The Creation of Graphical User interface for Mobile platforms

[1]Baydaa Jaffer Al-Khafaji
Computer Science Department,
College of Education for Pure Science/Ibn Al-Haitham, University of Baghdad, Baghdad, Iraq
bjkh68@yahoo.com

[2]Dr.May A. Salih
Department of Mathematics and Computers
Basic Education collage- University of Babylon, Babylon, Iraq
may_8096@yahoo.co.uk

Omar Adel Rashid
omar.adel.rashed.b@gmail.com

Abdulla Adil Rashid
abdullah.adil@ihcoedu.uobaghdad.edu.iq

Abstract

Aim of the Work Creating graphical user interface for Lime (Linda in a Mobile Environment) package in order to facilitate this package to be use by the user. In order to explore the difficulties that may the user face with this package.

Key word: Linda in a Mobile Environment, TSpaces, Local Server.

1. Introduction

The present clients request omnipresent system get to free of their physical area. Where Wireless gadgets, for example, PCs, telephones, individual advanced collaborators, smartcards, watches and so forth, are increasing wide prominence. Their registering capacities are developing rapidly, while their size is contracting, permitting a considerable lot of them to is held by hand. These gadgets can be associated with remote systems of expanding transfer speed, and programming improvement packs are accessible that can be utilized by outsiders to create applications [1]. The joined utilization of these advancements on close to home gadgets empowers individuals to get to their own data just as open assets whenever and anyplace. Applications on these kinds of gadgets, be that as it may, present moving issues to architects. Gadgets face brief and unannounced loss of system availability when they move; they find different has in a specially appointed way, they are probably going to have rare assets, for example, low battery power, slow CPU speed and little memory that should be misused

effectively; they are required to respond to visit changes in nature, for example, new area, move to an alternate gadget, high inconstancy of system transfer speed, that will stay by requests of extent lower than in fixed systems. When creating appropriated applications, originators don't need to manage issues identified with dispersion, for example, heterogeneity, versatility, asset sharing, and so forth. Middleware created upon arrange working frameworks furnishes application architects with a more significant level of reflection, concealing the multifaceted nature presented by dissemination. Existing middleware advancements, for example, exchange arranged, message-situated or object-situated middleware these innovations have been planned and are effectively utilized for stationary conveyed frameworks worked with fixed systems, however they don't have all the earmarks of being appropriate for the portable setting. Right off the bat, the association natives, for example, circulated exchanges, object solicitations or remote method calls, expect a high-transmission capacity association between parts that is continually accessible. In portable frameworks, conversely, impeachability and low transmission capacity are the standard as opposed to an exemption. Besides, object-arranged middleware frameworks, for example, CORBA, [2] basically bolster synchronous point-to-point correspondence that requires the customer requesting an assistance, and the server conveying that administration, to be fully operational all the while. Toward one side, the accessibility of a fixed system is accepted, and its offices are abused by the portable foundation this is the situation with Mobile IP [3]. At the opposite end, the fixed system is missing and all the system offices (e.g., steering) must be executed by depending just on the accessible versatile hosts, this is the presumption made by inquire about on specially appointed systems [4]. The attributes of the remote correspondence media, e.g., constrained data transmission and continuous detachment; favor a decoupled and artful style of calculation. Calculation is decoupled in that it is relied upon to continue even within the sight of detachment, a regular occasion right now. Calculation is artful in that it abuses availability at whatever point it opens up. This impossible to miss style of calculation requests another way to deal with the making of dispersed applications. Close by this type of physical portability, in which the hosts meander over the physical space and change the topology of the system, specialists are giving expanding consideration to different types of intelligent versatility also. In the last case, frequently alluded to as code portability [5]. The substance being moved is the code running on the hosts, which are typically not permitted to move.

2. Characterization of Distributed Systems

An appropriated framework comprises of an assortment of segments, circulated over different PCs (also called hosts) associated through a PC arrange. These segments need to collaborate with one another, all together, for instance, to trade information or to get to one another's administrations. In spite of the fact that this collaboration might be manufactured straightforwardly on organize working framework natives, this would be unreasonably perplexing for some application engineers. Rather, a middleware is layered between circulated framework segments and system working framework segments; its errand is to encourage segment collaborations. Fig (1). Illustrates an example of a distributed system.

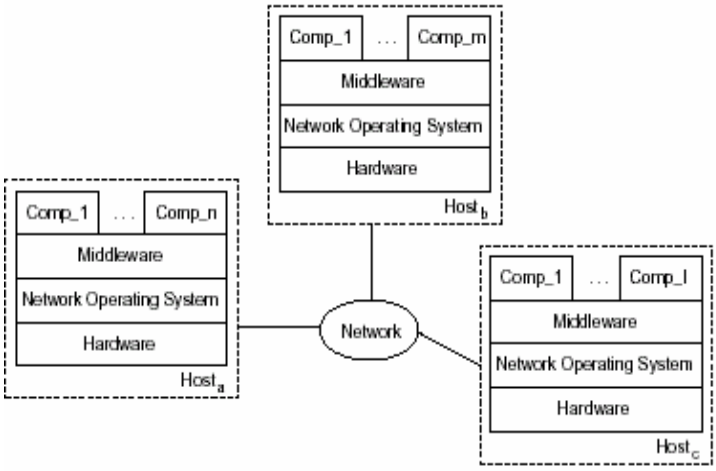


Fig. 1. Example of a distributed system [Emmerich 2000a].

This meaning of distributed system applies to both fixed and mobile systems. To comprehend the numerous distinctions existing between the two, there are three ideas covered up in the past definition that significantly impact the Type of middleware systems embraced: the idea of gadget, of system association and of execution setting. These ideas are delineated in Fig (2). [6].

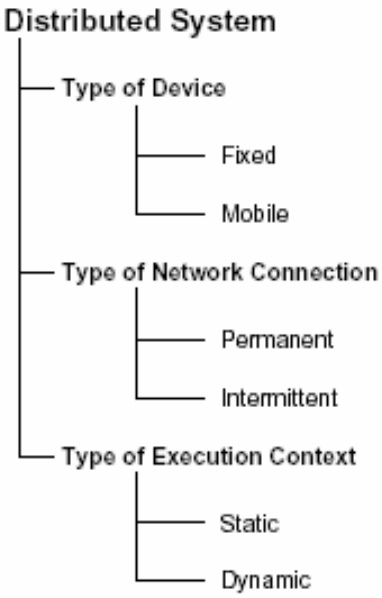
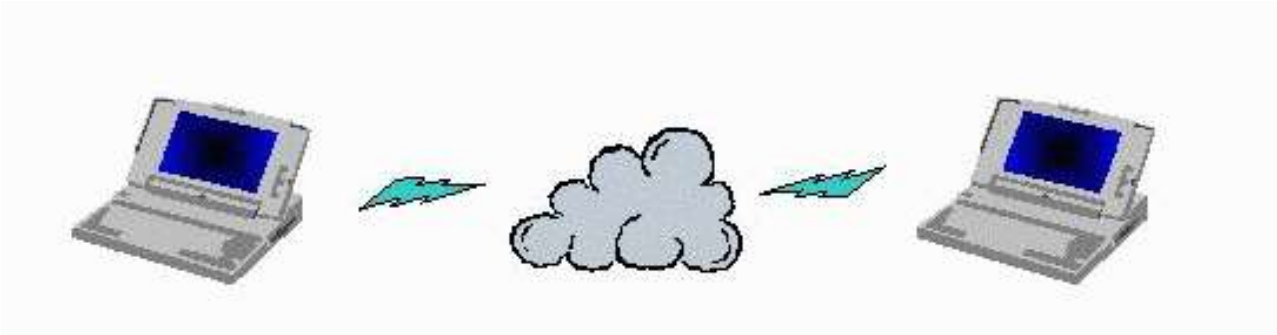


Fig. 2. Characterization of mobile distributed systems.

3. Spaces

Is arrange middleware for the new period of pervasive registering. A concise portrayal of TSpaces would be, a system correspondence cushion with database capacities. Be that as it may, it is regularly simpler to depict it as far as what it does [7]. It empowers correspondence among applications and gadgets in a system of heterogeneous PCs and working frameworks. TSpaces gives bunch correspondence administrations, database administrations, URL-based document move administrations, and occasion warning administrations. It is actualized in the Java programming language and along these lines it naturally has arrange omnipresence through stage autonomy, just as a standard kind portrayal for all information types. In progressively specialized terms, TSpaces broadens the fundamental Linda Tuplespace system with genuine information the executives and the capacity to download both new information types and new semantic usefulness. The notable highlights of the TSpaces framework are:

- Tuplespace Operator Superset: TSpaces actualizes the standard arrangement of Tuplespace



administrators: read (tuple), take (tuple), and compose (tuple). What's more, it incorporates both blocking and non-blocking variants of take and read, set arranged administrators, for example, examine and expending check, and a novel meeting administrator, rhonda.

- Persistent Data Repository: TSpaces utilizes a genuine information the executives layer, with capacities like overwhelming weight social database frameworks, to deal with its information. T Space tasks are acted in a value-based setting, which guarantees the honesty of the information.

- Database Indexing and Query Capability: The TSpaces information administrator files every single labeled datum for exceptionally productive recovery. The extended question capacity furnishes applications with the apparatuses to test the information with point by point inquiries, while as yet keeping up a basic, simple to-utilize interface.

- Progressively Modifiable Behavior: notwithstanding the extended arrangement of implicit administrators, TSpaces permits new administrators to be characterized powerfully. Applications can characterize new information types and new administrators that are downloaded into the TSpaces server and utilized right away. This is rather than social database frameworks that have constrained information type. Backing and restricted

powerfully characterized work (for the most part as triggers or, to a lesser degree, client characterized capacities).

- Event Notification: Applications can enlist to be told of occasions as they occur in the TSpaces server.
- Access Controls: Users can set up security arrangements by setting client and gathering authorizations on a Tuplespace and administrator premise.
- Large Object Support: Applications can indicate URLs as field esteems. The URL properties can be set as "duplicate prompt" or "remote reference". "Duplicate Immediate" suggests that the substance of the URL are replicated between the customer and server when the tuple is moved. "Remote Reference" leaves the URL where it is. In any case, another administrator, "getURL" brings the URL substance for neighborhood handling.

The TSpaces framework is proper for any application that has appropriation or information stockpiling prerequisites. It can perform huge numbers of the obligations of a social database framework without forcing an excessively prohibitive (and crude) type framework, an inflexible pattern, an awkward UI or an extreme runtime memory necessity. It might be said, it is a database framework for the normal ordinary figuring gadget - one that doesn't produce complex SQL questions, however one that needs solid stockpiling that is organize open.

3.1. The TSpaces Model

The model is basic. There are customers and there are servers Clients peruse and compose information from to a server with straightforward strategy calls, similar to this [8]:

```
Tuplespace ts = new Tuplespace(spaceName, serverName);  ts.write  ("mydata1", dataInstance);
```

That is it. Simple. No table definitions, no awkward SQL embed explanations. No decaying your intricate information type into SQL crude ints, buoys And character types. At that point, on the off chance that you need to peruse this information from the equivalent or an alternate application, you simply read it :

```
resultTuple = ts.read("mydata1", DataInstance); resultTupleSet = ts.scan(String, DataInstance);
```

Albeit a customer can converse with a few servers at the same time, in the present discharge, servers don't share data (for example it doesn't as of now support storing or replication across servers and servers don't coordinate on any single exchange).

3.2.The TSpaces Server

TSpaces servers can be run wherever [8]. We think that its advantageous to run them locally (to organize a couple of office machines) just as run them in division servers (for more extensive territory administrations). For instance, it have been assigned a default "TSpaces Server" machine that is picked by the customer if the customer doesn't indicate a server. Thus, much the same as "Name Server" and "Print Server" are notable nom de plumes, "TSServer" has been added to the rundown right now. Be that as it may, there is no torment engaged with running your own server, since they require fundamentally zero organization. For instance, one of clients, Joe, utilizes six unique machines. Joe assigns one machine to be the nearby server,

and different machines simply allude to that machine for the TSpaces administration. It have been normal that as applications give the idea that stay inhabitant on clients' work areas, TSpaces servers will get typical as foundation forms.

3.3. A TSpaces Client

A TSpaces customer program is only any old program that makes calls to the TSpaces server. When the server is running [9]. The customer application program just runs. Along these lines, for instance, in case you're running the Blue Clipboard application, you simply fire it up and it runs.

3.4.Running a TSpaces Server

So as to really run your TSpaces customer [9]. You have to have an accessible TSpaces server. During the testing stage, you will likely run your own duplicate of the server on your nearby machine and allude to it with the server name of "localhost". The example programs accompany a shell content for beginning a TSpaces server and you might have the option to utilize that content to begin your server. In the event that you compose your own content or conjure it straightforwardly, you need to guarantee that the CLASSPATH is set effectively with the goal that the container documents that contain the Server code is accessible. We should expect this is a WinNT framework and you introduced TSpaces in the c:\java catalog. The accompanying ought to be adequate to run the TSpaces server.

set

CLASSPATH=c:\java\tspaces\classes;c:\java\tspaces\lib\tspaces.jar

Java com.ibm.tspaces.server.TSServer [options]

There are various choices that can be determined when beginning the server.

Options:

[-a password]	Secret phrase for sysadmin userid
[-b]	Boot without restoring TupleSpaces
[-B]	Boot without restoring TupleSpaces or User/Group status
[-c ConfigurationFile]	specify location of the Configuration file
[-d checkpointDirectory]	specify a checkpoint directory
[-p port#]	specify a port number [default 8200]
[-i interval]	specify a checkpoint interval [-D] Turn on Debug yield
[-A]	Allow Admin actions via http
[-S]	Start the HTTP debug interface

That the spaces after the choice banners are required. The entirety of the alternatives can be indicated by the Configuration document. In the event that the - c alternative isn't indicated, it will search for a "tspaces.cfg" in the ebb and flow index in the event that still not discovered, at

that point it will run with a lot of hard-coded defaults. An example tspaces.cfg document is disseminated with the framework that contains remarks about the choices.

4. Starting a Local Server

In some cases it is helpful to have the customer really run the server as a string inside that equivalent Java Virtual Machine as the customer [10]. It lets the customer control the beginning and halting of the customers and may give critical execution improvements. For instance, it may be helpful when you have numerous customers on remote frameworks giving contribution to a primary focal server where they are prepared by a neighborhood customer. Right now need the neighborhood customer to have the option to get to TSpaces as quick as would be prudent.

TSServer ts = new TSServer();

Thread serverThread = new Thread(ts, "TSServer"); serverThread.start();

The above TSServer default constructor will let the entirety of the order line choices have the default esteems, including having the default tspaces.cfg record in the present registry. You can allude to the JavaDoc API for TSServer to perceive how you would determine different alternatives for the server. Presently to utilize the new superior interface that evades all TCPIP attachment overhead, you add the accompanying line to your customer code:

TupleSpace.setTSCmdImpl("com.ibm.tspaces.TSCmdLocalImpl")

The remainder of your code is unaltered. It ought to be noticed that solitary this customer would utilize this uncommon interface. Some other customers on another framework or even on this framework yet under an alternate JVM will keep on utilizing the attachment interface to speak with this server.

5. Main Window Screen

This window contains one menu at appendix –B– which its content is:

1- Create tuple space button which the user can obtain from pressing on it another screen that represent the way by which the user can create tuple space with it’s almost options.

2- deal with tuple space button which we can obtain from pressing on it another screen by using this screen the user can get information about the spaces that he create and also deal with these spaces.

3- Tuple operation button, the user can obtain by pressing on this button a screen provides the most tuple operations that the user may need

5.1.Create Tuplespace Screen

This screen provide an easy way to create tuple space by entering host name, space name and choose the options which make the tuple space more efficient then pressing on create button to make the server create the space, in case that the user choose access control list the user must enter owner name, password and user (according to the owner). Also he must give the permission for both user and anonymous for write, read, take operations look at figure .2 at appendix -B- to see the screen, the options that is provided are:

- **FIFO**, which makes the information that, has been stored in the user space returned in way of first in first out.
- **Persistence** option if this option have been chosen then the configuration of the space of that option will set false this cause the space not be saved across server restart.
- **Local host** option this option mean that the server is on the same computer of the user and this will facilitate the work for both client and server by using facility of:

```
TupleSpace.setTSCmdImpl("com.ibm.tspaces.TSCmdLocalImpl");
```

That has been discussed previously in chapter three. The last option is **ACL** (access control list) by chosen this option the protection will be given the space.

5.2.Deal with tuplespace Screen

This screen provide an easy way to deal with the desired space by entering host name, space name and press connect button pressing on this button the program will check if the space exist and having ACL or not after pressing on it a dialog box asking for user name and password if the space has protection on it, but if he does not has then this dialog box will never show, then choose the options provided by this screen which are: First option is exist option by choosing this option the server will check if the space exist or not and show a massage explain that. Second option is status, which returns the status of the server if it is running or not if it is then the program show massage containing the port and the host of the server.Third option is deletall which delete all the content of space, if the space has protection on it then the user has to enter the user name and the password, if he does not has it then the delete operation will be interrupted by the program ,but if he has it then the program will show a dialog box that warning from deleting the contents this space because it will make the space lose its protection(ACL),if the space does not has protection then the program will delete all the contents of the space. Fourth option is version which will show the version of the package. Last option is destroy that will destroy the space, if the space has no protection then the program will destroy it else the program will ask for user name password if the user has it then the program will show a dialog box check if the user really want to destroy the space, if his answer is yes then the space will be destroyed, if his answer is no the space will remain, if the user does not has the user name and password the program will interrupt the destroy operation.

5.3.Tuple Operation Screen

This screen will provide 13 buttons each button represent atuple operation except **connect** button, as previous screen the user must enter host name and space name then press connect button by pressing on this button the program will check if the space exist and having acl or not if the space does not exist and he has to enter another name ACL then the program will show a small screen asking for owner name and password after entering it the program will check if they were correct if they were correct if they are then can perform all the operations provided by this screen else the user will be consider as anonymous then the operations which the anonymous The first button in this screen is write button which pressing on it will show another screen which is write screen this screen provide an easy way to write a tuple to the space this done by entering the number of field that the tuple will have, then press create button, next step is determining the type of field and its content if the user want to show ID number of the tuple then he has to choose show tuple ID number choice, last step is pressing

write button this will make the program write the tuple to the space, The second button is read button which pressing on it will show the read screen, to read a tuple the user has to choose the type of query which are index query (read tuple by only mentioning one field in the required tuple), content query (read tuple by mentioning all the type and contents of the field of the required tuple), type query (read tuple by mentioning only the types of fields of the required tuple). After choosing type of query the user has to enter the name of indexed field and its content if he chooses first choice, but he has to enter the types and contents of the tuple if he choose content query, if he chooses type query then he has to enter types of the fields of the tuple after this step the user has to press on read button or waittoread button until the tuple written to the space. The third button is scan button which pressing on it will show scan screen, to read a group of tuple the user has to do the same steps in reading one tuple except the user has to press scan button instead of red button or consuming scan button which will read and delete the tuples The fourth button is begin trans button which pressing on it will add the space of the user to transaction, how to do this and what the benefit of doing this is mentioned previously in chapter three.The fifth button is end trans which pressing on it will execute commit trans operation, which mentioned in the previous three and what it does.The sixth button is consuming scan which pressing on it will show the screen of scan which been discussed previously. The seventh button which pressing on it will show the take screen, to take (read and remove) a tuple from the space the user has to do the same steps that mentioned in reading tuple except that the user has to press take button instead of read button The eighth button is multi write button which pressing on it will show multi write screen, to write a group of tuples the user has to enter the number of tuples the user want to write then press create array button, then he has to followed the same steps mentioned previously for write one tuple to write each tuple to the group and pressing add tuple every time creating one tuple to add this tuple to the group, but to write all the tuples to the space the user has to press write array button instead of write button and the user can choose show tuples array ID number choice if he want that. The ninth button is count tuple button which pressing on it will show count tuple screen, to count the number of required tuple in the space the user has to follow the same steps in reading tuple and press count tuple button instead of pressing read button .The tenth button is delete button which pressing on it will show the same screen for count tuple the user has to follow the same steps in counting tuple except pressing delete button instead of count button .The eleventh button is update button which pressing on it will show update screen, to update a tuple the user has to do the same step in reading one tuple and create the field to be add to the tuple or to be replaced by a field which already exist in the tuple after that the user should enter position of the field in the tuple then press update button if he want to replace the field which have been entered its position or press add button if he want to add the field in the desired position in the tuple The last button is close space button which pressing on it will execute clean operation, which will close the space from doing any operation on it.Note: if there is exit button in any screen pressing on this button will make the user exit from this screen.If any operation that have been discussed need showing results these results will be shown in text area which its position is the right top of the screen.

6. Conclusions

In this project the lime system have been presented by using GUI (Graphical User Interface), this system adapts the Linda model of transiently shared tuple spaces, tuple location, and reactive statement. The point of this work making graphical UI for Lime (Linda in a Mobile Environment) package in order to facilitate this package to be use by the user. In order to explore the difficulties that may the user face with this package. It has been searched to furnish application engineers with instruments that encourage fast improvement of such applications through cautious choice of a little arrangement of interchanges natives and builds.

References

1. D. Garlan and D. Le Metayer, editors. Proc. of the 2nd Int. Conf. on Coordination Models and Languages (COORDINATION '97), volume 1282 ofLNCS. Springer, Sept. **1997**.
2. Pope, A. **1998**. The Corba Reference Guide: Understanding the Common Object Request Broker Architecture. Addison-Wesley.
3. C. Perkins. IP mobility support. RFC 2002, IETF Network Working Group, **1996**.
4. D.B. Johnson. Routing in Ad Hoc Networks of Mobile Hosts. In Proc. of the Workshop on Mobile Computing Systems and Applications, pages 158, 163, **1994**.
5. A. Fuggetta, G. P. Picco, and G. Vigna. Understanding Code Mobility. IEEE Trans. on Software Engineering, May **1998**.
6. Licai Capra and Wolfgang Emmerich and Ceclia Mascolo Department of Computer Sceince University College London **2003.**
7. Rashid, B.; Rehmani, M. H. Applications of wireless sensor networks for urban areas: A survey. Journal of network and computer applications.**2016**, 60, 192-219.
8. Lewandowski, S. M. Frameworks for component-based client/server computing. ACM Computing Surveys (CSUR).**1998**, 30, 1, 3-27.
9. Hazzard, E. Openlayers 2.10 beginner's guide. Packt Publishing Ltd **2011**.
10. Cardinal, R. N., & Aitken, M. R. Whisker: a client—server high-performance multimedia research control system. Behavior research methods.**2010**, 42, 4, 1059-1071.