SimBAD: A Virtual Lab based on the Jini Framework

http://dx.doi.org/10.3991/ijoe.v10i2.3210

F. Colace, M. De Santo and L. Greco
DIEM – University of Salerno, Fisciano (Salerno), Italy

***Abstract*—E-Learning is offering new approaches and opportunities in the field of education above all in the sector of the virtual laboratories. The opportunity to interact with the real instruments that are in a lab represents an effective way for the implementation of a teaching approach oriented to a problem solving strategy. So, a great number of software tools and environments have been developed with the aim to reproduce the learning style of lessons based on laboratory experiences. In this paper we exploit Jini Technologies to design and implement a distributed architecture for a virtual lab oriented for supporting the activities of an Electronic Measurement Course. The aim of this environment is offering to the students the opportunity to experiment a real interaction with the laboratory's instruments everywhere and every time. The environment offers services that allow the remotely programming of the instruments and manages the multi-access to the various tools. A prototype of the architecture and its services will be discussed and an evaluation campaign will be showed.**

***Index Terms*—e-Learning, Virtual Lab, Trial and Error approach, Distributed Systems**

I. INTRODUCTION

Thanks to the technological improvements of the last years, above all in the field of Web Based Services, distance education is an effective approach for supporting the traditional formative processes. E-Learning platforms and services achieve a very high quality level and can integrate the traditional learning and teaching processes [1][11][12] with novel approaches and services [13][14]. In particular, there are some services oriented to implement the pedagogical approach of a laboratory [15] [16] [17] [18]. In general, these frameworks aim to reproduce an emulation or simulation of typical educative processes of a laboratory but do not offer a real and effective interaction with the laboratory's instruments. This kind of problem penalizes strongly distance learning environments that cannot reproduce a real "trial-and-error" methodology. On the other hand, there is a real interest in virtual lab development: in fact education center with a great number of students have to introduce and manage a huge number of labs and instruments for a long time: it could be really expensive and many technicians have to be involved in the management process. The opportunity to offer remote labs' sessions to the students can reduce the management costs and introduce a more effective and rational use of a laboratory [7]. In this way, the laboratory can be used anytime from students according to well-defined usage policies. The wide diffusion of the World Wide Web and Java language allowed a simple management and control by remote of laboratories' instruments [5][6][8]. Departing from these considerations, the main aim of this paper is the introduction of a distributed environment for supporting the main activities of a laboratory for electronic measurement. The proposed distributed framework, named SimBAD (Simple Broadly Available Distributed Learning Support Environment, but also Simulation is bad), offer services to teachers and students for living real laboratory experiences according "trial and error" and "problem solving" approaches. For the development of this kind of framework the design phase is capital: all the actors of the system and their services have to be defined. For the implementation of the SimBAD system Java and Jini technologies have been adopted: Java allows the implementation of a multiplatform, secure and distributed framework as SimBAD. The adoption of the Jini technologies solves the management of the various services displaced on different hosts in various nets and of laboratories instruments [9]. In SimBAD, the "service" points out both electronic devices and software applications. Such services do not offer only the possibility to interact remotely with the instruments but create, by the use of interfaces and technologies, educational and pedagogic mechanisms that characterize the learning processes in a laboratory. In SimBAD there are two different kinds of services: the first ones propose a simple interaction with the instruments. In this way, the students can take confidence with the instruments and learn their basic functionalities. Obviously, this approach is not different from a simple simulation even if, in this case, the realism of the learning process is guaranteed by the real remote interaction among users and instruments [19]. In addition other services, that allow a more sophisticated control of the equipment by the students, have been developed. With these services, named high-level services, students can understand all the potentialities offered by the instruments and have a full control of the same. In this way, for example, students can program remote tools and receive the results of elaborations on their remote terminal: so, the learning process, offered from this type of services, has the same value of the traditional one. In this paper we will describe in details the SimBAD architecture, the implemented services and the results obtained by the use of this framework. The paper is organized in the following sections: in section 2 and 3 we will give a brief description of SimBAD and its main features. In section 4 we will describe some implemented services. In section 5 some experimental results will be showed. The conclusion section will close this paper.

II. SIMBAD: A VIRTUAL LAB BASED ON JINI FRAMEWORK

In the last years the architectures of the informative systems are evolved from a centralized schema to a distributed model, supporting in this way decentralization and co-operation. The reasons for the crisis of the centralized model are numerous. Those principals are based on the collapse of the prices of hardware tools, the distributed system's continuity and quality of the service in comparison to the centralized ones and the greater reliability of the net infrastructures [2][3][4]. The designing of a system with these characteristics asks for the resolution of various problems like for example the choice of communication protocols and the general synchronization of each component that are in the framework. An effective approach to the distributed system can be the adoption of software framework as a middleware, a software bus that can implement a virtual channel and manage different services physically displaced on different hosts. The wrapping mechanisms, commonly offered by middleware, allow exploiting legacy applications and the gradual passage from old to new architectures. In this paper the Jini framework has been exploited as middleware**.** Jini is a general framework for the distributed environment and its main aim is the implementation of a plug and play network [10]. The main entities of a Jini system are the services, the lookup service and the services clients. In Jini a service can be both a real device both a software service. In fact in the Jini framework every device (as a scanner, a printer, a cellular telephone, a DVD and also a toaster) or a software application is a service and its interface explains what can be done with it. Jini can realize a "spontaneous networking": a set of distributed services that can change in dynamic way. The most important service, in a Jini system, is called "Lookup service". The main aim of this service is to keep track of all the enjoyable services in a Jini community, and has an essential role in the search of the services that are inside in the same community. In this way if someone wants to make available a service must simple register the proxy service in the lookup service. This registration phase is called "join of the service" in the network. In this way a client can simply ask for a service to the lookup service and choose among those available. Clients can use Jini services through objects called Java proxies. We can imagine a proxy of the service as a driver that resides inside the service itself. The opportunity to download the proxy of a service is the key idea of Jini framework: in this way electronic devices can be used without the installation of their drivers. This feature will be very important in the designing of the distributed laboratory because in this way the various laboratory instruments can be used by the students downloading only its service proxy. The proxy of a service communicates through the network with a "back-end" code written by the service supplier. This code allows the communication, with a protocol established by the supplier of the service, between the proxy of the service and the same service. The proxy can be an RMI stub that enables the communication with the back-end of the service implemented by a remote object. In such case, the process of elaboration, related to the execution of the remote methods, is performed on the server of the service. The choice to realize the proxy as a stub RMI can be useful when the service is purely hardware as a measurement instrument. In this case user can send a command to the instrument invoking a remote method on the proxy, but the real execution will happen on the server of the service that, in this case could be the same tool. Another very important concept, in a Jini environment, is the group: a Jini group assembles some services that can cooperate in order to resolve a specified assignment. The set of join, lookup and discovery protocols provides the necessary infrastructure to the federation of Jini services but this is not enough to guarantee its effectiveness. To such purpose Jini introduces a further software level allowing the construction of a distributed system stable and able to recover failures. This new software level is constituted by the following concepts: Leasing, Remote events and Transactions. The Jini leasing process grants in loan, for a well-defined period, a service to a user. In this way Jini forces who asked for a service and still wants to use it, after the end of the loan period, to request a renew. This policy of resource's administration avoids deadlock processes. The lease concept makes a Jini federation stronger because bad functioning of some system services or client cannot compromise the stability of the whole community. Moreover Jini technology makes available some mechanisms of notification of remote events. L*ookup service* implements the mechanism of the remote events and can notify to the interested parts the addition, the disappearance or the change of the state of various Jini services. Finally transactions mechanism gathers a whole connected application in such way only if all the tasks are completed the states of all involved application change. So at the end of a transaction the system can know with certainty the real state of applications. In a distributed environment the transactions mechanism resolves the problem of a partial failure of an operations subset. The transaction Jini mechanism guarantees the four essential ownerships of the transactions: atomicity, consistence, isolation, durability. A transaction manager works as supervisor on the result of these operations, notifying to the entities that have recorded such operations the result of the transaction. If the transaction doesn't succeed, it will have to be repeated. In the next section more details about the SimBAD architecture will be furnished.

III. SIMBAD: THE PROPOSED ARCHITECTURE

In this section of the paper we describe the architecture of the distributed environment SimBAD. In the designing phase of the architecture we have followed these logical steps:

- To analyse the problem of the distance learning process paying particular attention to virtual lab and to individualise the system specifications
- To define the actors in the environment
- To define a modular structure for the environment
- To identify the principal classes (and them interactions) necessary to the implementation of the use cases
- To implement the use cases

The first phase of the project has brought us to individualize the basic functionalities that the system had to offer and the actors that would use the system. From the point of view of the actors of the system we identified three typologies of actors:

- User
- Administrator
- Group Administrator

For each of these actors we have identified the interest use cases depicted in figure 1:

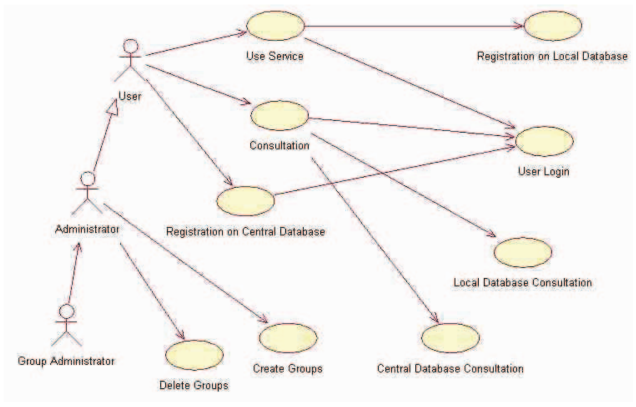

Use Service
Registration on Local Database
User
Consultation
User Login
Administrator
Registration on Central Database
Local Database Consultation
Group Administrator
Delete Groups
Create Groups
Central Database Consultation

Figure 1. Use Case Diagram for SimBAD Jini Environment

We have identified for the user actor three client sessions:

- "Use service" session: the user, after a logon procedure, can access the list of the services available for his profile. In practice this use case is composed of three phases: the choice of a server for the acquisition of the service, the real choice of the service and the use of the same.
- "Registration on a central database" session: the user on the basis of his profile is able to manage the information storing them on a central database
- "Consultation" session: it allows the consumer to consult, in base to his/her profile, the stored data

Besides the user actor, that contains inside teacher and student profiles, the system foresees the presence of other two typologies of user: the administrator and the group administrator. Such figures manage the groups and the belonging consumers to the various groups. After this analysis phase, it was possible to define the software modules that must compose the system architecture. With the term module we intend the whole software components used for the construction of a precise part and functionality of the system. In general the design choice of a modular architecture allows the reduction of the planning complexity and the opportunity to replace or introduce some modules without repercussions on the entire system. The SimBAD system has been projected thinking about three fundamental modules:

- The services module
- The authentication module
- The communication module

In order to better understanding and explaining the architecture of the system it is opportune to describe in the next paragraphs in detail the various modules.

A. *The Service Module*

All the software components, that allow realizing the enjoyable services in the system SimBAD and the distribution of such services, belong to this module. In SimBAD we have two types of software components: groups and services. Groups can store and furnish services. In practice they represent the access point to the system from which it is possible to use services. We can imagine groups as special servers that are able to distribute services to authorized users. In our architecture only the group administrator and administrator can create, manage and delete groups. In general inside the SimBAD system we can have N groups and n available services. From the point of view of the implementation the group concept coincides with the same one defined in the Jini framework: in fact our group concept matches with the lookup service implemented in Jini. So our idea is to associate to various lookup service SimBAD groups. In this way services can record themselves in their reference groups using the join Jini protocol. For such motive every service must have in its structure a software module that records the proxy of the service inside the lookup service. We defined this software module as "service server". Besides every service is characterized by the presence of some attributes that associate descriptive information to the same service allowing a policy management. Every service has to implement the software SimbadService interface. Such interface declares only the method start( ) that subsequently is implemented by the proxy of the service. The client after the download of the service can invoke really such method. Figure 2 shows the temporal sequence of the events that characterize the recording of a service in a lookup service and the start of the same from a client.

So we can describe the general structure of a service in the environment SimBAD in this way: a server of the service (Recording and Leasing Management Module), the attributes associated to the service and the proxy. There is also a Back End Object that guarantees the communication between Client and Service. Every service as previously said has to implement SimBADService interface.

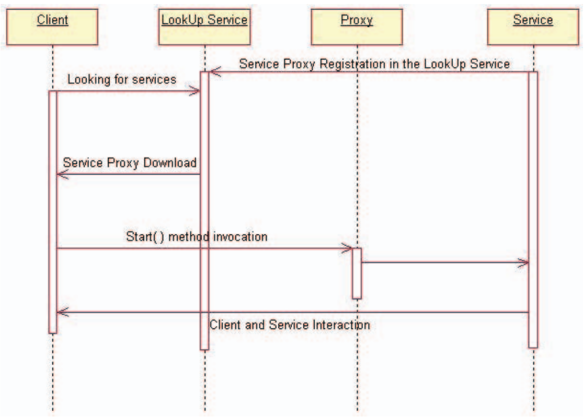

Client
LookUp Service
Proxy
Service
Service Proxy Registration in the LookUp Service
Looking for services
Service Proxy Download
Start( ) method invocation
Client and Service Interaction

Figure 2. Sequence Diagram that shows phases and classes involved in registration and use of services

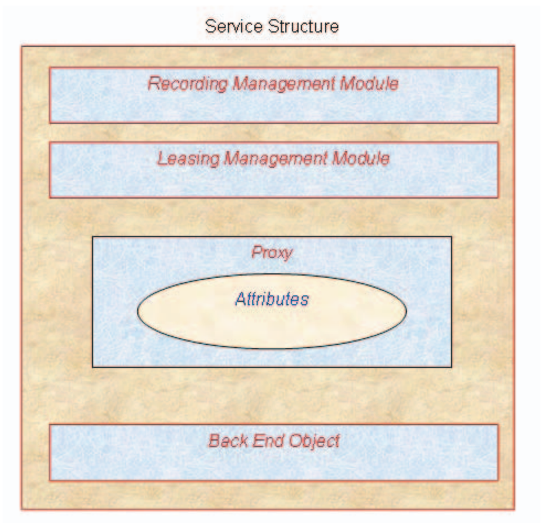


Figure 3. SimBAD service general structure organization

B. *The authentication module*

This module aims to manage the resources access policy in SimBAD environment. It allows the consumers to gain access to the authorized groups and services. In the system a database, managed by the system administrator, contains user profiles composed by the following parameters:

[Username] [Password] [Group] [ServiceLevel] [StoreLevel] [DocViewLevel]

Similar parameters define services and groups profiles. In the group parameter we define, in fact, a group names while numerical values, that countersign the level of the services and interaction with the database of the contents, are associated to the remainders parameters. These numerical values define the access level of every service and define users that can use them. The parameter [StoreLevel] discriminates what documents can be filled in the group database by user. The parameter [DocViewLevel] defines the documents stored in the central database that user can read.

C. *The communication module*

This module is a software layer allowing the communication among client and server processes displaced on different JVM. As previously said the logical phases that characterize the use process of service from a client can be so schematized and are depicted in figure 4:

- Client connection to the SimBAD system
- Search of the groups to which the consumer has the right to enter for the use of the services
- Search of the services inside an accessible group selected by the consumer
- Selection of a service
- The service proxy download
- The start of the service

The communication module supports all the previously depicted phases and implements the protocols and rules for every action mentioned in the figure 4.

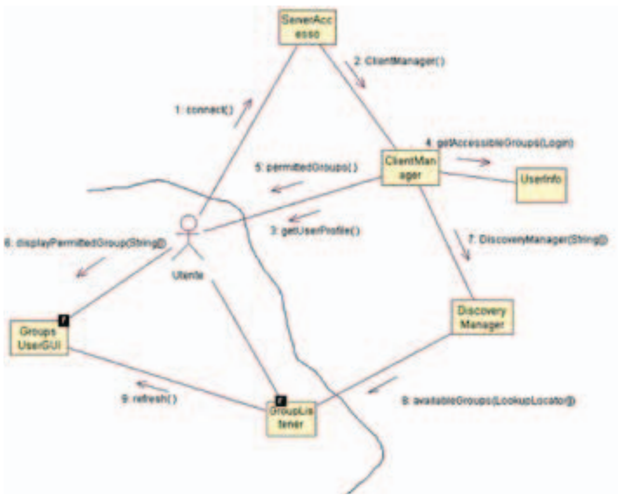


Figure 4. Connection to SimBAD Environment and communication protocol

IV. SIMBAD: THE SERVICES

In this section we describe in details some services designed and introduced in SimBAD environment. For this first prototype we have designed services that aim to help teacher and students in the laboratory practice of Electrical and Electronics Measurements course of engineering faculty of University of Salerno. Many students attend this course and so it is very difficult to offer enough laboratory practices. In fact there are not many instruments and so students must wait long time before using them. Some services that can be used by remote in any moment need. So we have implemented a Group called "Measurements Group" where services can register themselves. We have designed and implemented two different typology of services: the basic services and the advanced services: the first are oriented to students that need to simple interact with the instruments, while the second are oriented to more expert students that want control more in detail the instruments. In the next section we will explain the details of various services.

A. *The basic services*

The basic services, implemented for the environment SimBAD, allow simple interactions of the students with remote tools that are present in a measures laboratory. For the realization of the virtual lab prototype we have selected tools type 488.2 connected to a computer through an IEEE-488 bus. The services have to allow the student to interact with the basic functionalities offered by the instruments. In the realization of this type of service a first and serious problem is to regulate the accesses to the various instruments. Obviously we want avoid the contemporary access to the same resource from two or more students in order to avoid trouble interferences. So we implemented a special service, not available to the students, that manages, in automatic way, communications with the bus 488. In other words we implemented the communication module previously described. We denominated this service ServerBus "service". The other services must use this one in order to communicate with the various tools. From an architectural point of view the only service that has to be present on the computer physically linked with the bus IEEE-488 is the ServerBus, while all the other services can be distributed on different computers. This architectural organization balances the load on the entire environment and in particular on the computer where the 488 card is situated improving the performances of the same one. As previously said we implemented some services that allow the students access to the basic functionalities of the instruments. In this paper we will explain in detail Fluke45 service that manages the interaction between students and the Fluke45 multimeter instrument.

B. *The Fluke45 service*

The FLUKE 45 is a five digits multimeter with two displays and sixteen different measure typology. This instrument is a multipurpose instrument and can effect measures of tension and current (middle value and TRMS), of resistance and of frequency. It can also verify the functionality and the characteristics of semiconductor diodes. The two displays can contemporarily measure two different parameters of the same signal and visualize the obtained measures. By remote (not only from the same LAN but also using a modem connection) student, using this service, can train himself with the tool and its measure configurations. Students can use the service through a web browser and in the same web page can also download handbook or support contents that describe in more details the experience of laboratory. As previously said the Fluke45 service doesn't physically have to be present on the computer connected to the bus and so to the instrument, but it must use ServerBus service to access to the multimeter. As all the SimBAD environment services also the Fluke45 service has a well-defined structure constituted by the followings modules:

- Registration module
- Leasing management module
- Proxy
- Back-End process

A further module, called ServerBus Module, is added to the general structure in order to manage the communication with the service ServerBus. The registration service phases in the system follow the scheme described in figure 5.

As previously said more students could have unloaded the proxy of the service and therefore contemporarily to ask for the access to the same service. This case could bring to incongruous situations. We introduce inside the system a queue manager that can manage the access of the students to the service. So student must verify the service availability (free or busy) before using it. If the service is free user can block and use it. The queue is managed with the FIFO (First In First Out) rule. In order to avoid the permanent block of the service every consumer can use the service only for a well-defined time period. When the assigned time expires student can renew the use of service or leave the control of the same one. The implementation of this mechanism is based on leasing concept offered by Jini technology. Obviously we must also guarantee that the instruments that share the same bus IEEE-4888 don't contemporarily try to communicate on the bus. The services have methods that allow communicating to the other interested services the occupation of the bus. Obviously after the end of the operation the service launches a signal that communicates the release of the bus. The generic consumer, after the authentication phase to the interest group, can unload the proxy of the service and finally use it according the scheme depicted in figure 6.

Fluke45 service offers to the students a friendly graphic interface that faithfully reproduces the panel of the tool. This interface is showed in the figure 7.

In the virtual panel two additional buttons are present: the lock and unlock buttons. They allow to block or to leave the control of the instruments accord with the philosophy described in precedence. The virtual interface is composed of two main panels called display and control. The display panel represents the panel of visualization of the tool and show all the results of the effected measures, the instrument state and the address of the tool on the 488 bus. The control panel, instead, represents the Fluke45 push-button panel and with it the student can actually send commands to the remote instrument. With this type of service student can remotely reproduce the same experiences that he would have conducted in the laboratory. In fact Fluke 45 service allows a total control of the instrument and of its functionalities. In this way students can easily understand the philosophy of the experiment in which the instrument is inserted. In order to exploit all the

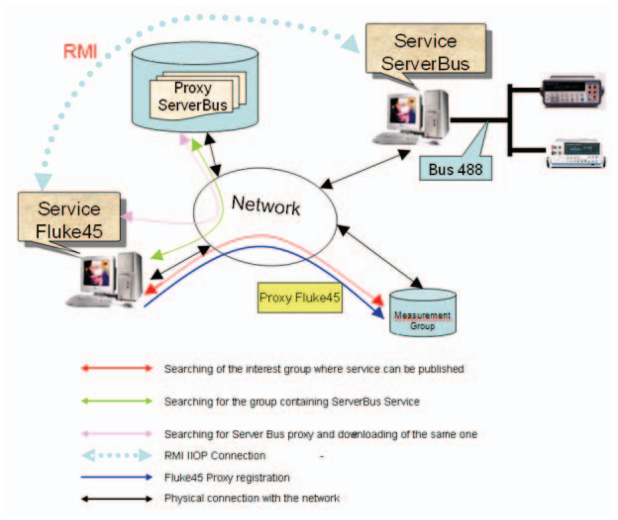

RMI
Proxy ServerBus
Service ServerBus
Bus 488
Network
Service Fluke45
Proxy Fluke45
Measurement Group
Searching of the interest group where service can be published
Searching for the group containing ServerBus Service
Searching for Server Bus proxy and downloading of the same one
RMI IIOP Connection
Fluke45 Proxy registration
Physical connection with the network

Figure 5. Fluke 45 service registration phase

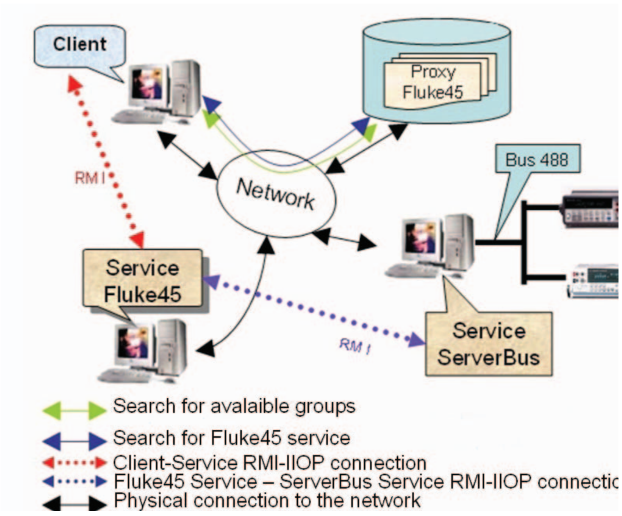

Client
Proxy Fluke45
Bus 488
RMI
Network
Service Fluke45
RMI
Service ServerBus
Search for avalaible groups
Search for Fluke45 service
Client-Service RMI-IIOP connection
Fluke45 Service – ServerBus Service RMI-IIOP connectic
Physical connection to the network

Figure 6. Client request and use of Fluke 45 service

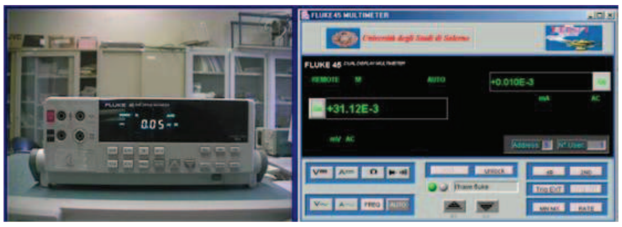


Figure 7. Real instrument and the virtual graphic interface offered to the students by SimBAD Fluke45 Service

potentialities of the 488 bus and to make the experience of laboratory from remote closer to real one, we created another SimBAD service called trigger. With this service students can effect synchronized measurements between various involved instruments. This service use and synchronize various services sending opportune signals to all the instruments involved in the experience of laboratory.

The service offers a simple interface where students can choose the address of instruments, and so of the services, that want to control. After this choice students can interact in the same time with the selected services and instruments. Such functionality has numerous applications as for example the measurement of the tension for every phase of a tri-phase system.

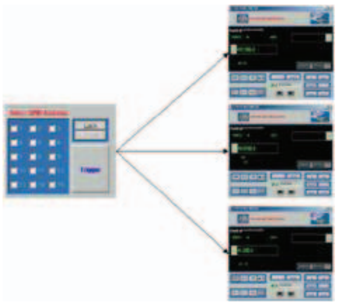


Figure 8. Trigger Service and synchronization among various services

C. *Advanced services*

After the basic services we implemented some services for students that have the need of better understanding the instrument operation. In order to achieve these results the advanced services do not offer a simple interaction with instruments but allow more control on the same one. In the next paragraph we will explain in more details one of them: the Exec service.

D. *The Exec service*

Students, after understanding the basic notions of the instruments, can develop some own programs in order to really control and make the most of the same one. So we designed and implemented a service that allows performing software applications, created by the student, on remote instruments. These applications can send data from the remote instruments to the students and vice versa. This service, denominated Exec, allows performing any executable applications on remote predisposed instruments. Obviously Exec service is a SimBAD service composed by four principal parts:

- Registration module
- Leasing management module
- Proxy
- Back-End process

At the starting this service produces on client side a graphic user interface that allows using the functionalities of the service (Figure 10).

Student is not physically located on the computer that controls the bus, so he does not know previously what devices are connected to it. So clicking on a special button countersigned by the light bulb is possible to physically get the list of the present tools on the bus (figure 11).

At this point student can prepare, on his computer, program (in any programming language) that must work on remote instruments. Two typologies of programs can be performed through this service:

- without student interaction
- with student interaction

The programs without interaction do not foresee the introduction of data from the student during the execution on remote instruments. The other program typology requires, when necessary, data input inserted by the student during the execution on remote instruments. This happens when a program needs of some parameters, for example the address of remote instruments with which to communicate, in order to advance in its execution.

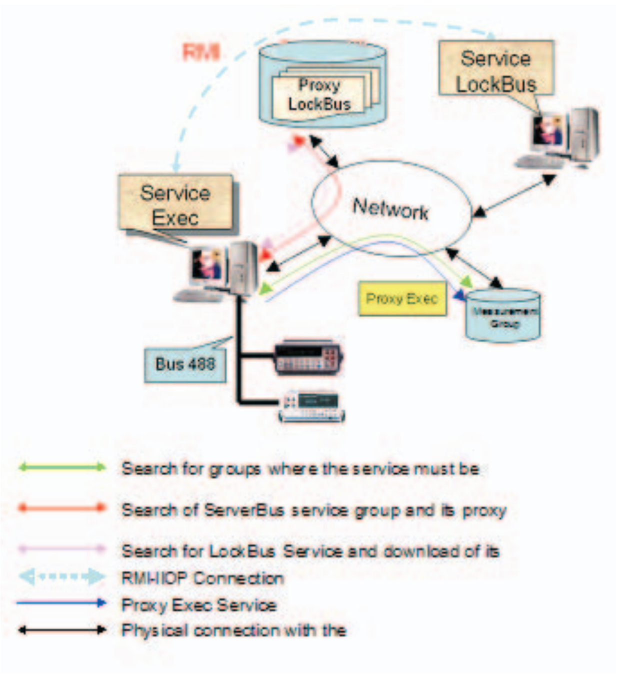

RMI
Proxy LockBus
Service LockBus
Service Exec
Network
Proxy Exec
Measurement Group
Bus 488
Search for groups where the service must be
Search of ServerBus service group and its proxy
Search for LockBus Service and download of its
RMI-IIOP Connection
Proxy Exec Service
Physical connection with the

Figure 9. Registration of Exec Service in SimBAD Environment

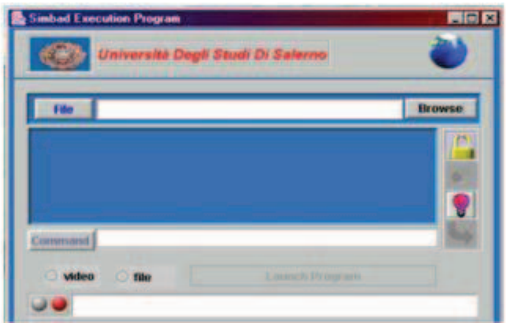
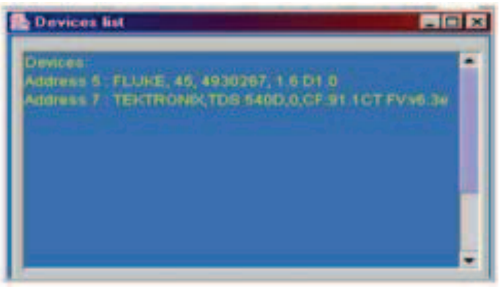


Figure 10. Starting Graphic User Interface of Exec Service



Devices list

Devices:
Address 5 : FLUKE, 45, 4930267, 1.6 D1.0
Address 7 : TEKTRONIX,TDS 540D,0,CF:91.1CT FV:v6.3e

Figure 11. List of devices on the bus IEEE-488

Through the graphic interface the student can:

- Select the application to perform
- Select the format of desired output (on video or on file)
- Specify the destination file in which service memorizes the output of the execution of program
- Launch the program in remote
- If the student program foresees the introduction of parameters from the outside, a text area denominated "Command" allows to insert them.

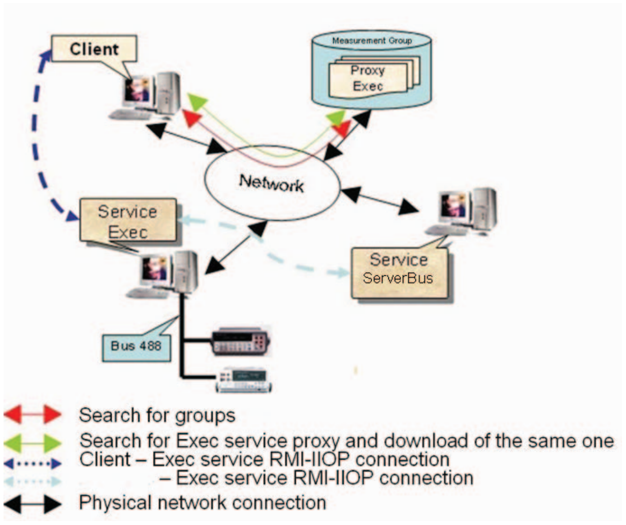

Client
Measurement Group
Proxy Exec
Network
Service Exec
Service ServerBus
Bus 488
Search for groups
Search for Exec service proxy and download of the same one
Client – Exec service RMI-IIOP connection
– Exec service RMI-IIOP connection
Physical network connection

Figure 12. Execution of Exec program

In order to use service student must select and lock the instruments involved in the laboratory experience. After this phase service must transfer student program on the computer where it must be launched. This computer launches the program and interacts with instruments and has to support communication between client and service if interactions need. At the end of experiment results have to be sent to the student. The transfer of the executable file happens in three phases:

- the executable file has to buffer in a byte array
- an RMI call is effected to a remote method and the name of the file and the array of byte have passed as parameters
- the file is reconstructed and memorized on a specific computer linked to the 488 bus and finally launched

V. EXPERIMENTAL RESULTS

In our experimentation we have considered a blended course: Introduction to Electronic Measurement (about 50 students) belonging to the faculty of Engineering and a comparison with traditional approach was developed. We started the course and the end of each topic an exercise laboratory has been proposed. The class has been divided in two groups: the first one, composed by 23 students and named blue group, used a traditional approach while the other one, composed by 24 students and named red group, used the virtual lab for the exercise. At the end of each exercise a test has been submitted for measuring the average knowledge level gained by the two groups. At the end of the course a questionnaire about the laboratory experiences has been submitted. The obtained results are depicted in figure 13.

As we can see the obtained results show as the Virtual Lab allows an effective formative approach and indeed increases the students' knowledge level. Also the analysis of the questionnaire about the laboratory experiences is interesting. It was designed according to the Likert Scale and the main results obtained can be so summarized: in general the blue group is quite satisfied by the laboratory experience, but about the 80% of the students of this group complains about the loss of time to support the laboratory exercises. Another problem detected by the group is the impossibility of being able to repeat the experience in the laboratory. On the other hand, the red group

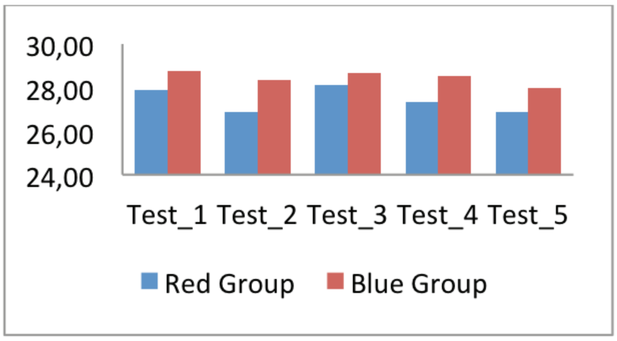


Figure 13. Test: Obtained Results in the range 0 (lower mark) – 30 (higher mark)

showed a real satisfaction for the laboratory experiences and appreciated the opportunity to practice at any time of the day. Some students of the red group underlined that they preferred to have real contact with the instruments.

VI. CONCLUSION

In this paper we have described a distributed environment called SimBAD. This multipurpose distributed environment aims to offer services to various clients. Services belong to various groups that support a precise class of user. Firstly we designed our architecture keeping in mind the main characteristics of a distributed envinroment. In this phase we used a description language in order to describe the various use case, sequence diagram and comunication protocols. After this first phase we designed a group and services to support the course of Electrical and Electronics Measurements. The real aim of these services is to support students in laboratory experience. In particular they allow to control and interact with remote measure instruments and through this services teachers can prepare laboratory experiments. Students through these services can interact with instruments as if they were in a real lab. We designed and implemented other services, called advanced, that reproduce more sophisticated mechanism of interaction between students and instruments. These services, in fact, reproduce the typical pedagogical approach of laboratory experiment: the "trial-and-error" approach. In this way students can have their laboratory experiments without lose nothinq in quality, by remote, also from their homes. These services allow, also, a better management of laboratory instruments and an increase of activity hours of laboratory. The used technology (Java and Jini) guarantees the portability of services.At present the described prototype of SimBAD environment is under test: about fifty students of Electrical and Electronics Measurements course use our tool to have their laboratory experiments. The future developments of our environment can be summarized in the following steps. At the first we aim to implement new services and groups. We are designing services, for measurement group, able to control VXI instruments. Obviously these new services must communicate with the existing ones. We are implementing also new groups and new services for other courses.

REFERENCES

[1] F. Colace, M. De Santo, M. Vento, Evaluating On-Line Learning Platforms: Case Study, Proceedings of HICSS36 conference, IEEE Press USA, 2003

[2] G. Coulouris, J. Dollimore, T. Kindberg, Distributed Systems, Concepts and Design", Addison Wesley, 1994

[3] A. Goscinski, Distributed Operating Systems, the Logical Design, Addison Wesley, 1993

[4] S. Mullender, Distributed Systems, Addison Wesley, 1993

[5] L. Cristaldi, A. Ferrero, V. Piuri, Programmable instruments, virtual instruments, and distribuited measurement systems: What is really useful, innovative, and technically sound, IEEE Instrum. Meas. Mag., vol 2, Page(s): 20-27, 1999

[6] H.J.W. Spoedler, Virtual instruments and virtual environments, IEEE Instrum. Meas. Mag., vol 2, Page(s): 14-19, 1999

[7] S.E. Poindexter B.S. Heck, Using the Web in your courses: What can you do? What should you do?, IEEE Contr. Syst. Mag., Vol 19, pp 83-92, 1999 http://dx.doi.org/10.1109/37.745773

[8] J. M. Dricot, Ph. De Donker, M. Dierickx, F. Grenez, H. Bersini, Development of Distributed Self-adaptive Instrumentation Networks Using Jini technology, IEEE Intern. Works. On Virt. Int. Meas. Sys., Budapest, Hungary May 19-20 2001

[9] W. Keith Edwards, "Core Jini 1.1 Beta Specification", Prentice–Hall

[10] Jini Sun Homepage: http://www.sun.com/jini/index.html

[11] Regueras, L.M., Verdú, E., Munoz, M.F., Perez, M.A., de Castro, J.P., Verdú, M.J., Effects of Competitive E-Learning Tools on Higher Education Students: A Case Study, Education, IEEE Transactions on Volume: 52, Issue: 2, 2009 , Page(s): 279-285

[12] Gaudioso, E., Hernandez-del-Olmo, F., Montero, M., Enhancing E-Learning Through Teacher Support: Two Experiences, Education, IEEE Transactions on Volume: 52, Issue: 1, 2009 , Page(s): 109-115

[13] Colace, F., De Santo, M., Ontology for E-Learning: A Bayesian Approach, Education, IEEE Transactions on, Volume: 53, Issue: 2, 2010 , Page(s): 223 - 233

[14] Chtouki, Y., Harroud, H., Elkhalidi, M., Samir, B., A service composition framework for providing e-Learning services, Information Technology Based Higher Education and Training (ITHET), 2010 9th International Conference on , 2010 , Page(s): 219 - 225

[15] Khot, R.A., Choppella, V., DISCOVIR: A Framework for Designing Interfaces and Structuring Content for Virtual Labs Technology for Education (T4E), 2011 IEEE International Conference on, 2011 , Page(s): 121 - 127

[16] Choppella, V., Brahmajosyula, V.K., Vutpala, M., Kole, S., Process Models for Virtual Lab Development, Deployment and Distribution, Technology for Education (T4E), 2011 IEEE International Conference on, 2011 , Page(s): 293 - 294

[17] Alexiadis, D.S.; Mitianoudis, N., MASTERS: A Virtual Lab on Multimedia Systems for Telecommunications, Medical, and Remote Sensing Applications, Education, IEEE Transactions on Volume: 56, Issue: 2, 2013, Page(s): 227 - 234

[18] Fraile-Ardanuy, J., Garcia-Gutierrez, P.A., Gordillo-Iracheta, C., Maroto-Reques, J., Development of an Integrated Virtual-Remote Lab for Teaching Induction Motor Starting Methods, Tecnologias del Aprendizaje, IEEE Revista Iberoamericana de, Volume: 8, Issue: 2, Publication Year: 2013 , Page(s): 77-81

[19] Villar-Zafra, A.; Zarza-Sanchez, S.; Lazaro-Villa, J.A.; Fernandez-Canti, R.M., Multiplatform virtual laboratory for engineering education, Remote Engineering and Virtual Instrumentation (REV), 2012 9th International Conference on, 2012 , Page(s): 1-6

AUTHORS

Francesco Colace is with the DIEM – Università degli Studi di Salerno, Fisciano, SA 84084 Via Papa Giovanni Paolo II 132, Italy, (fcolace@unisa.it).

Massimo De Santo is with the DIEM – Università degli Studi di Salerno, Fisciano, SA 84084 Via Papa Giovanni Paolo II 132, Italy, (desanto@unisa.it).

Luca Greco is with the DIEM – Università degli Studi di Salerno, Fisciano, SA 84084 Via Papa Giovanni Paolo II 132, Italy, (lgreco@unisa.it).

This article is an extended and modified version of a paper presented at the 2013 IEEE International Conference on Teaching, Assessment and Learning for Engineering (TALE2013), held 26-29 August 2013, Bali Dynasty Resort, Kuta, Indonesia. Submitted 26 September 2013. Published asr e-submitted by the authors 09 March 2014.