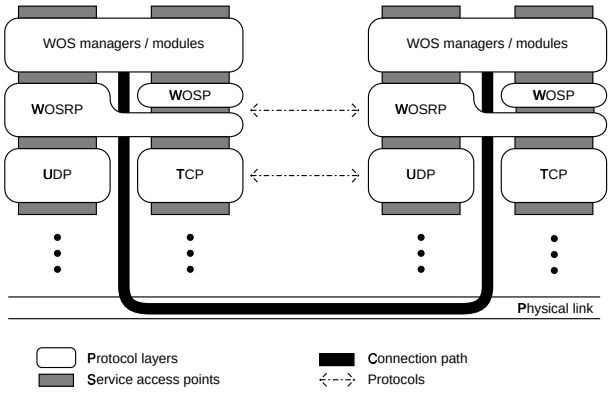

Figure 4: Interfaces between Protocol Layers

REFERENCES

[1] Gilbert Babin, Peter Kropf, and Herwig Unger.
"*A two-level communication protocol for a web operating system (WOS^{TM}).*" -
Euromicro Workshop on Network Computing, Västerås, Sweden, August (1998). To appear.

[2] J. Plaice and W.W. Wadge.
"*A new approach to version control*." -
in: IEEE Transactions of Software Engineering, S. 268–276, 19[3], (1993).

[3] Herwig Unger, Peter Kropf, Gilbert Babin, and Thomas Böhme.
"*Simulation of search and distribution methods for jobs in a Web operating system (WOS^{TM}).*" -
1998 Advanced Simulation Technologies Conference (ASTC 1998), Boston, Massachusetts, USA, April (1998).

Gilbert Babin
received the B.Sc and M.Sc. from Université de Montréal (Canada), and the Ph.D. from Rensselaer Polytechnic Institute. Since graduating in 1993, he has been a faculty member at Université Laval (Canada). He has been working on the integration of heterogeneous, distributed databases using reactive agents and a central knowledge repository. In addition, his current interests include the formalization of the requirement engineering process, the Web Operating System (WOS^{TM}), and activity report generation from activity databases.