Distributed Platform for Control of Robots at Distance

Daniel E. Vera Gonzalez
Universidad Católica Andrés Bello
Caracas, DF 1080, Venezuela

and

Wilmer Pereira
Universidad Católica Andrés Bello
Caracas, DF 1080, Venezuela

I. Abstract

This project consists of making a research about how JINI technology can control a device remotely. These devices are controlled through a local network and different platforms. However this project is limited to work with a robot that represents these devices.

The main objective of this investigation is to construct an application that allows interacting with a robot from JINI technology. Specifically is used the Robotics Invention System of Lego Mindstorm. Lego Mindstorm allows to construct robots that work in a specific environment.

JINI is a network technology. It helps to make the network more flexible because facilitates using and administrating labors. JINI follows a client/server model. Server publishes the services and clients obtain these services to use them.

In this document once is made an analysis of the necessary tools, an experiment is made. This experiment includes a robot able to explore a specific land in search of obstacles. This robot is remote control by an application that simultaneously is controlled by an operator located in the network. This operator aside from manipulating the robot can visualize the land composition. This allows the operator to see if there are or not obstacles in the surface and to know where these are located. Also there are certain numbers of users waiting for the resource. These users can visualize the exploration done by the operator. The user can act as operator once the resource is released.

II. Objectives

1. Investigate and study JINI technology.
2. Investigate and study Java, especially using RMI (Remote Method Invocation).
3. Investigate and study devices compatibles with Lego Mindstorm.
4. Design a JINI component, this should allow interacting with a robot.
5. Make some experiments. These experiments have to establish an opened JINI and robot communication. Specifically, the experiment consists in develop a remote application. This application must be able to control and receive information from a robot. This robot must explore a limited land and identify the exact obstacles position within this area.
6. Design and build a robot that could fulfill with objectives named before.

III. Important Information

RMI (Remote Method Invocation)

In this section is presented short information about RMI. RMI is the main mechanism of communication used by JINI Technology. Therefore, also is used in this project in order to invoke methods that allow manipulating the robot remotely.

The basic idea of RMI is that objects running in a single JVM (Java Virtual Machine) are able to invoke methods in others JVMs. Also, each JVM can be located in the same machine or different machines connected to a network.

The RMI architecture consists of three layers: Stub layer (client) and receiver object (server), Remote Reference layer and transport layer (figure 1).

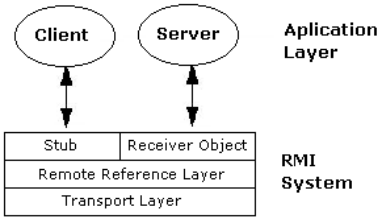

Figure 1: RMI Architecture

Stub: It's an Interface between client application and remote object. Its responsibilities are: to initialize the calls to the remote object, to serialize arguments to send them through the network, and deserialized the arguments given back in the calls.

Receiving Object: It's in charge of translating invocations of the remote reference layer, as well as to manage the answers. Between its activities they stand out: to deserialize the arguments, to make calls to the remote object methods, and to serialize the return values.

The remote reference layer: It's responsible of implementing the communication policy. This can be represented by different types: invocation unicast point -point, strategies of reconnection.

The transport layer: It's responsible for the establishment and maintenance of the connection, to take care of incoming calls, and to establish the communication for the incoming calls.

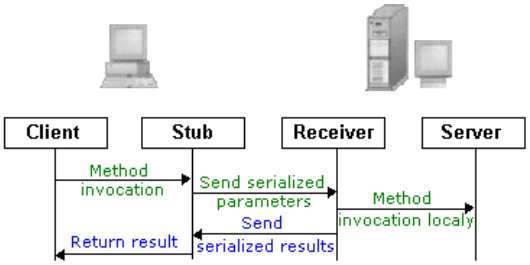

Figure 2: Method invocation process in RMI

As shown in figure 2, the client invokes the method from the object stub located in the local JVM. This object is in charge to serialize the parameters in order to send them to the receiver object. The receiver object invokes the method locally. Once it obtained the result or the corresponding exception, the receiver object sends value to the stub object. At the end the Stub deserialize the result and returned it to the client.

JINI Technology

As mentioned before, the experiment is based in a remote control of a robot using JINI Technology. To make the experiment successfully we have to know a little bit about how JINI works. Following are mentioned the most important concepts of JINI Technology. These concepts will be used after this chapter to describe some characteristics of the experiment.

The JINI Technology is based on a simple concept: devices should work together. They should interconnect without problems, drivers, operating systems problems, and wires or strange connectors. Sun Microsystems (2000).

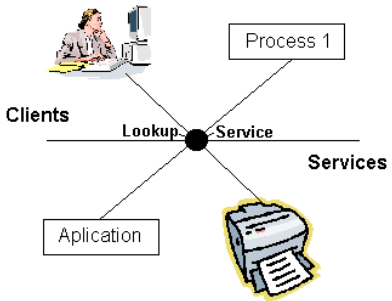

Figure 3: JINI System Overview

As show in figure 3, a JINI system is a distributed system based on the idea of federating groups of users and the resources required by those users. The overall goal is to turn the network into a flexible, easily administered tool with which human and computational clients can find resources. Resources can be implemented as hardware devices, software programs, or a combination of the two. The focus of the system is to make the network a more dynamic entity that better reflects the dynamic nature of the workgroup by enabling the ability to add and delete services flexibly.

JINI Technology has many important elements. These elements are shown as follow:

Services: the most important concept within the JINI architecture is that of a service. Sun Microsystems (2000) define a service as an entity that can be used by a person, a program, or another service. A service may be a computation, storage, and a communication channel to another user, a software filter, a hardware device, or another user. Two examples of services are printing a document and translating from one word-processor format to some other.

Lookup service: Services are found and resolved by a lookup service. Sun Microsystems (2000) The lookup service is the central bootstrapping mechanism for the system and provides the major point of contact between the system and users of the system. In precise terms, a lookup service maps interfaces indicating the functionality provided by a service to sets of objects that implement the service.

Protocols: Sun Microsystems (2000) express that the heart of the JINI system is a trio of protocols called “discovery”, “join”, and “lookup”.

1. Discovery: occurs when a service is looking for a lookup service with which to register. (figure 4).

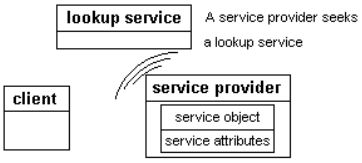

Figure 4: Discovery

2. Join: occurs when a service has located a lookup service and wishes to join it. (figure 5)

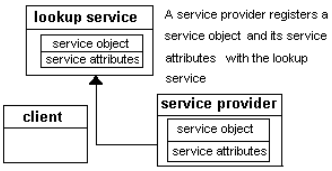

Figure 5: Join

3. Lookup: occurs when a client or user needs to locate and invoke a service described by its interface type (written in Java) and possibly other attributes (figure 6).

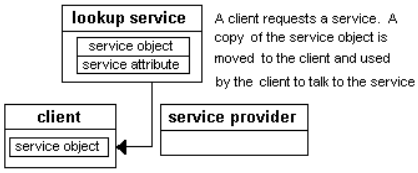

Figure 6: Lookup

Once the client gets a service object, it could invoke that service directly without a lookup service.

Lego Mindstorm

Once known the entire network mechanisms used in the experiment, it’s necessary to briefly know what Lego Mindstorm is and what its main components are.

The main component is the RCX, which is the brain of the system (shown in figure 7). It can be programmed from a computer. The computer transmits the programs to the RCX from a wireless communication using infrared signals. According to Dave Baum (2000) the RCX can be divided in three layers: hardware layer, system ROM layer, and firmware.

The hardware layer is a device's bottom layer. It's composed by following mechanisms: a microcontrol or CPU, a screen or LCD, Memory. Also it can use up to three independent motors and four different types of sensors (Touch sensor, light sensor, rotation sensor, and temperature sensor) (figure 7).

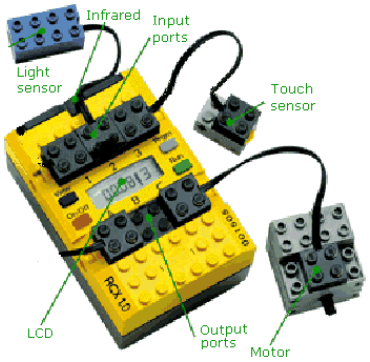

Figure 7: RCX components

Actually, there are a lot of firmwares. Some of them are firmware standard, LegOS, and pbForth. In this experiment is used firmware standard. This is because reasons explained in later sections.

The Standard Firmware is used when programs are written using anyone of compatible environments (code RCX, NQC, Robolab or Spirit.ocx).

The standard firmware has different types of programming. The main ones are: using NQC and code RCX (used by Lego Mindstorm software).

Although the use of these tools (NQC, Lego Software) facilitates the RCX's programming, these tools can't make advanced tasks like controlling the RCX on real time. For this, it's necessary to transmit RCX's pure code from the computer thus allowing to manipulate the device in a while given. Kekoa Proudfoot (1998-1999) deciphered the codes necessary to control the RCX from the computer using firmware standard.

In order to be able to send these commands, it's necessary to install a program that allows manipulating computer's serial ports. Also is required a program that allows to transmit RCX's commands using RCX communication protocol. At the moment in Internet exists some programs able to establish this communication, among them "`send.c`" created by Kekoa Proudfoot (http://graphics.stanford.edu/~kekoa/rcx/tools.html) and "`rcx.jar`" created by Dario Laverde http://www.escape.com/~dario/java/rcx

IV. Implementation

In this project was used the Unified Process methodology. It was proposed by Booch, Jacobson and Rumbaugh (1999). In This way the project is divided in four main stages: Inception, Elaboration, Construction and Transition. In figure 8 is shown every phase and each iteration.

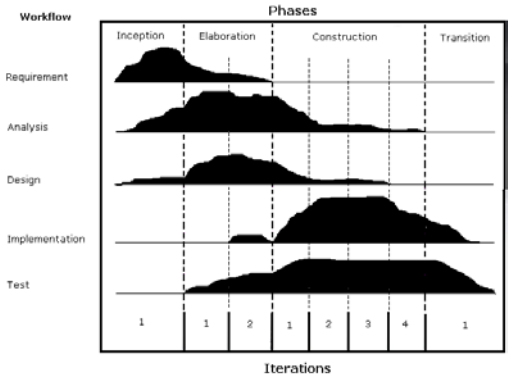

Figure 8: Phases and iterations of project development

Inception Phase

Development:

- Several meetings where established with specialized people in the area of robotics and nets. These meetings helped to proposed some experiments that allowed to show all functionalities of JINI technology.
- Several remote interviews with expert people of JINI technology and Lego Mindstorm. Some of them are engineer Iain Shigeoka a list distribution member of Sun Microsystems and Kekoa Proudfoot experts of Lego Mindstorm technology. This allowed to know the experiment feasibility and every opinion of each one of these people about the project

Results:

- The main objective of the experiment must be a constant communication between user(s)-robot. This allows evaluating the communication between devices through JINI Technology. For this same reason A.I. (Artificial intelligence) are not gotten up in the project.
- The application must be able to remotely control and visualize the robot's actions on an efficient way. This must support a considered client's number. The clients can accede at the same time to the system. The client application should has Aspects of graphical interface to facilitate learning process.
- All user located in the local network (LAN) can control the robot. The application must administer the accesses of these clients and obtain that all of them can control the robot.
- The minimal necessary resources for the execution of the experiment are: a computer that works like client and server, a robotics Invention system of Lego Mindstorm including a touch sensor, two motors, and two rotation sensors.
- The robot must move by the land in search of possible obstacles. It will be remote-control from a place within a local network. The robot has to recognize all obstacles by crashing with them.
- The land must be uniform, preferably smooth.
- Just one user can control the robot in a while (mutual exclusion). In order to obtain this, a waiting line has to be implement. This line allows the user who has been more time in delay to obtain the resource to control the robot (once the operator has released or lost the resource).

Elaboration Phase

Investigation and design of the local components (Iteration 1)

Development: In this iteration, the work focuses in investigating and designing necessary components for a robot manipulation locally. There are many activities made in this stage. Some of them are:

- Searching information in bibliography and Internet about Lego Mindstorm.
- Studying different ways of RCX´s programming
- Obtaining firmware.

Results:

- Concluding about using standard firmware better than LegOS. In order to be able to control the robot in a while given, it's necessary to send signals from the computer to the RCX while the robot is exploring the land. LegOS has a denominated communication protocol LNP (Lego Network Protocol) that allows to send and to receive messages to the RCX from the computer. Although this can control the robot in a given period of time, exists one application that obtains it at the moment, but is written in C language. JINI supports only services and clients who run in JVMs which rejects this possibility. However, standard firmware has an application developed in Java by Dario Laverde who allows to send code RCX and control it in a while given.
- Definition of what kind of command has to send from the infrared tower to the RCX. Reached the conclusion that the programs had to be stored in the RCX. Then these programs are executed from the computer sending code RCX. The exploration results are received by the user checking the sensors' state from the computer.
- Obtaining different robot's designs which can evaluate the displacement of rotation sensors versus displacement per time. Also it's desired to evaluate the RCX position (Horizontal or Vertical).
- Obtaining a local architecture.

Investigation and design of remote components (Iteration 2)

Development: In this iteration is investigated and designed all components of technology JINI. These components were able to remotely manipulate the robot. Between the activities are:

- Running practical examples obtained from Internet. These were observed at http://www.jini.org. Also were studied projects made by Jan Newmarch (2001) who had worked jointly with Lego Mindstorm and JINI.
- General Configuration of JINI environment. This includes an execution of a Lookup server, and development of a client and server able respectively to obtain and to publish a service
- Design of the application's Interface.

Results: The most important result of this stage is a distributed architecture shown in figure 9:

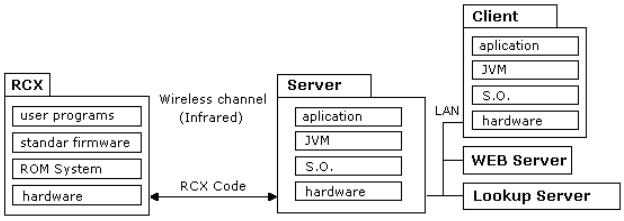

Figure 9: Distributed system architecture.

Construction Phase

This phase is formed principally by three essential processes. The first one is the development or programming code. Then all unit tests of each component. At last, the redesign of all components if necessary.

Locally components development and test (Iteration 1)

Development:

- Firmware installation. This firmware is the standard Lego version 1.0.
- Construction and test of different robots. This should be made taking care of the results obtained in the preview phase. The test of each robot is made according to three specific programs: the first program moved forward the robot until it encounters an obstacle. The second and third program moved the robot 90 degrees to right and left respectively. The design which has the best performance would be the definitely prototype. In this step is also evaluated the robot movement by rotation sensors Vs. time movement as RCX's position.
- Environment's construction where the robot will work.
- Robot's programs development that permits fulfill the device exploration.
- Installation of javax.comm. and rcx.jar packages.

Results:

- Selection of the Omega Centaury robot-design. As can be observed in figure 10, this robot has in its front a touch sensor able to recognize any obstacle at the exploration process. It also has two rotation sensors that will permit measures precisely robot's displacement. RCX vertical position do not permits robot lose contact with infrared tower when it's moving or turning.

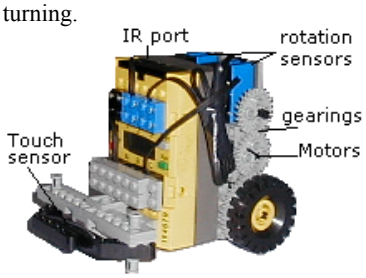

Figure 10: Omega Centaury Prototype

- The land is a card, which is divided in squares of 30 Centimeters.
- It's obtained a robot's local manipulation from an application made in Java. (sending code RCX). This is reached from packages java.comm and rcx.jar.
- It's obtained RCX's programs that allow robot exploration. Table 2 explains each one of them.

Program	Function
GoAhead()	It moves robot forward until the robot find an obstacle or arrive to a free space
Back()	It moves robot backward if the robot find an obstacle
Middle(message msg)	It can move robot forward or backward depending on message.
Turn90Lft()	It turn left robot 90 degrees
Turnr90Rgh()	It turn right robot 90 degrees

Table 2: RCX's programs specification

Once the experiment runs, is observed some limitations:

- Lights make some Interference when RCX is communicating with Infrared tower. This is aggravated by RCX's vertical position (which places the infrared port pointing up). Reason why ceiling lights interferes in process.
- It is necessary that the infrared tower be placed at the communication threshold in order to avoid loss of packages.

Application development and test (Iteration 2)

Development: In this iteration the procedure is to develop an application that served like interface between the operator and the robot. Once constructed this application, it is adapted to the packages obtained in previous iteration. The purpose is obtaining a robot local manipulation from that application.

Results: the most important result in this stage, is to take a locally control of the robot through a graphical application. This application is shown in figure 11.

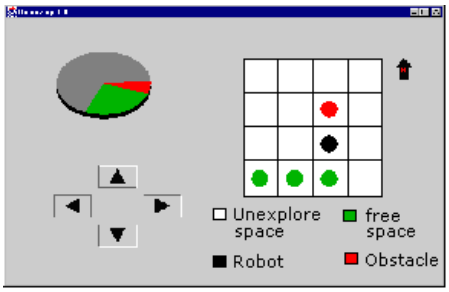

Figure 11: Application prototype

Remote component development and test (Iteration 3)

Development:

In this stage, a JINI component is created. This component allows obtaining a remote object. Some activities are explained following:

- A Lookup Server Installation who allows to publish and to offer services in a network.
- A Web server installation that allows transmitting archives from a place to another one. This server is necessary because it transmits a service from a service provider to client (in this case, service means an stub object). This service is sent using HTTP protocol.
- A service provider development. This can be able to publish a service in any server located in the network.
- A client development. This is able to look a service anywhere within a network as obtain it and use it according to the service specifications.

Results: the most important result obtained in this stage is a JINI component. This is shown in figure 12.

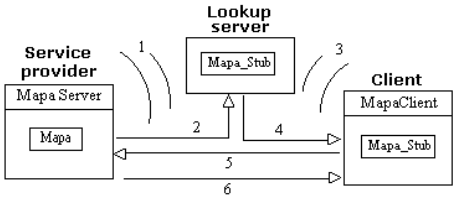

Figure 12: JINI component behavior

1. The Service Provider makes a search of all the Lookup Server existing in the network (Discovery).
2. The Service Provider publishes the "Mapa_Stub" service (it contains all exploration methods) in all found Lookup Service (Join).
3. The client makes a search of Lookup services (Discovery).
4. The client gets a "Mapa_Stub" service (Lookup).
5. The client invokes some exploration methods using RMI. These methods are executed in the server side.
6. The Service Provider sends a remote notification that indicates exploration's results.

All components Adaptation and test (Iteration 4)

Development: once culminated each component separately (Application, robot's manipulation locally, and JINI component), this phase is in charge of connecting each one of these named components. This stage allows obtaining a remote robot manipulation using an application linked to a JINI component.

Results:

- A client represented by an application that obtains a remote service (From JINI). This service is a centralized map that represents the exploration state as well as all methods to manipulate the robot
- A server (connected to the robot by infrared signals) able to send notifications to the clients about its exploration status
- Implantation of a mutual exclusion model. A single user can manipulate the robot in a while given (operator). The other users enter in a waiting line ·
- Any user in delay can visualize the exploration made by the operator.

Transition Phase

Development:

In this phase is made the experiment assembly with its respective tests as well as small improvements in the system. Some activities are explained following:

- The experiment's assembly using multiple platforms (Solaris, Linux, Windows).
- System Improvements. If the resource is not used by the operator by a certain time, the resource is assigned to the first user of the tail ·
- Respective System tests.

Results: in figure 13 is shown the application receiving some information about an executed exploration. In figure 14 is

shown the experiment's assembly given by a client who controls a robot invoking methods located in the server.

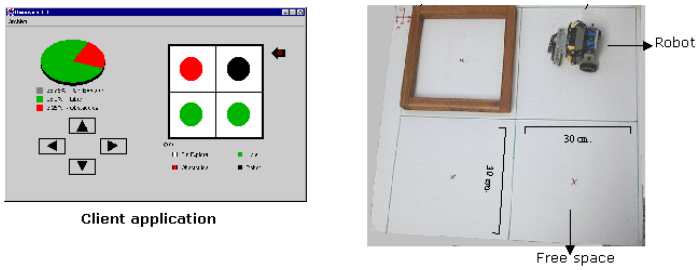

Figure 13: Remote exploration using client application

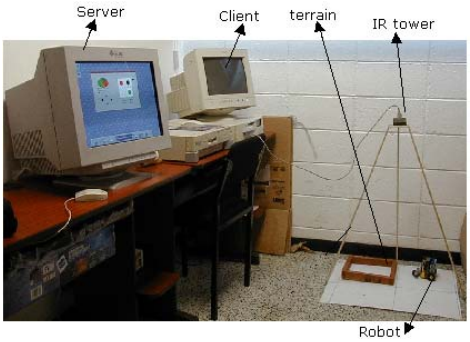

Figure 14: Experiment running

Once the experiment ran several times the following results are obtained:

- Communication's loss between the RCX and infrared tower. The experiment's conditions must be reduced to a place little illuminated, as well as a robot position and correct distance between the tower and the RCX.
- Once the robot makes several displacements in the land, can lose the correct position. It occurs because there is not an external datum point.

Along this document is studied that JINI technology can offer hardware device services as software services.

The RCX as device has processing and memory capacity, but it is too limited. Therefore the RCX cannot run a Java Virtual Machine and it cannot store JINI packages that allowed directly device's communication with a network. Consequently the RCX needs to interact with a centralized computer that acts like proxy between network and device.

The communication between infrared tower and the RCX's infrared port is very limited too. Therefore an advanced use is not recommended when a constant communication movable elements is needed.

The JINI technology advantages that are observed in the experiment are shown next.

- Flexibility was shown at use of many network communication protocols
- It permits a more dynamic network. Clients doesn't need to know where service is
- It extends the advantages of a client/server model. It facilitates to create services centralized vs. distributed (depending on the case).
- It has a security list so it is not necessary to configure firewalls that allow restricting accesses to no authorized resources.

V. References

- Dave Baum, Michael Gasperi, Ralph Hempel & Luis Villa (2000). Extreme Mindstorms: an advanced guide to Lego Mindstorms.
- Artima Software, Inc. (1996-2001). FAQ for JINI-user mailing list. http://www.artima.com/jini/faq.html.
- Jan Newmarch (12 de Junio de 2001). Jan Newmarch's Guide to JINI Technologies versión 2.8. http://pandonia.canberra.edu.au/java/jini/tutorial/Jini.xml.
- Carlos Beltrán Gonzalez (1998-1999). RMI mano a mano con SSL: Construyendo aplicaciones distribuidas seguras. http://java.programacion.net/taller/joa_rmissl.php
- Sun Microsystem (2000). Jini™ Architecture Specification versión 1.1. http://www.sun.com/jini/.
- Ivar Jacobson, Grady Booch & James Rumbaugh (1999). The Unified Software Development Process. Adisson Wesley Longman, Inc.
- Sun Microsystems Inc. (2001), commApi package Version 2.0.2 for Solaris Sparc & Version 2.0 for Microsoft Windows. http://java.sun.com/products/javacomm/index.html.
- Dario Laverde (1999). rcx.jar. Extraído de la página web: http://www.escape.com/~dario/java/rcx/
- Kekoa Proudfoot (1998-1999). RCX Internals. http://graphics.stanford.edu/~kekoa/rcx/#Protocol
- Cay S. Horstmann & Gary Cornell (2000). Core Java 2 Volume II – Advanced Features. Prentice may.