Server concept for user driven management of networked multimedia applications

Thomas Merz, Andrea Bör
Institute of Communication Networks, Munich University of Technology
80290 Munich, Germany, e-mail: {merz, andrea}@lkn.ei.tum.de

ABSTRACT

The challenge in presenting and managing multimedia content in Local Area- or Wide Area-Networks is to reconcile the information probably hosted on different servers with different operation systems and different system configurations and conditions. To do this with the help of a kind of portal solves the problem to get access to the information and programs, but is still a very static method without any user driven management.
In this paper we will discuss a solution to add user driven management to networks with distributed applications, which need not to be explicitly designed for being remotely controlled. The aspects of scalability and security are a main part of this approach and will be treated respectively in the following sections. The aspects of high availability and load balancing are mentioned briefly due to being integrated until end of year 2000.

1. INTRODUCTION

A lot of institutes at universities have a lot of interesting and informative know how in the form of executable programs, written in programming languages like C, C++ or nowadays more and more JAVA, normally written as relatively isolated applications. The input methods of these programs are limited on interacting with their GUI (Graphical User Interface), which is commonly based on the specific client and having a lack of communication possibilities with global management services.
The problems caused by this kind of organization are presented in the following section 1.1. Our solution to these upcoming difficulties is illustrated in 1.2.

1.1. The challenges

The problem is to find a way to present these programs and ideas to an interested audience coming from inside (the own faculty or university) or outside from places all over the world and to let them interact with these applications. There are two main aspects to consider, when trying to find a specific application or information about an application:

- You have to know ***where*** to search for
- You have to know ***what*** to search for

The ***where*** may be an UNIX-user's home directory, as it is the standard for the educational branch, or if it is a special application, designed for Windows (NT, 2000,...) on a specially dedicated server, "hidden" somewhere in the cloud of the institute's LAN.
The next question is ***what*** to search. If you have no specific clue or keyword, it'll be an admirable act if you can find what you are looking for, using a LAN-wide search engine for example. Even if you find what you were looking for, it is the question, if you found *all* the hints you need to understand what a specific application is supposed to do. Maybe you now want to execute this or another application, to see how it's working, but normally you wouldn't be allowed to, due to technical or security restrictions. Consider you are an institute's internal, doing educational work in giving lectures, and you want to show your students during the lecture how some of the learned stuff works in practice, maybe realized by a student in his/her diploma thesis. You now have to have access to execute and to configure the application, you have exactly to know where a specific configuration file is to be found, what parameters to change and so on. Something will always go wrong in the precipitance of a lecture if you don't know the application very, very well.

1.2. The solution

The solution might be a kind of portal dedicated to be a contact point to all (or opened to the public) programs and information available. But in this case you only give access to *static* kind of information. So there is no way to adapt to the function by a change of parameters to one's (remember the staff member giving a lecture!) actual needs. Consequently you need a system which assists you in maintaining on the one hand a portal with search and linking functions and on the other hand a complete management system for editing the published information by the portal, for configuring your application's behavior, for defining access privileges to your application and so on.
A hybrid functionality of portal and management system is realized by KIARA, a server of the "Institute of Communication Networks" [1]. In order to keep things simple for everyone having to adapt his/her own application to the management service or to enhance the system, we decided to avoid using predefined services like CORBA, which premise of having to modify existing codes of programs.
A disadvantage of our solution in comparison with existing ones is, that we manipulate the remote controlled application from "outside", whereas a service like CORBA does it from the application's side. In this way we might have some difficulties in realizing some very special control actions.
Another reason for using our own system instead of existing services is the need of having a management application, which can work with (nearly) any operation system and program independent of the language it was written in (C, C++, JAVA,...), without having to deal with the system administrator of each dedicated server to require for additionally software components like ORB (Object Request Broker).This simply causes a higher level of involved people.

The remainder of the paper is structured as follows. Section 2 gives an overview of the KIARA system architecture. The technical details of our solution are illustrated in Section 3. Some security aspects are outlined in Section 4. In section 5 we describe the principles of communication between the KIARA system and the multimedia applications. We conclude with a summary of the system's features.

2. THE SYSTEM „KIARA"

2.1. The hardware & basic software of KIARA

KIARA temporary exists in the form of a single Linux driven server [2] connected to the institute's LAN. It can be reached via the Internet from anywhere in the world and can interact as management server with any connected server, if the "dedicated server part" of the management software is installed there. In the long run, it is considered to expand KIARA to a high availability and load balancing system. The high availability will be explained in section 5 of this paper.

As shown in Figure 1, KIARA supports the user (students, institute's members,...) with basic software requirements like:

- Web server [3]
- Database [4]
- Programming languages like JAVA including special add-ons [5], [6]
- XML-Server (eXtended Markup Language), which is in use for some other projects [7]
- WML/„WAP" (Wireless Markup Language, Wireless Application Protocol) as used for internet access of mobile systems
  Some parts of the KIARA-management system are accessible via WAP.
- Management system, as described in this paper
- SSL (Secure Socket Layer) for secure communication [8]

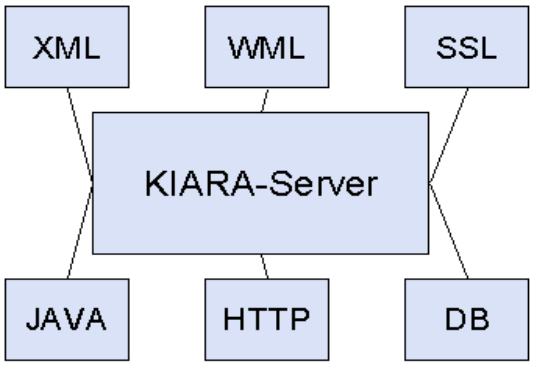

Figure 1. The basic services on KIARA

2.2. The management part

The main aspect of KIARA is the management system, giving the public and especial authorized users or user groups appropriate rights and privileges on certain programs and information (see Figure 2).

The authorization trends to three different kinds of users explained in this section:

- „public"
- „admin"
- „root"

Not mentioned in the following sections is a group of

users, which have parts of an administrator's rights. The responsible administrator and the supervising root restrain the privileges of these users to running programs in a defined way. They are slightly more powerful than the rights of a „public" user. These rights are normally given to students who need to run an application for doing research for their own work.

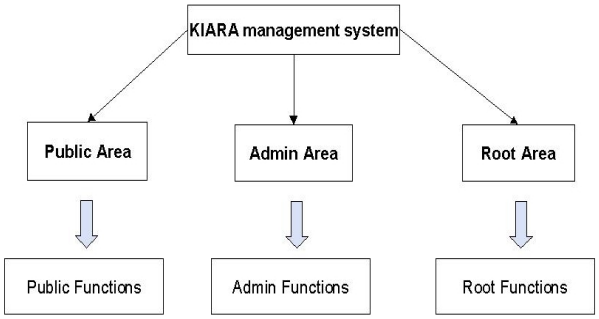

Figure 2. KIARA-management areas & user types

2.2.1. The public part

The public part is the area of the management system, which is open to everyone. No privileges are needed to execute functions in this area.
Everyone is allowed to use a database driven search engine, (using special search terms) to find opened to the public programs and information. The database entries display first basic information about the work and offer the possibility to link to more detailed information like a work's homepage and/or the author's email-address. Whether the author only offers information about his/her work or the possibility to execute his programs is managed in the „Admin"-Section.

2.2.2. The „Admin"-Part

An administrator (in the following sections mostly called "admin") usually is the author of published work on or via the KIARA-system.
After identifying (login and password) the admin is privileged to modify the information on his work published by the public portal function. Furthermore he/she is able to configure his application, even if it is running on a different server and operating system, to view the load and traffic his/her application is causing (only if the application is running on KIARA physically or the server's administrator has installed the appropriate software). He/She is able to configure the usage (generally yes/no, only privileged users etc.) of his/her applications.

2.2.3. The „Root"-Part

The management-„root" *can but need not to be* the physical root of the KIARA-system. Mentioning the root in connection with the management system means the person or group, having total control and access to the managing system and its interior functions. The root usually has all the possibilities an admin also has, with the difference, that he/she is able to alter the information and configuration of *all* users whereas an admin only can modify *his/her* work.
The root is privileged to

- activate, alter or delete administrator accounts of the management system
- control the administrator via traffic- or load–supervising
- control the monitoring functions of the whole system (each user entering the system can be traced throughout his/her whole visit, even if he/she is only using public functions)
- alter the configuration of the management system itself, which is a very powerful function.

3. TECHNICAL BACKGROUND

The sequencing parts describe the technical concepts and means of the server KIARA and the management system.

3.1. The HTML-Surface

The surface is presenting the KIARA management system in HTML (Hyper Text Markup Language, the standard web interface). Due to the idea of giving a system requiring nothing but a standard HTML-Client like Netscape's Navigator or Microsoft's Explorer, instead of having to deal with slowly loading JAVA-Applets on the client side, HTML has the advantage of being slim and thus following very fast in loading and acting.

3.2. The relational Database

The Database (MySQL [4]) running under KIARA's operation system Linux [2] is one of the interior and most important parts of the whole system. It stores the authentification parameters for a user's current session and the user's basic privilege parameters (login, password). Furthermore it stores the content of all public or privileged accessible information, like a project's info texts, links, application names etc. Only some very basic parameters (e.g. the connection parameters for the basic use of the database) are stored in not public accessible plain ASCII files (see section 5 for details).

3.3. The Perl Programming Language

Perl [9] is one of the programming languages used in the management system. This language is normally used when it comes to the need of having to interact from the surface to the operating system, e.g. when having to check the server's load, starting external functions, etc. The only thing needed by Perl, which might be critical on some system, is the package DBI (for database connects), but which can be easily installed within minutes by a system administrator, if it isn't already running, as it is a standard package of Perl 5.

3.4. The PHP Programming Language

PHP3 [10] is usually used when it comes to read or store content in interaction with the database. The easy and very fast database interface and simple integration in the HTML environment makes it an ideal part of the system and helps to make the whole management system very scalable.

3.5. The usage of cookies

Cookies are used on the client side to store some information about the user and parts of his/her temporary session identity (ID). With the help of cookies it is possible to trace the behavior and interests of a user, as well as to link him to the KIARA system and under special circumstances to a special dedicated server a program is running on controlled with the help of KIARA's management system. More details about this topic will be discussed in section 4 and 5.

3.6. The usage of shell scripts

Shell Scripts are sometimes used to make some filtering operation easier, especially when it comes to manipulation of a Linux' program's output (e.g. output of "*netstat*" and "*tcpdump*" for load and traffic monitoring)

3.7. The usage of cronjobs

The cron daemon is a basic tool in every Unix derived operation system. With its help, it is possible to start and stop programs in certain intervals (hour, minute, day, ...) or at special points of time.
The KIARA management system makes use of this behavior in numerous ways:

- Updating the day's statistical traffic results to the database
- Updating the day's load results to the database
- Updating the day's tracing statistics to the database
- Cleaning temporary log files
- Restarting parts of the system which parameters cannot be changed while being in service.

3.8. KIARA's scalability

Due to its "HTML-Perl-PHP" framework, the management system is easily scalable.
Despite to programming languages like Java or C, in which normally one routine is used to do all the work, the functions here are totally separate of each other. It's no big deal to remove one function (e.g. because of a security hazard) and all the other parts of the system will do their jobs as usual, it's just needed to change the executing rights of the function, so that the management system isn't allowed to execute it any more. This can be done in seconds.
On the other hand if you want to add a function, you do not need to compile the whole code new and you don't have to work in the existing code.
Just write your function in Perl or PHP3, copy it to the appropriate directory, make a link to it in the menu bar and add the existing security-function with ("include"/"require").
The function is now properly integrated to the management system.
The security model will be discussed in the following section.

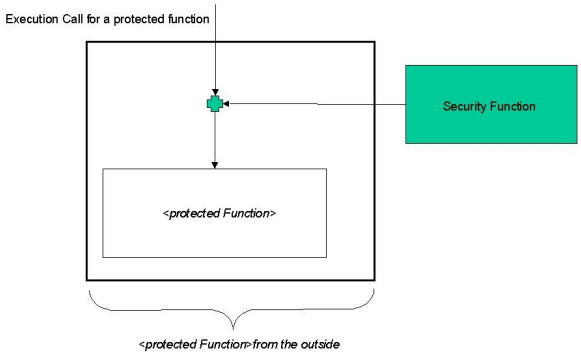

Figure 3. Including the security function

4. SECURITY FEATURES AND FUNCTIONS

A major aspect of each system offering restricted functions is of course its security. In the following paragraph is explained how KIARA's security model works.

4.1. The usage of SSL

The SSL (Secure Socket Layer) protocol [8] is activated in the standard case and offers encrypted communication between the client and the server. Basically SSL assures the user that his/her confidential data (passwords, ...) are transported unreadable for the public from the client to the server and vice versa.

4.2. The usage of the database

The database stores the user's permanent identification permission parameters like login and password. Each time a user logs in, the password is encrypted and will be compared to the encrypted correspondent part stored in the database. Analogical it happens to the login, with the exception, that the login is stored in plain format in the database. It's also checked if the client is using the correct port, which means if SSL is activated or not. If any one of these three parameters fails to return a true value, the system will not let the user log in.
As soon as all compared and retrieved data is considered as true, the following process is activated:

- Temporary session parameters are stored in the database
- Physical identification parameters of the client-server connection are stored in the database
- Part of the session's and physical parameters are stored as cookies on the client side.
- Some session, physical and content parameters are copied to a second database, which is used, if a program should be started on a different server than the KIARA management system is running on.
- For additional security of stored data, only local programs (running on KIARA itself) can communicate with the original database.
  The content of the second database is world wide readable, because no critical data is stored here.
  The reason for the second database will be discussed in the next section.

Every time a protected program is called an automated identification process is activated. The delivered identification parameters from the client are compared to the ones stored in the database. If any part of the process fails, the function will not be executed and the session will be terminated.
The identification schedule is shown in Figure 3.

The database is also used to give applications the possibility of using load balancing [11]. Every server, used by the KIARA-system for executing programs, can be used to send updates of its own workload to the database. This happens with little programs running in the background of such a system. These programs detect the server's load every minute and send the result to the KIARA management system, which stores the data in the database.
A program like the institute's „Simulation Manager", a basic server program for remote simulations, can read these up to date information, and, if a new thread is demanded by its applet, it can tell KIARA to start this thread on the server with the least average workload (and a suitable operation system for this program, of course).

4.3. The usage of time slots

Time slots are a part in the security model used to restrict the duration of a certain session to a configurable value (manageable by the management root). If an attacker gets access to valid session parameters and manages to convince KIARA that he/she is a legal system user this status will only last for the predefined duration.

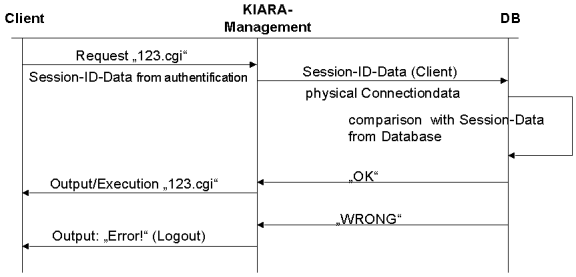

Figure 4. Identification process when requesting a protected function in the Admin- or Root-Area

4.4. The PHP3- & Perl Security Functionality

The security features described in the paragraph above need to be implemented, so that the functions, which depend on a security model, can be served well.
To achieve this goal three slightly different functions exist in the KIARA management system, two of them written and adapted for the use with any Perl-program (one of them for the use on external servers, see next section for details), and one for the use with any PHP3-program. To implement a new program you have to do the following steps:

- You need to know in which area (user group, admin, root) it is considered to work
- The next step is to integrate („require"/ „include") the security function in the code of the new program.

The program is now protected with the appropriate rights.

4.5. Basic Configuration Files

Some basic information on the configuration of the management system need to be stored in an other way than as content of the database, because it contains among primary information on how to connect to the database(s), where to find certain programs etc. In a normal Apache web server environment it is no big deal to be able to read files created or modified by the "Httpd" program. If this happens an attacker to the system is able to see communication internals like passwords, addresses, ports etc. To avoid this hazard the "Httpd"-daemon runs as a special user within a special user group. Only this user has read- and write-access to the configuration files.
To be sure no one uses the Apache user to execute some malicious programs, it is advisable to give no one the right to publish his/her work in the standard htdocs-Directory, which would premise to give this person a login password for the "Httpd"-daemon user.
Users of the system have to store and to execute their programs within the structure of their specific home directories. If they do so the executable programs like C- or JAVA- programs run under the rights of this user and have in this way no access to the basic configuration files, which are, as previously mentioned, only readable from the "Httpd"-daemon user.

5. COMMUNICATION BETWEEN KIARA AND THE APPLICATION

In order to start, stop or configure a user's application, it is necessary to manage a communication between the client and KIARA for identification purposes, between the client and the server the application is running on for controlling the program and between KIARA and this server once again for authorization reasons.

5.1. KIARA and remote applications

The communication between the client and the server is described in detail in section 4. The difference to a normal managing act, which happens to be completely local on KIARA itself, e.g. if the public accessible portal information of one's work is altered, it might now be necessary to keep in mind that the application is running on a complete different server. For this purpose KIARA has a kind of intelligent re-routing program running. When the user logs to his KIARA account it looks up its database and in this way knows which home directory on which server and which local user is dedicated to this KIARA user.
This re-routing mechanism is needed if the user wants to

- edit a configuration file
- start an application
- stop an application

All other management steps (e.g. adding a new configuration file to the desktop menu) are done completely local in KIARA in interaction with the original database.
If a new application or configuration is added it is put to the existing ones in the appropriate database field.
The database accepts only those entries, which are part of the user's physical home directory tree.
To make the changes accessible from the remote host, the user now has to log out and login once again.
The reason therefor lies in the security model.
It allows only KIARA based ("local") applications to read from the original database. All other hosts can get their information only from the mentioned second database, which is updated if the user logs in.

5.2. Options for remote management

Assuming the identification was successful, the management system offers the user a desktop in which he/she is able to maintain the account itself and the applications the user has configured. These possibilities are shown to the connected user so he can choose what to do. Users with administrator's rights for this application are able to:

- Alter and create configuration files of the program
- Define what files may be altered by a standard user
- Define special configurations in which a certain user can start and modify the behavior of the program by editing the program's configuration files, which are linked to his account.
- Control an application's load and traffic

Users with „user" rights for this application are able to:

- Edit configuration files and use them to run a certain program.
- Start programs in certain configurations (if defined so in their profile by the administrator)

All steps in managing the application are done with the help of a universal interface, which is able to identify the structure of a certain file. It can e.g. identify the type of value assignment to a variable

(„=", space character, tabulator character,...), how a commentary looks like and so on. Using this knowledge an input area with different fields and commentary parts (in which the purpose of each variable ⇔ value pair should be explained) is created in which the user can make appropriate changes. If it happens that a file type can not be displayed correct, a so called "Profimode" exists, which opens the configuration file like a normal text editor to make changes directly to the file without the indirection of the input interface. The starting and stopping of a program is also done with the interface-based system, in which the programs to execute and its parameters can be defined. The only action required on the user's side is to add the configuration file or application to his/her personal managing menu. The underlying technical details are managed by KIARA without participation of the user.

5.3. Performing actions on the remote server

Actions performed directly on the remote server are, as mentioned in 5.1, the altering of a configuration file and the starting/stopping of an application.
An "application" can also be a tool to set a display variable to the right value, to execute shell scripts remotely and so on. These features assume the remote controlled server has installed the "KIARA dedicated server part" under the appropriate user's home directory. As most of the management of a remote application is done locally on KIARA in cooperation with the database there must be an interface between the "local" and the "remote" management.
This interface is actually activated when a user follows the link of a configuration file to edit or an application to start or stop (The links are generated from the database entries). The KIARA management system reroutes the user to the database known server and home directory in which the "dedicated server part" of the KIARA management software is installed. In the same moment it transmits some authentification parameters of the current session to the client, which the client has to send back to the "dedicated server" every time he wants to perform an action directly on the remote host.
The remote host checks these parameters with those of the copied second database on KIARA (it also does some internal tests like time slot recalculations) for permission checking. After a database check, if the file or application, which the user wants to execute or alter, is assigned to him there, the appropriate execution or file altering function is performed.

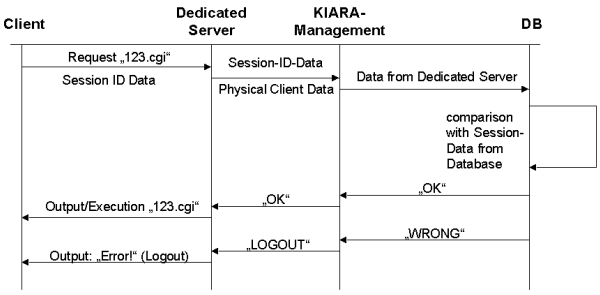

Figure 5. Client communicates via the dedicated Server with the KIARA management system

6. CONCLUSION

In the previous chapters we presented our solution of a multi functional server structure. One main emphasis is to make the system easily scalable, universally applicable, secure and redundant against failures. Although being predominantly used for remote simulation programs written in the programming language JAVA, we set up a system that is not linked to one specific kind of application, in order to be expandable and suitable for future needs. Further upcoming developments will even expand and improve KIARA. As some parts of the system for example offer the possibility to be controlled with the help of a HTML browser and WAP capable devices, it would be a valuable feature to generate these contents with the help of XML. Furthermore the aspect of load balancing and first of all high availability need to be reconsidered and enhanced to build a stable and reliable system which might not only be used at and by the Institute of Communication Networks, but at other institutes as well.

REFERENCES

[1] KIARA online
http://kiara.lkn.e-technik.tu-muenchen.de

[2] The Linux Operating System
http://www.linux.org

[3] Apache Web Server
http://www.apache.org

[4] The MySQL Database Server
http://www.mysql.org

[5] The JAVA Programming Language
http://www.javasoft.com

[6] JAVA with Apache
http://java.apache.org

[7] XML with Apache
http://xml.apache.org

[8] SSL with Apache
http://www.apache-ssl.org

[9] The Perl Programming Language
http://www.perl.com

[10] The Hypertext Preprocessor (PHP)
http://www.php3.de

[11] Load Balancing
http://www.linuxvirtualserver.org