ISTITUTO NAZIONALE DI FISICA NUCLEARE

Sezione di Bari

INFN/TC–98/21
10 Settembre 1998

M. Castellano, G. Tomasicchio, A. Ventrella:

THE AFS PROJECT AT INFN: DEVELOPMENT OF A HIGH LEVEL ADMINISTRATION LAYER

PACS: 07.05.Ys, 07.05.Bx, 07.05.-t
Keywords: AFS. ARC, distributed file system, Kerberos, groupware

Presented in part as Thesis in Electronic Engineering at the Polytechnic of Bari
Thesis, 18 February (1998), Bari, Italy

Published by ***SIS–Pubblicazioni***
Laboratori Nazionali di Frascati

INFN – Istituto Nazionale di Fisica Nucleare
Sezione di Bari

INFN/TC–98/21
10 Settembre 1998

THE AFS PROJECT AT INFN: DEVELOPMENT OF A HIGH LEVEL ADMINISTRATION LAYER

M. Castellano[1)], G. Tomasicchio, A. Ventrella
INFN–Sezione di Bari, Via Amendola 173, I–70126 Bari, Italy

Abstract

The research activities of the Istituto Nazionale di Fisica Nucleare in the field high energy physics, based upon world-wide collaborations, is increasingly requiring a work organization based on the teamwork. As a basis to support research groupware activities, INFN Computing Committee decided to apply a working group on the Andrew File System, a network distributed file system architecture, as a solution which provides a common name space to access file resources spread on a geographical wide area. This paper is a contribute for the collaborative administration activity of the AFS distributed environment. Here a project to allow a distributed administration over the INFN AFS-cell is drawn. The solution is described and largely developed.

[1)] Corresponding Author: Marcello Castellano, E-Mail: Castellano@ba.infn.it

Work supported by INFN–Bari

1 – INTRODUCTION

The Teamwork is a people organization model largely used in the scientific research institutions to provide coordination and cooperation to reach a common scientific goal in efficient way. It requires to define communication channels among entities belonging to the teamwork. Several years ago channels were defined by means of direct contacts among persons, mail messages, electronic store devices and so on. Over the time, large organizations equipped with information technology media were able to define communication channels based on computer networks. The collaborative work, based on the distributed information technology is nowadays well known as groupware [1]. A distributed information system is considered based on the groupware technology when several workstations are connected among them, providing the services to easily communicate and share information among users. The groupware technology becomes relevant especially when large projects are managed in wide collaborations, like in experimental groups, involving several different research institutions.

In High Energy Physics (HEP), several interesting applications of the groupware technology are used. One of them concerns the sharing of project/experiment information resources and data repositories. The Andrew File System (AFS), a distributed file system architecture has been taken into account to allow the transparent access and the organization of network-wide distributed information resources. The AFS file organization creates a uniform vision of information for any scientific project, distributed on computing resources spread on a wide area network (WAN). AFS allows an efficient distribution of large quantity of data over the network, with a reduced traffic on WAN. This is the result of an advanced data replication technique based on the Kerberos security technology, increasing the reliability of the system by means of multiple copies of information, and supplying facilities for data distribution, backup and management of distributed services like "mailing" or "Internet News". Projects based on AFS take advantage by a developer-friendly multi-platform development environment with able to access to the same data and programs on whichever system. AFS is an effective solution for a workgroup on a computer network.

2 – THE ANDREW FILE SYSTEM

AFS is a distributed file system, whose main purpose is to provide a high level visibility of the files spread around different computer platform on LAN and WAN networks. AFS allows the file sharing among distributed heterogeneous machines by means of client/server processes cooperating among them over the RPC protocol in the TCP/IP communication stack as represented in fig.1.

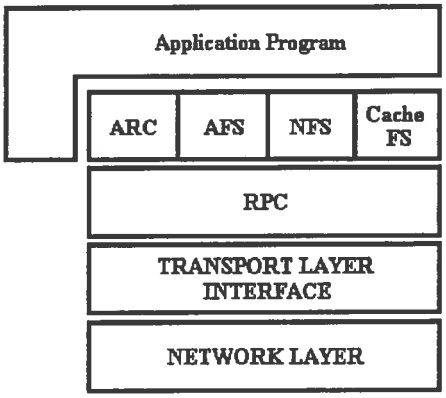

FIG. 1 – TCP/IP stack with AFS service.

2.1– AFS Global Name Space

The AFS information tree structure is shown in fig. 2.

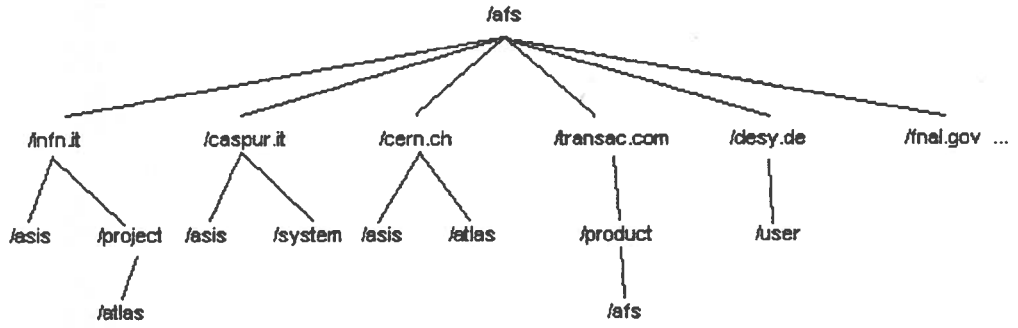

FIG. 2 – AFS common Global Name Space.

The AFS common Global Name Space is realized through a naming structure based on the directory syntax model that assure uniqueness of names in a common global area. Each directory is defined by a declaration of both the 'volume name' and 'mount point' in AFS environment.

The AFS space is organized in cells, which represent the first level of /afs tree. Every cell is an independent administrative domain of a given organization which corresponds to its own DNS-domain. For each AFS cell the login, the Kerberos Authentication, the File View, the ACL protection and Management service are available. The figure 3 represents the AFS Client/Server model which involves AFS clients, running a Cache Manager process for volumes location, AFS file servers and AFS Authentication servers for kerberos-based user authorization [2].

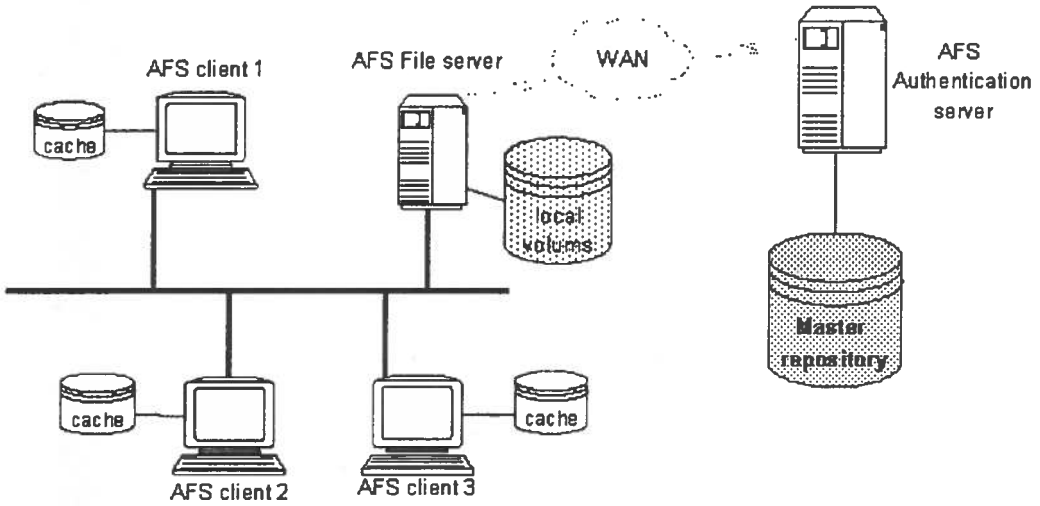

FIG. 3 – AFS Client/Server model

The Cache Manager controls the AFS cache reserved on a client to improve the access to volumes on remote servers. The AFS server machines store software and common data, supply services to the clients in network and keep updated the following databases:

- Volume Location;
- Authentication;
- Protection (ACL);
- Backup.

AFS distinguishes server machines in file and authentication server. The former uses a remote authentication database, while a local database is required in the latter. Table 1 shows the various AFS servers reporting all the processes running for each server type.

TABLE 1 – AFS daemons processes and their functions on various AFS server machines.

Simple File Server

AFS daemon processes	Functions
boserver : BOS (Basic Overseer Server)	Monitors all the other processes running on the machines (simples).
fileserver (File Server)	Manages program requests, data files and directories for the Cache Manager.
volserver (Volume Server)	Manages operations on the volumes, as creation, remove and move.
salvager (Salvager)	Controls consistence and state of the system, logging the errors found; run only if anyone of the processes is unsuccessful.
Update Server (client version)	Requests the right version of files to the Binary Distribution machines.
runntp (client version)	Supplies the synchronization mechanism for server machines.

System Control

AFS daemon processes	Functions
runntp (server version)	AFS interface program to NTP daemon.
Update Server (server version)	Guarantees that all the machines run the same version of server process.

Binary Distribution

AFS daemon processes	Functions
Update Server (server version)	Same as System Control machine.

Authentication Server

AFS daemon processes	Functions
kas (Authentication Server)	Controls the user identity into system at the authentication time.
pts (Protection Server)	Creates a list of all AFS users and creates entries for their directories into ACL and Protection database.
vl (Volume Location Server)	Finds volumes and files location, manages Volume Location Database
bus (Backup Server)	Manages Backup Database.

3 – INFORMATION SHARING USING AFS IN SCIENTIFIC ENVIRONMENT

Relevant applications of information sharing in the HEP-field are both the software distribution and software development for world-wide experimental collaborations. The distribution of the official experimental software at the European Centre of Nuclear Research (CERN), represents a model provided by the Application Software Installation Service (ASIS). This service is supported by the CERN-IT Division. Nowadays, this service is largely based on the AFS solution. The software maintenance occurs through replication of directory trees from a server to another also between different AFS cells. This activity is called 'mirroring' and it is implemented through the "mirdir" command. The command has been developed at Consortium of Supercomputing Applications for the University and research (CASPUR) and includes several options to select some directories or files. For what is concerning the remote development of the software, it is possible to refer a model that does not concern only one production site, since a project software of medium-large size might involve the collaboration of people spreaded geographically in more groups. Each group might be responsible for the development of a subproject component. In this sense, the software development might happen even locally to every group, by using an AFS volume on a local server and then released over a central collaboration AFS-repository of the experiment. Each group belonging to the same scientific collaboration is responsible (qualified in reading and writing) for a set of directories, while the rest of the collaboration accesses in single reading to the software produced by the group. As an example the fig. 4 shows a typical information structure of a HEP experiment:

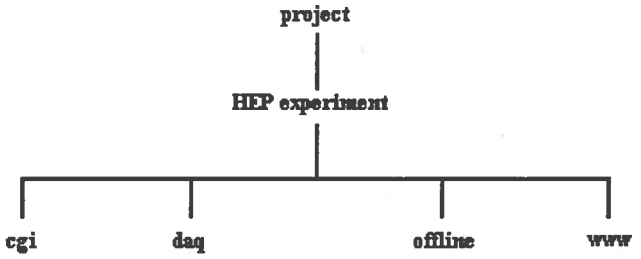

FIG. 4 – Tree structure of HEP experiment.

where:

- **cgi** is the collection directory of all the processing web scripts of request form for scientific information;
- **daq (Data Acquisition)** is the group of all the information (programs and documents) regarding the development project of data acquisition for the HEP experiment;
- **offline** is a collection of all programs and documents of the simulation and offline analysis;
- **www** is the group of documents published on Internet, relative to the HEP experiment.

This distributed model for the software development, offers the following advantages:

- The development happens locally, divided in project parts, and is independent from the state of the network link with CERN;

- Protections, directories structure and the host server are defined in the research institute, without need of interactions with the management of the cern.ch AFS cell.

The alternative to the usage of an AFS mount-point is that can be defined a batch job of mirroring which, in agreement with CERN administrators, regularly copies on the cern.ch cell the new release of project parts of the external collaboration.

The use of AFS in INFN has been evaluated like a real necessity and answer to the modern requirements of distributed collaborations.

3.1 – The AFS cell for INFN

The National Institute of Nuclear Physics has adopted AFS as a distributed file system. At this purpose has been declared a new AFS cell in the world reserved to the INFN institution called infn.it [3]. The tree structure of the infn.it cell is shown in fig. 5.

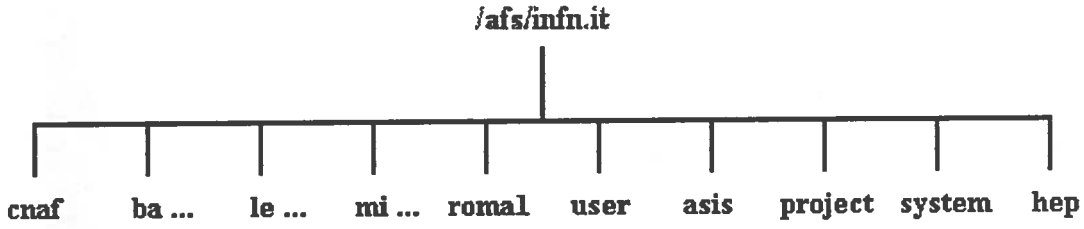

FIG. 5 – INFN AFS tree.

Inside the project subdirectory, are situated the names of the main scientific experiments at INFN, which in turn might be linked or mirrored from other laboratories.

The INFN has defined three volume types as follows:

- **user.<user name>**
- **project.<project or experiment name>**
- **section.<directory name>**

The volume section.<directory name> is in charge of the local file server administrators in the "INFN sections" (i.e. ba, le, mi, ...). A good naming convention which assign it to the path /afs/infn.it/section/<directory name> allows easily to identify the relative volume-mount point in infn.it cell.

In infn.it cell, three authentication servers have been distributed to cover the geographical link map: CNAF (afs1.infn.it), Roma1 (afs2.infn.it) and Naples section (afs3.infn.it), while elsewhere there are section file servers. The evaluation phase of the INFN cell served also to develop utilities and relative know-how to data replication, mirroring and management of the system. As an example, it comes regularly scheduled a mirror of the ASIS/CERNlib collection every thirty/forty days from CASPUR, while the updates of TRANSARC software come carried out by the responsible of every section server. In fig.6 is shown the AFS servers map which describes the whole infn.it cell scenario.

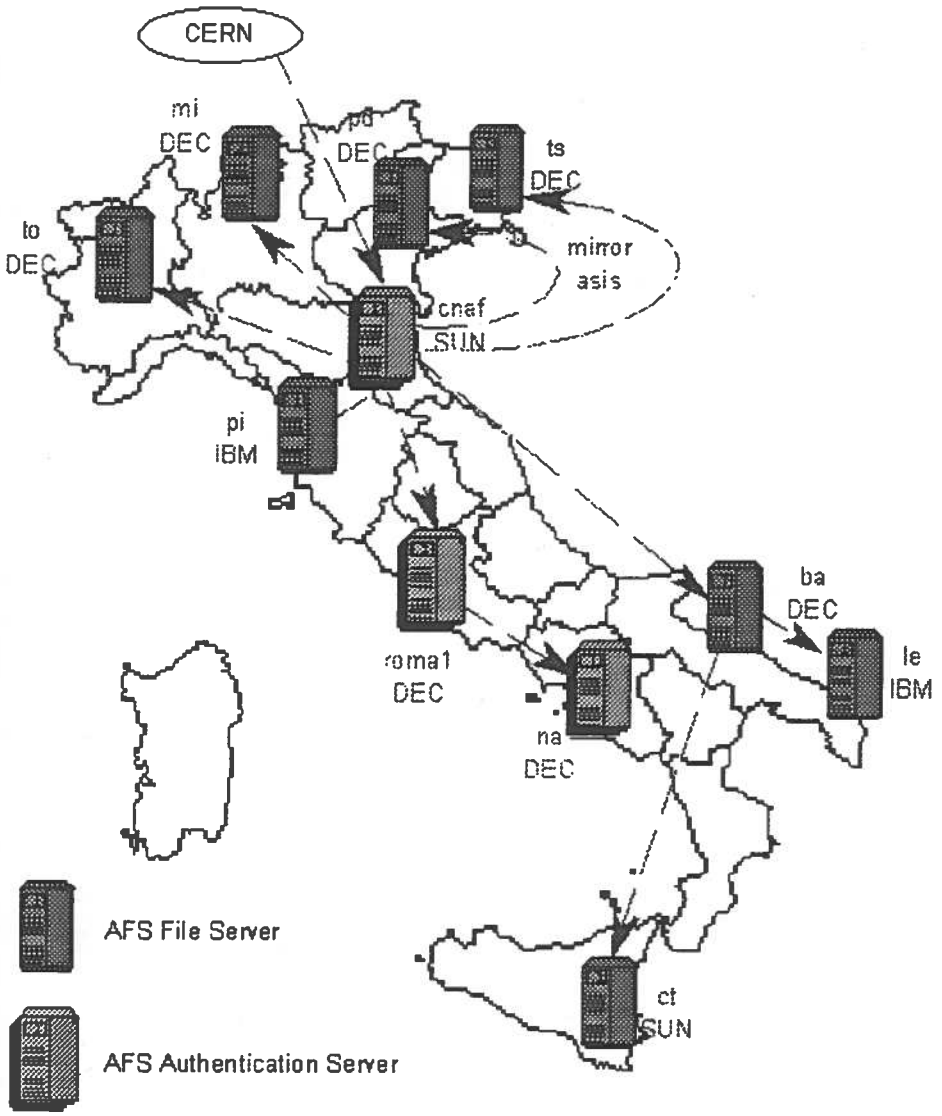

Fig. 6 - AFS servers map into infn.it cell

In order to keep the server processes running on various machines be compatible among them, a time synchronization policy has been required. For this reason, the server version of the "runntp" process synchronizes all the servers of cell, by means of the standard Network Time Protocol (NTP) server. Instead, on the Simple File Server machines run the client version of runntp process, which is synchronized with the analogous processes running on System Control Server machines. In infn.it cell every section server and client synchronizes itself with the reference clock for the infn.it cell which is supplied by Roma1 authentication server (afs2.infn.it) that works as System Control machine and Authentication Server machine. The CNAF server (afs1.infn.it) and Naples server (afs3.infn.it) are synchronized with Roma1. The Roma1 server synchronizes itself with the synchronization source server of Genoa (ntp-ge-1.infn.it) that is not part of the infn.it cell and in turn synchronizes itself with the CERN synchronization server.

The CNAF network centre is the point with better connectivity towards the CERN, therefore on its AFS server is defined a mirroring service of the ASIS software directly from CERN. The CNAF volume containing the ASIS mirroring, can be replicated in other AFS server machines of the infn.it cell and therefore, the access from local machines (AFS clients) to CERN software happens in an efficient way, by using the AFS server topological nearer to every workstation client, so that keeping low the traffic on WAN links.

From indications stressed in the first evaluation phase of AFS, has become necessary to extend the AFS servers structure in INFN and also to define new AFS cells, according to the development of the network connectivity and the specific requirements in every section.

4 – A PROJECT FOR HIGH LEVEL AFS ADMINISTRATION

The AFS resource administration, provided by Transarc, foresees several commands classified in different suite as follows:

- "vos" suite: addsite, backup, backupsys, creating, dump, lock, move, release, remove, remsite, rename, restore, syncserv, syncvldb, unlock, unlockvldb, zap;
- "fs" suite: mkmount, cleanacl, setquota, setvol, setacl, rmmount, copyacl;
- "bos" suite: restart, salvage, shutdown, start, stop.

The "vos" commands allow to contact the «Volume Server» and «Volume Location Server» processes for the AFS volumes management, those of the "fs" suite are used for AFS file system management, and finally, those of the "bos" suite are used for the servers collection management of the whole cell.

The AFS basic management provides a unique centralized administration model therefore only one administrator is authorized to perform all the AFS system functionalities. A major constraint to this model is that giving the administrator access to others managers means to extend all the privileges for the cell management and not only to allow harmless operations on local server resources. In a geographical distributed cell it becomes important to provide a distributed administration model, in such a way several local administrators are authorized of using AFS commands to operate over local resources and to give them local visibility. The authentication databases are controlled by a group of managers which are authorized to insert, remove and modify account and home directory of AFS users. In this sense we talk about "distributed responsibility" since are defined more persons which are responsible to the management of their own users in the authentication database. This distributed management facility has been implemented by the ARC software developed by Rainer Toebbicke at CERN [4]. Through ARC different section server managers are able to operate like privileged users on the server of their competence, but not on all the other cell servers, avoiding the risk that local management errors spread on the whole INFN cell. The manager name list is in a specific ARC authorization data base where is placed the ARC server for infn.it cell. All the privileged commands must be packed up in arc, that is preceded from the call to the ARC routine [6]:

arc -p -h <ARC Server> <afs command>

ARC server checks for the authorization to execute the command, since its activates a communication that uses the Kerberos technology. Notwithstanding the power of ARC software, some additional considerations have been pointed out by the INFN-working group. These considerations were regarding the ARC syntax complexity, based on several flags, and the necessity of a user-friendly high level interface to perform all the AFS administration functions.

4.1 – The system architecture for AFS high level administration

In order to simplify the distributed management of AFS resources, in INFN-Bari a project to manage in a powerful and easy way the AFS administrative functions has been

drawn. The goal of the system, named "AFSadmin", is to provide both a high level graphical user-friendly interface and a collection of scripts for high level commands to allow distributed administration over the INFN AFS-cell. A layered scheme of the system is shown in fig.7.

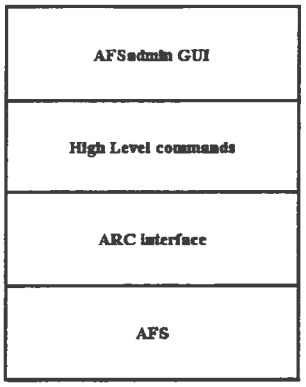

FIG. 7 – High level management stack

4.1.1 – *High level commands*

The higher level commands layered on ARC interface are summarized in the following with the administration script sets [7]:

a) User management scripts:

afsadduser, afsveruser, afsuinfo, afschuinfo, afspwuser, afsslquser, afssquser, afsslltuser, afssltuser, afsrmuser.

b) Volumes management scripts:

afsaddsite, afsbackup, afsbackupsys, afsdump, afslock, afsmove, afsrelease, afsremove, afsremsite, afsrename, afsrestore, afssyncserv, afsunlock, afsunlockvldb, afszap.

c) File system management scripts:

afsmkmount, afscleanacl, afssetquota, afssetvol, afssetacl, afsrmmount, afscopyacl.

d) Server management scripts:

afsrestart, afssalvage, afsshutdown afsstart, afsstop.

As an example of a prototype script to create an AFS user account we have the following sintax:

afsadduser <user> <passwd> <quota> <token> <uinfo>

A modular description of AFSadmin is represented in fig. 8 according to the formalism of the *Structure Chart* model [5].

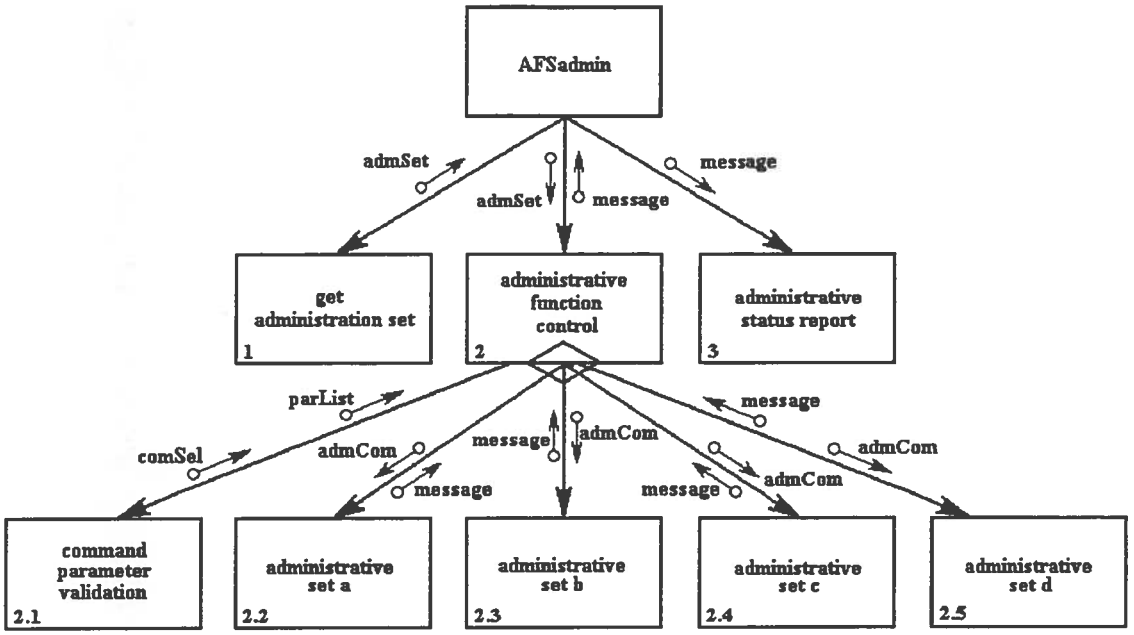

Fig. 8 – Structure chart of the AFSadmin system

In the following we describe the functions of the modules shown in the figure 7:

1 – sends the administration set (admSet) to module 2, in such a way this can be able to select any type of administrative set;

2 – manages all the administrative set modules in mutual exclusive way to execute a given command (admCom);

3 – makes a report of errors or results from executed command (message) from AFSadmin system;

2.1 – sends the command selection (comSel) and the parameters (parList) to module 2, in such a way to process a defined command in a given administrative set module;

2.2 – processes the requests concerning the User management in the cell;

2.3 – processes the requests concerning the volume management in the cell;

2.4 – processes the requests concerning the AFS cell directory tree management;

2.5 – processes the requests concerning the management of AFS server processes and server machines.

The modules 2.2-2.5, regarding a given administration function set, rely on a different high level administration script collection, which allows a local administrator to perform several basic AFS commands.

4.1.2 – *The graphical interface of AFSadmin*

The administration system has been integrated with a user friendly graphical interface developed in Tcl/Tk environment [8]. Tcl/Tk is an interpreted command line language with advanced features like lists, keyed arrays, sets, event handlers, text extraction and substitution primitives. In particular the Tk component allows a complete programming toolkit for designing complex user friendly graphical interfaces (GUI).

The key point in the Tcl/Tk implementation is that it is freely available on all common computer platforms (Unix, VMS, Windows and MacOS). Furtherly, there exist plug-ins and web server extensions able to execute Tcl/Tk code directly inside a web browser. This means that a given web page can allow the execution of the Tcl/Tk program on a remote server from whatever graphic terminal.

The first level graphic form of *AFSadmin*, shown in fig. 9, represents the interface to select the administration function set.

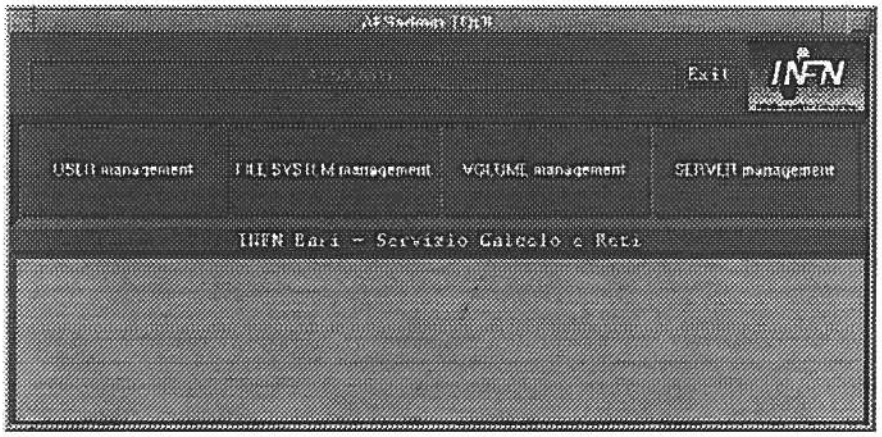

Fig. 9 – First level form

Figure 10 represents a simple request form which allow to provide the parameters list needed for a command of the *User management* administrative set chosen from a *radio-button menu*. The message box in the bottom of the GUI layout allows to show the execution status of the launched commands as well as their relative results. In this layout, when an administrator wish to perform a certain operation, he might push on one of the four administrative set and then select the specific command among all those available in the set.

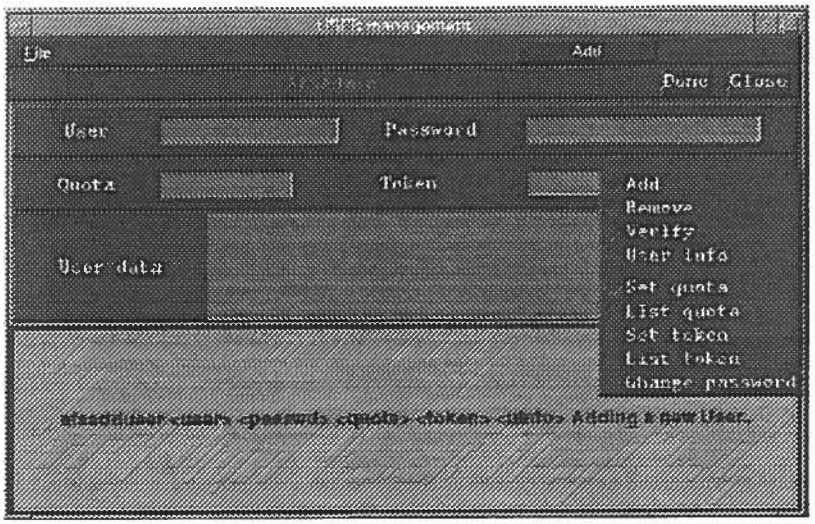

FIG. 10 – *AFSadmin* parameter request form

4 – CONCLUSIONS

In this paper, a project to provide high level facilities for AFS administration, has been presented and largely developed. This system allows to integrate the AFS management functionality, in such a way, to have a single software environment which extremely facilitates the AFS administration. The system interfaces could be improved both to allow any administrator to customize it adding for example any new script for AFS management by using specific configuration files and to develop a version able to run the AFS administration system with restricted access on the web.

5 - ACKNOWLEDGEMENTS

This project has been realized under the auspicious of the convention between INFN and Politecnico of Bari with the aim to provide at the graduate electronic engineering students the relevant aspects of the advanced scientific and technological research retained strategic for INFN activities. We would like to thank the director G. Maggi and the technical support of the "Servizio Calcolo e Reti" of INFN-Bari. Finally, we are grateful to the AFS INFN workgroup for the useful suggestions and discussions.

REFERENCES

(1) F. Filippazzi – G.Occhini "Groupware: Processing to work together" – Franco Angeli.
(2) G. Tomasicchio "L'architettura AFS e il suo impiego nell'INFN", Nota Interna INFN-BA/TC-96/1 [http://www.ba.infn.it/~tomas/afs/afs.html].
(3) AFS HOME PAGE [http://www.roma1.infn.it/AFS/afs_home.html]
(4) ARC program at CERN [http://wwwinfo.cern.ch/~rtb/arc.html]
(5) E. Yourdon, L. L. Constantine "Structure Design: Fundamentals of a Discipline of Computer Program and System Design"
(6) ARC program for AFS INFN.IT cell
[http://www.ts.infn.it/computing/local_paper/arc_for_afs_1-1.ps]
(7) A. Ventrella - Sistemi informatici distribuiti e Groupware "Un'Esperienza su AFS nell'Istituto Nazionale di Fisica Nucleare", Tesi di Diploma Universitario Facoltà di Ingegneria POLITECNICO di BARI, relatore Dott. M. Castellano.
(8) J. K. Ousterhout, "Tcl and the Tk toolkit" Addison Wesley Prof. Computing series, 1994.