# VINPAN
# VOTING IN PERSONAL AREA NETWORK

THANABORDEE THANARUKVUDHIKORN
THANAWAN SABAIJAI
OHM SAMKOSES

A SPECIAL PROJECT SUBMITTED IN PARTIAL FULFILLMENT
OF THE
REQUIREMENTS FOR THE DEGREE OF BACHELOR OF SCIENCE
INTERNATIONAL PROGRAMS, FACULTY OF SCIENCE
KING MONGKUT'S INSTITUTE OF TECHNOLOGY
LADKRABANG
2007

# VinPAN
# Voting in Personal Area Network

Mr.Thanabordee Thanarukvudhikorn
Ms. Thanawan Sabaijai
Mr.Ohm Samkoses

A Special Project Submitted in Partial Fulfillment of the Requirements
for
The Degree of Bachelor of Science
International Programs, Faculty of Science
King Mongkut's Institute of Technology Ladkrabang
2007

**Title of special project** VinPan(Voting in Personal Area Network)

**Student**
Thanabordee Thanarukvudhikorn 47050361
Thannawan Sabaijai 47050362
Ohm Samkoses 47050368

**Degree** Bachelor's Degree of Science

**Department** Mathematics and Computer Science, Faculty of Science

**Program** Computer Science (International)

**Advisor** Teera Fagon

# ABSTRACT

We propose an electronic polling system on the mobile phone communicates by Bluetooth technology which is reliable and can be used in real life. The implementation was done by using Java (J2SE and J2ME) and the experiments were done in both ways, on the built-in emulator of Sun Java(TM) wireless toolkit and in real mobile phones. Our project focused on using in the conference room which the voters are sitting at fix locations, the distance between them are not too far, and they do not concern about the speed of the polling process.

# Acknowledgement

The author would like to take this opportunity to express our thanks to teachers and persons who have given useful advice and support in this research.

The author wishes to express our deep gratitude to Mr. Teera Fagon our advisor, for his guidance, attention, and encouragement throughout this research.

We are greatly appreciated all the professors who have given invaluable knowledge while we are studying in the department of computer, Faculty of Science, King Mongkut's Institute of Technology Ladkrabang.

Lastly, the authors would like to express our deepest appreciation to our dearest parents for their love, care, and encouragement.

Thanabordee Thanarukvudhikorn
Thanawan Sabaijai
Ohm Samkoses

# Table of Contents

Page

# List of Tables

# List of Figures

# Chapter 1
# Introduction

## 1.1 Rationale

Nowadays, demands of customers are very important for many industries. The marketing manager needs to know what market wants to gain more income. The opinion of the customers is the key to the better marketing plan and one way to get that valuable information is to gather them by using polls.

The original polling system was paper-based and then evolved to Internet-based. These typical polling systems are insecure and require too many resources. Paper-based polling system wastes too much papers and time, and also difficult to process the result. Internet-based polling system needs too many factors, mainly the internet connection which loses the flexibility and mobility. These reasons are the motivation of VinPAN (Voting in Personal Area Network) system – a mobile phone polling system via Bluetooth™.

Since mobile phone is a very common communication device that almost everyone has and in the next few years the number of people who own one or more mobile phones expected to be up to 90% of total citizen. With this information, we design VinPAN as a polling system on something that almost everyone has and can use anywhere, anytime.

Another technology that is the best for our system is Bluetooth™ technology which can be used to form a Personal Area Network (PAN). Bluetooth™ is a small device which provides a way to connect and exchange information between devices such as mobile phones, laptops, PCs, printers, digital cameras and video game consoles over a secure, globally unlicensed short-ranged radio frequency. It also can operate through obstacles such as walls.

The VinPAN system assumes that most of the people have mobile phone with Bluetooth™ device equipped (almost of the new arrival mobile phone is already included this feature). The service provider (poll provider) creates the poll question by a typical personal computer (PC). The poll provider can use our Question generator or manually create an

XML file. Then install the question to a mobile phone which has our server application installed. We will define that mobile phone as a Server. The question will be send via Bluetooth™ up to the seven nearest mobile phones which are directly connected to the server, forming a Piconet. Those seven phones will send the question to other mobile phones connected to them, which will form a huge network composed of many Piconets. This huge network is a Scatternet. When the reply messages from the clients are sent, they'll be sending back to where they come from until reached the server. Finally, all answers will be analyzed and displayed on the server's screen or may stored in the database in future development.

By implementing the system this way, these are the advantages of the VinPAN system:

- The poll provider can obtain the result with low operating cost because the transmission of data via PAN is a free.
- Can limit the area to collect the poll because Bluetooth™ is short-ranged so only people in area of the server can vote.
- Has high accessibility because Bluetooth™ can communicates though walls and obstacles.

## 1.2 Objectives

1. VinPAN can be used to sends and receives the poll from any mobile phones with VinPAN installed.
2. Encourage developers to develop more Bluetooth™ related applications
3. Increase the flexibility of polling system. People can be used for polling anywhere, anytime.
4. To promote usage of mobile devices.

## 1.3 Scope of Study

Objectives in the section 1.2 have shown the scope of our system. However, there are some further limitations. First, our system supports only the phones which implements MIDP 2.0. Second, the connection speed and security is beyond our scope of study. Third, the system is developed to proof the concept; the polling result will not be stored.

## 1.4 Research Methodology

1. Study how to implement the program by using Java Micro Edition (Java ME)
2. Implement Question Generator program by using Java Standard Edition (Java SE)
3. Implement and Test VinPAN by using Java ME and Java Wireless Toolkit's emulator and real mobile phones
4. Process and analyze test result

# Chapter 2

# Related Literatures

## 1. Bluetooth

Bluetooth is an industrial specification for wireless personal area networks (PANs). Bluetooth is a standard and communications protocol primarily designed for low power consumption, with a short range based on low-cost transceiver microchips in each device. Bluetooth uses radio waves (in the 2.4 Gigahertz range), and is designed to be a secure and inexpensive way of connecting and exchanging information between devices without wires.

Bluetooth devices can function in two modes:

- Circuit switched (the most common mode for voice communications, on land and wireless digital networks), and
- Packet switched (the mode for Internet data, as well as for higher bandwidth mobile communication systems on the horizon, like GPRS [General Packet Radio Service]).

Bluetooth can choose to act as master or slave. Bluetooth has network system call piconet. Piconet is an ad-hoc computer network of devices using Bluetooth technology protocols to allow one master device to interconnect with up to seven active slave devices (Figure 2.1).

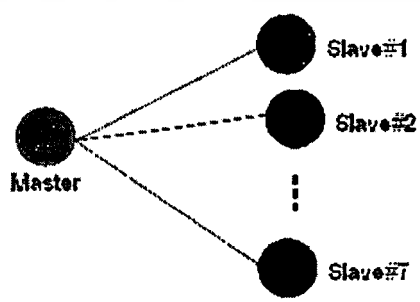


Figure 2.1 connection of piconet

Each of the active slaves has an assigned 3-bit Active Member address (AM_ADDR). There can be additional slaves which remain synchronized to the master, but do not have a Active Member address. These slaves are not active and are referred to as parked. For the case of both active and parked units, all channel access is regulated by the master. A parked device has an 8-bit Parked Member Address (PM_ADDR), thus limiting the number of parked members to 256. A parked device remains synchronized to the master clock and can very quickly become active and begin communicating in the piconet. Scatternet is a type of ad-hoc computer network consisting of two or more piconets (Figure 2.2).

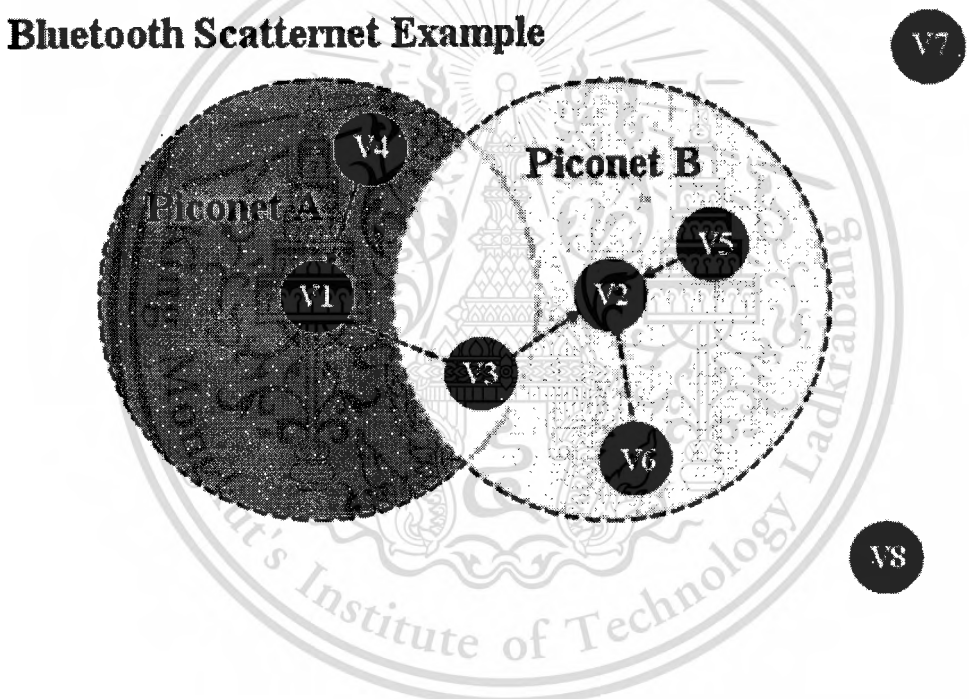


Figure 2.2: connection between piconet and piconet (scatternet)

Piconet A consists of master V1, slaves V3 and V4. Piconet B consists of master V2 and slaves V3, V5, and V6. Vehicles V7 and V8 are out of range and cannot communicate. Vehicle V3 can be used as a router between the piconets, although this is not a built in feature of the Bluetooth specification.

Bluetooth can be used to create an ad hoc network between mobile nodes with no central administration. This is accomplished by the following steps:

1. All devices scan for inquiries.
2. One device (the Master) initiates an inquiry.
3. Devices in range of the Master reply to the Master's inquiry. These devices become Slaves.
4. Up to 7 Slave devices can connect to the Master to form a piconet.
5. Any Slave device can request to become the Master of a piconet at any time.

The distinction between Master and Slave is only to allow easier synchronization over the frequency hopping spread spectrum (FHSS) communications link. All Slaves synchronize to the Master's clock and the Master sets the frequency hopping sequence.

# Bluetooth network forming Algorithm

```
CASE 1.  TWO ISOLATED NODES DISCOVER EACH OTHER
      Call PFMR
CASE 2.  A MASTER DISCOVERS AN ISOLATED NODE
      Case 2.a No. of active slaves in the existing piconet < 7
            Call PFMR
      Case 2.b No. of active slaves in the existing piconet = 7
            Determine if there is any existing non-bridge slave of the piconet that
            is within radio range of the isolated node
            If there is no such node
                  Send the least busy slave to the Park mode and include the new node
                  as an active slave
             Else
                  Call SFMR to make the isolated node as the master of a new piconet,
                  transfer up to 4 non-bridge slaves to this piconet that are within
                  its radio range, making the slave with the highest capacity as the
                  bridge node
CASE 3.  A SLAVE DISCOVERS AN ISOLATED NODE
      Case 3.a No. of active slaves in the existing piconet < 7
            If master of the existing piconet is within radio range of the isolated
            node
                  Call PFMR to include the isolated node in the existing piconet
            Else
                  Determine if there is any existing non-bridge slave of the piconet
                  that is within radio range of the isolated node
                    /* There is at least one such node, which is the slave that
                    discovered the isolated node */
                  Call SFMR to make the isolated node as the master of a new piconet
                  Transfer up to 4 non-bridge slaves that are within its radio range,
                  making one of them as the bridge node
      Case 3.b No. of active slaves in the existing piconet = 7
            Similar to the Else condition of Case 3.a above
CASE 4.  A SLAVE (S1 of piconet P1) DISCOVERS A MASTER (M2 of piconet P2) AND
            VICE VERSA
      Case 4.a Total No. of active slaves in P1 and P2 = 14
            If there is no bridge node between P1 and P2
                  Send the least busy slave of P2 to the Park mode
                  Make S1 as a bridge between P1 and P2
      Case 4.b Total No. of active nodes (master + slave) in P1 and P2 > 8 and < 16
            If No. of active slaves in P2 < 7
                  If there is no bridge between P1 and P2
                        Make S1 as a bridge between P1 and P2
                        Transfer up to 4 non-bridge slaves from the larger piconet
                        to the smaller piconet that are within radio range of the
                        master of the smaller piconet
                          /* The last step will hereinafter be referred to as
                          "Perform Load Balancing" */
                  Else
                        Perform load balancing
            Else
                  If there is a bridge between P1 and P2
                        Perform Load Balancing
                  Else
                        Determine if there is any non-bridge slave of P2 that is
                        within radio range of M1
                        If at least one such non-bridge slave exists
                                Make that slave of P2 as a bridge between P1 and P2
                                Perform load balancing
                        Else
                                Send the least busy slave of P2 to the Park mode
                                Make S1 as a bridge between P1 and P2
```

```
      Case 4.c Total No. of active nodes (master + slave) in P1 and P2 <= 8
            If all the nodes of P1 are within radio range of M2 and all the nodes
            of P2 are within radio range of M1
                 Merge P1 and P2 by calling PFMR and make either M1 or M2 as the new
                 master of the combined piconet depending on whichever has higher
                 power
                 If there was a bridge between P1 and P2, make it a non-bridge slave
            If all the nodes of P1 are within radio range of M2
                 Merge P1 and P2 by calling PFMR and make M2 as the new master for
                 all the nodes of P1 and P2 combined
                 If there was a bridge between P1 and P2, make it a non-bridge slave
            If all the nodes of P2 are within radio range of M1
                 Same as above with roles of M1 and M2 reversed
            If some of the nodes of P1 are not within radio range of M2 and some of
            the nodes of P2 are not within radio range of M1
                 If there is no bridge between P1 and P2
                      Make S1 a bridge between P1 and P2
                      Perform Load Balancing
                 Else
                      Perform Load Balancing
CASE 5.  A SLAVE (S1 of a piconet P1) DISCOVERS A SLAVE (S2 of a piconet P2)
      Case 5.a Total No. of active slaves in P1 and P2 = 14
            Similar to Case 4a above
      Case 5.b Total No. of active nodes (master + slave) in P1 and P2 > 8 and < 16
            If there is a bridge between P1 and P2
                 Perform Load Balancing
            Else
                 If there is at least one non-bridge slave in P1 within radio range
                 of M2 or one non-bridge slave in P2 within radio range of M1
                           Make such a slave a bridge between P1 and P2
                           Perform Load Balancing
                 Else
                           Interchange roles of S1 and M1 or S2 and M2 depending on
                           whether P1 is smaller than P2 or not
                           Follow the fourth outer if condition of Case 4c above
      Case 5.c Total No. of active nodes (master + slave) in P1 and P2 <= 8
            The first three outer if conditions similar to Case 4c above
            If some of the nodes of P1 are not within radio range of M2 and some of
            the nodes of P2 are not within radio range of M1
                  Similar to Case 5b above
CASE 6.  A MASTER (M1 of a piconet P1) DISCOVERS A MASTER (M2 of a piconet P2)
      Case 6.a Total No. of active slaves in P1 and P2 = 14
            Similar to Case 5a above
      Case 6.b Total no. of active nodes (master + slave) in P1 and
            P2 > 8 and < 16
            Similar to Case 5b above
      Case 6.c Total No. of active nodes (master + slave) in P1 and P2 <= 8
            Similar to Case 5c above
```

Figure 2.3: Bluetooth Network Forming Algorithm

Bluetooth's protocol uses all possible scenarios when two nodes discover each other. The different scenarios have been shown as different cases in Figure 2.3.

The figure refers to two routines, namely,
- Piconet Formation and Modification Routine (PFMR) in Figure 2.4
- Scatternet Formation and Modification Routine (SFMR) in Figure 2.5

Respectively, depending on the roles of the nodes that discover each other and the configurations of the piconets (if any) to which they belong.

A newly started node takes the role I and executes an Isolated Node Protocol (INP). It enters the Inquiry or the Inquiry-scan state with equal probability. The time it spends in either of the states is chosen randomly between 0.3125 and 1.35 sec. These two limits were

determined through extensive simulation. Switching from one state to the other is repeated till it latches with another node. While running INP, the new node can be latched either to another isolated node or to a master/slave of an existing piconet performing node discovery. If the second node is also isolated, the two together form a piconet of just two nodes using the Piconet Formation and Modification Routine of Fig. 2.4, with one taking up the role of master and the other that of slave. The node with higher capability in terms of memory and available power is given the role of master as shown in the figure. PFMR is also called if a master discovers an isolated node and the node can be included in the existing piconet. However, if the piconet already has seven slaves and no nonbridge slave is in the radio range of the new node, the master sends the least busy slave into Park mode and includes the new node in its piconet. If there are more than seven slaves in a piconet, the master keeps on sending an active slave into Park mode and Unparking an already parked slave following either a least priority policy or any other policy as governed by the master, ensuring fairness among the slaves. Apart from listening to beacon train or broadcast messages sent by a master, a parked slave can also spend the rest of the time in device discovery, acting like an isolated node. If a master does not receive any response from a parked slave within the link supervision timeout period set for that slave, it considers the slave to be disconnected or moved to another piconet. If there is at least one nonbridge slave of an existing piconet that is within radio range of the new node, a new piconet is created with the isolated node as its master. One such nonbridge slave within radio range of both the original master and the new master is considered the bridge node connecting the two piconets.

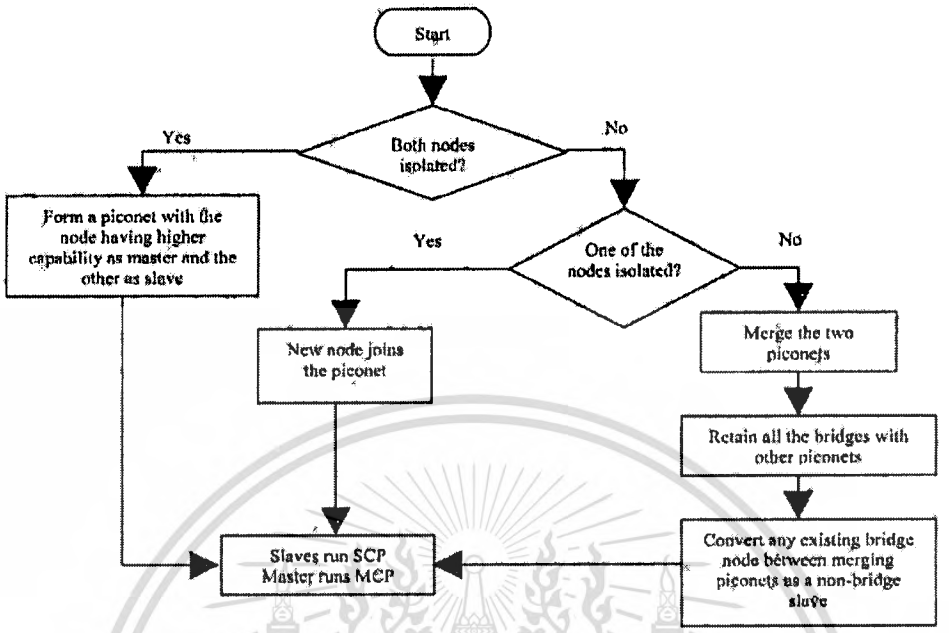


Figure 2.4: Piconet Formation and Modification Routine(PFMR)

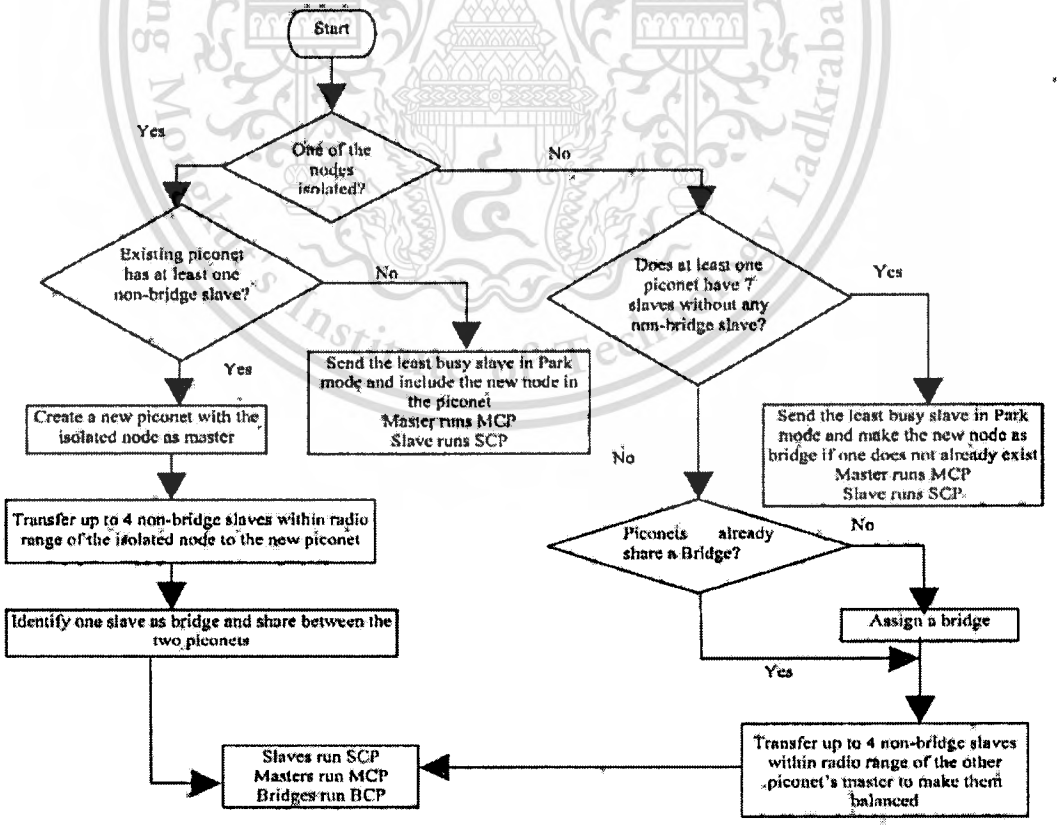


Figure 2.5: Scatternet Formation and Modification Routine(SFMR)

The Scatternet Formation and Modification Routine of Figure 2.5 are used to handle this condition. If there is more than one nonbridge slave of the original piconet that are within radio range of the new master, local load balancing is performed by transferring up to four such slaves for improving the scatternet configuration. If a slave discovers a node, it forms a temporary piconet with the discovered node for information exchange. We use a special Link Manager Protocol PDU (Protocol Data Unit) in order to facilitate such communication and information exchange in the temporary piconet. Details of such a PDU are given in the next section. While choosing a slave for device discovery, a master prefers the one with the least traffic load. In order to cover a large geographical area surrounding its piconet, the master can also choose a slave at random. Such a node is polled at the start of a polling cycle if its calculated priority is greater than the average priority of all the slaves. Otherwise, the master polls it at the end of the polling cycle. Thus, the master ensures that the designated slave has the maximum time for device discovery. When a slave discovers an isolated node and the number of existing slaves in the piconet is less than seven, the isolated node can be included in the existing piconet by calling PFMR if it is within radio range of the master. However, if the isolated node is not within radio range of the master or the number of existing slaves is seven, we create a new piconet having the isolated node as master and one of the nonbridge slaves of the existing piconet as the bridge. Since the slave under consideration had discovered the isolated node, existence of at least one slave within radio range of the isolated node (master of the new piconet) is guaranteed. SFMR is used to handle this situation. Local load balancing is also applied, if possible. Instead of discovering an isolated node, it is possible that, during node discovery, the slave discovers or is discovered by an existing master or a slave of another piconet. In these situations, if the total number of slaves in the two piconets discovering each other is 14 and no bridge exists between these two piconets, the master sends its least busy slave into Park mode and makes the discovering or discovered slave a bridge node between these two piconets. It may be argued that one could create three piconets using the two masters and the 14 slaves if a sufficient number of bridges were identified.

However, this leads to high message complexity in scatternet reconfiguration and, hence, we avoid such restructuring. If the total number of slaves in the two piconets is less than 14, restructuring is done as required by calling either PFMR or SFMR. A master can also go in device discovery mode if the traffic load is low in its piconet or the number of slaves is low (two or three). The slaves stay in power-saving

mode during this phase. Also, a master can send a slave for more than one polling cycle in device discovery phase if the number of slaves is low (two or three). Situations in which a master discovers an isolated node or a slave of another piconet have already been discussed above. However, a master may also discover or be discovered by another master. Such situations are handled in a manner similar to the case where a slave discovers a slave.

The following figures (Figure 2.6 – 2.11) show how to form the network based on Bluetooth network Algorithm from Figure 2.3

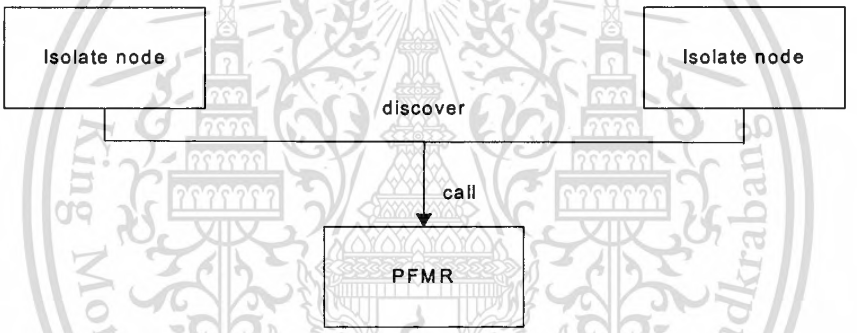


Figure 2.6: Two isolated nodes discover each other (Case 1)

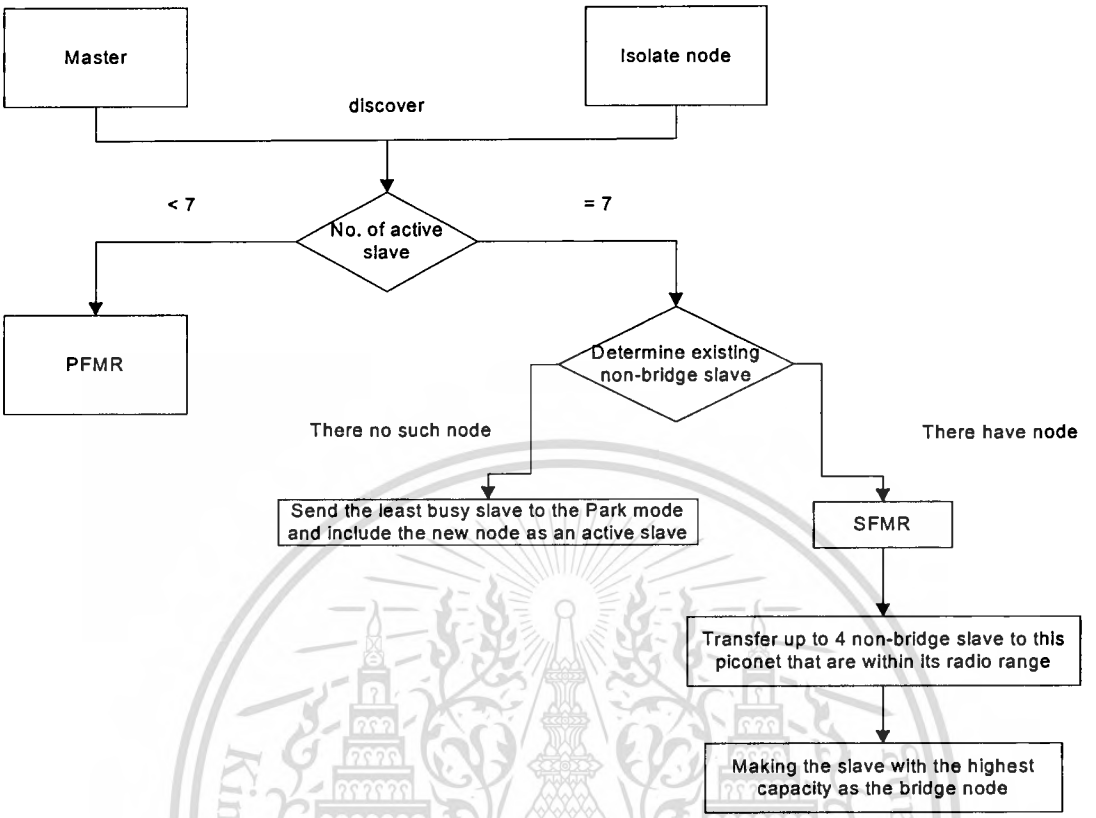


Figure 2.7: A master discovers an isolated node (Case 2)

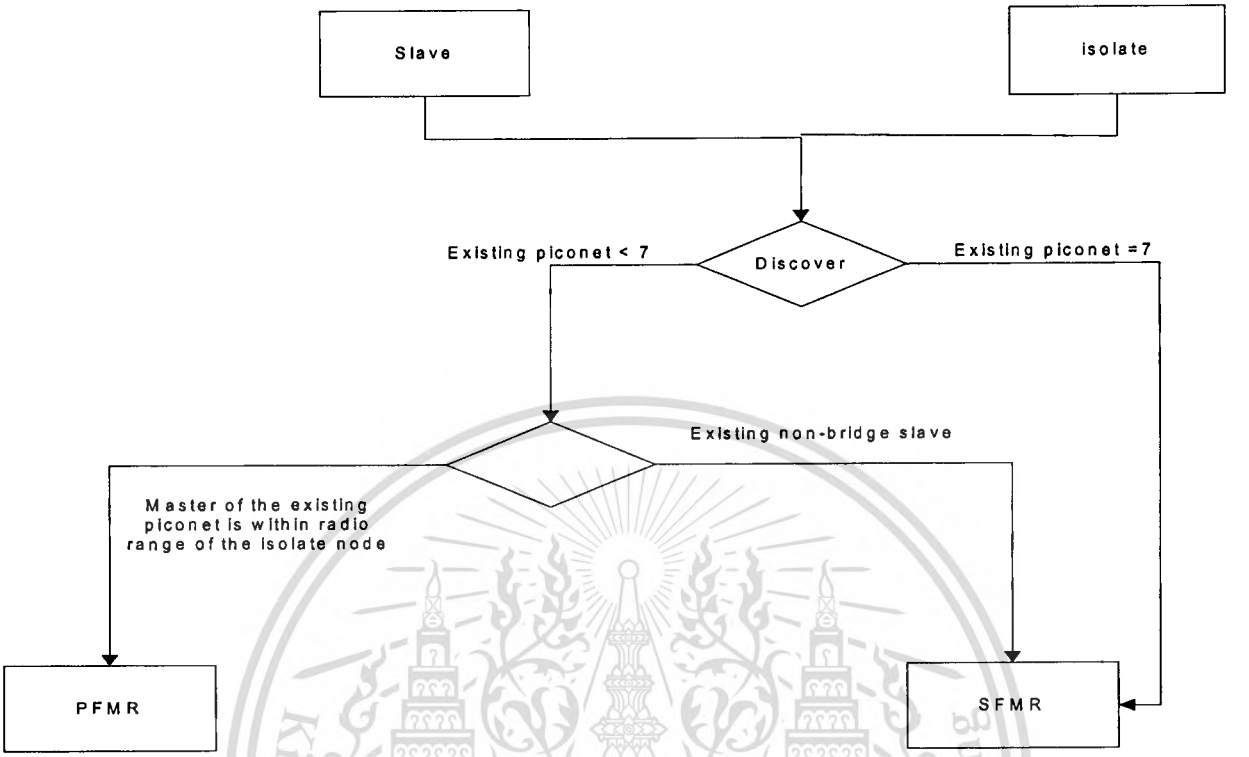


Figure 2.8: A slave discovers an isolated node (case 3)

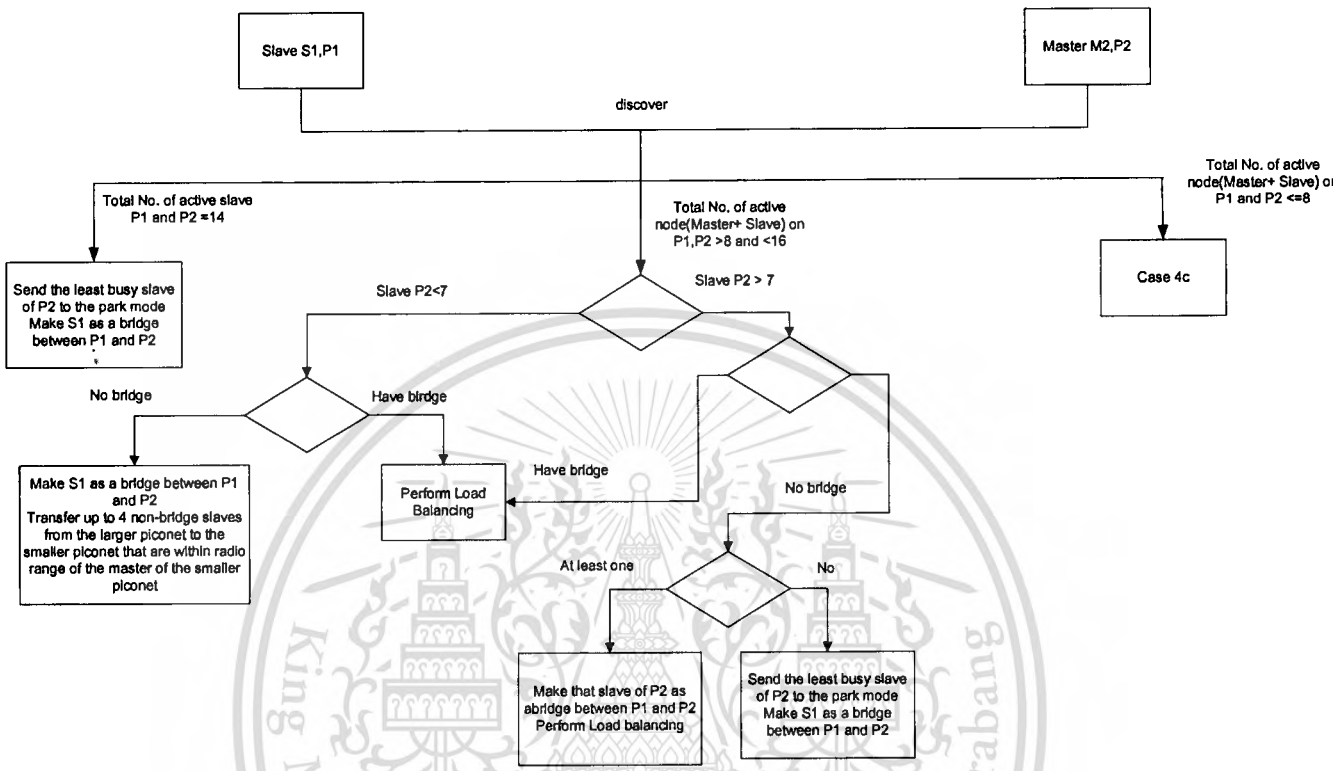


Figure 2.9: A slave (S1 of piconet P1) discovers a master (M2 of piconet P2) and vice verse (case 4)

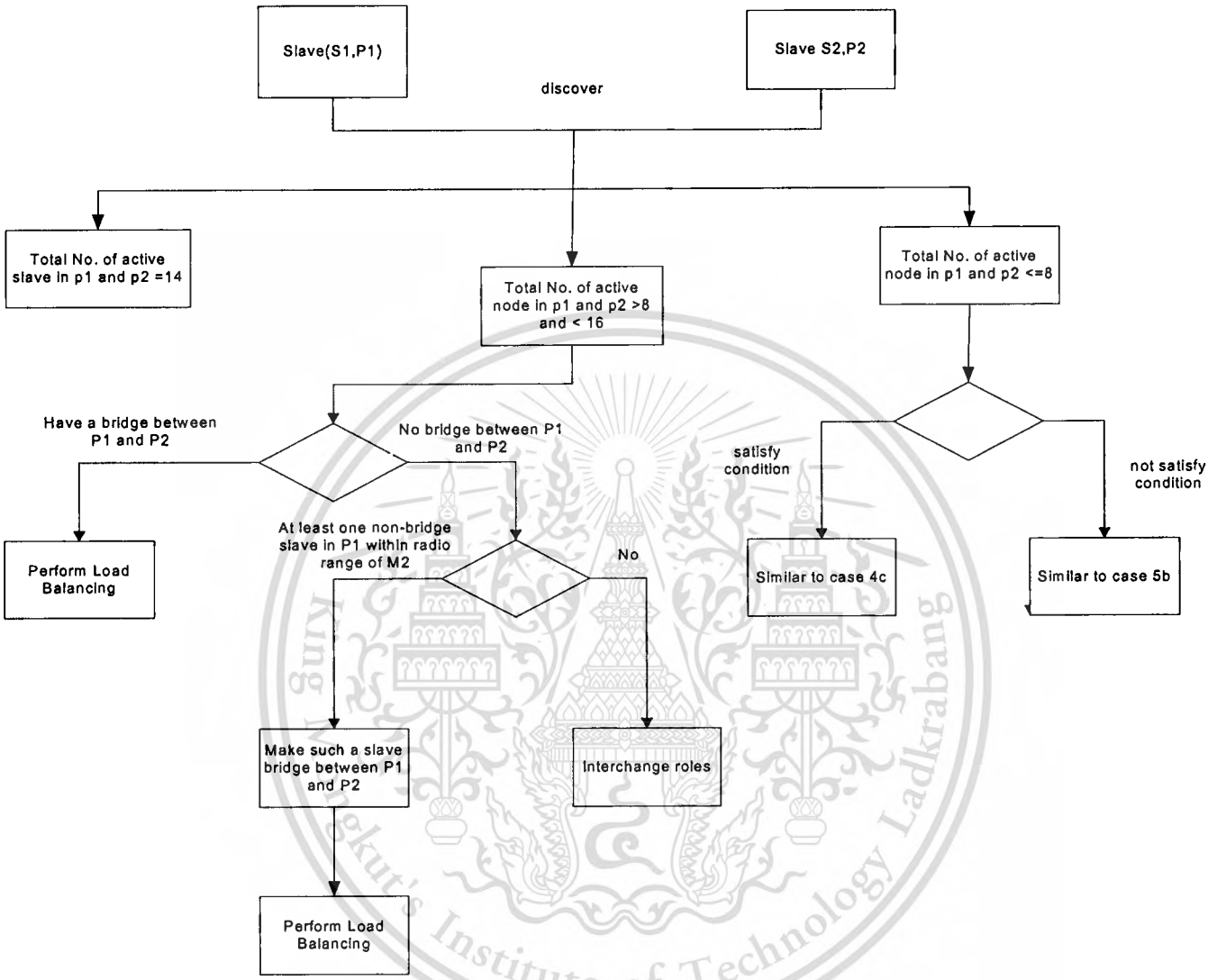


Figure 2.10: A slave (S1 of a piconet P1) discovers a slave (S2 of a piconet P2) (case 5)

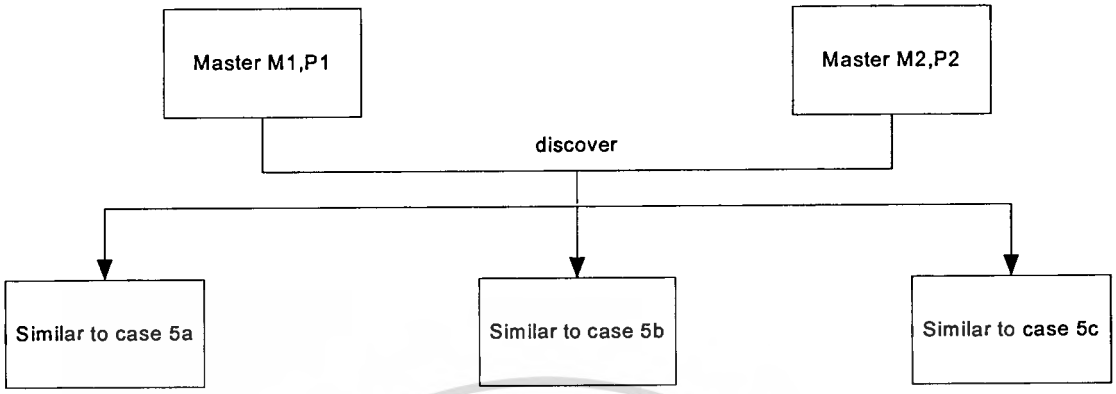


Figure 2.11: A master (M1 of a piconet P1) discovers a master (M2 of a piconet P2) (case 6)

## 2. Java Micro Edition (Java ME)

Java ME (Java Micro Edition) is a technology that allows programmers to use the Java programming language and related tools to develop programs for mobile wireless information devices such as cellular phones and personal digital assistants (PDAs). Java ME consists of programming specifications and a special virtual machine, the K Virtual Machine, which allows a Java ME-encoded program to run in the mobile device. There are two programming specifications: Connected, Limited Device Configuration (CLDC) and the Mobile Information Device Profile (MIDP). CLDC lays out the application program interface (API) and virtual machine features needed to support mobile devices. MIDP adds to the CLDC the user interface, networking, and messaging details needed to interface with mobile devices. MIDP includes the idea of a midlet, a small Java application similar to an applet but one that conforms to CLDC and MIDP and is intended for mobile devices. We use Wireless Toolkit program to test system that we develop. This program has emulator for test before use in real mobile phone. Wireless Toolkit program is a state-of-the-art toolbox for developing wireless applications that are based on Java ME's Connected Limited Device Configuration (CLDC) and Connected Device Configuration (CDC), and designed to run on cell phones, mainstream personal digital assistants, and other small mobile devices.

## 3. Personal area network (PAN)

A personal area network (PAN) is a computer network used for communication among computer devices (including telephones and personal digital assistants) close to one person. The reach of a PAN is typically a few meters. PAN can be used for communication among the personal devices themselves (intrapersonal communication), or for connecting to a higher level network and the Internet

In this project, we use PAN for connection between bluetooth devices. System will build PAN every time for solve the problem of person number of vote.

# Chapter 3
# VinPAN System

## 3.1 System Overview

According to the requirements, the system specifications should be in this following listed:

1. Server can create the question
2. Server can send the question to clients
3. Clients can receive the question
4. Clients can send the answer back to Server
5. Server can collect the answer

According to the system specifications, the system can be separate into 3 sub-systems:

**1. Question Generator**
- Create the question
- Export the question

**2. VinPAN Server**
- Search and connect to clients
- Send the question to those clients
- Display the results

**3. VinPAN Client**
- Search and connect to server
- Search and connect to nearby clients
- Receive question
- Forward the question to those clients
- Send the answers (both it own and clients connected to it)

(All 3 sub-systems roles will be like Figure 3.1)

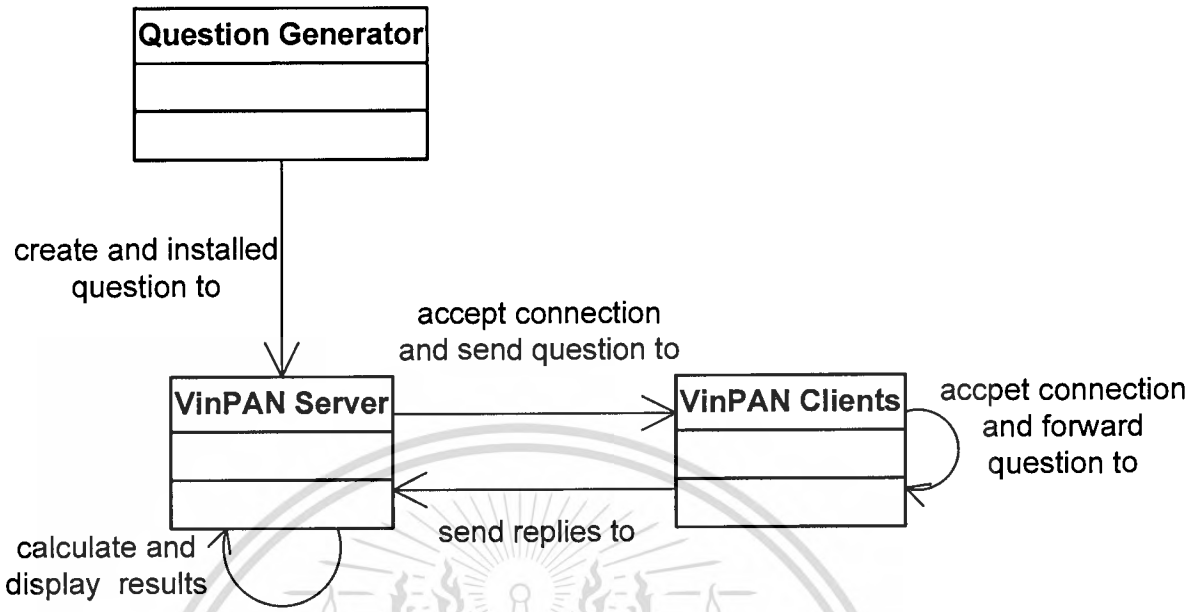


Figure 3.1: Role of the 3 sub-systems of VinPAN

## 3.2 Question Generator

The Question Generator must succeed these factors:

### 3.2.1. The user interface must be easy to use and understand.

The user should know where they should put their data without reading the manual. It must provide a way for user to choose how many choices and what type of question do they want (single or multiple answers). The number of choices is from 2 – 5 choices because, if there are too many choices, it'll not fit the mobile phone's screen which is very small. The textbox should make sense in what kind of data should be filled and can check for errors done by user to reduce the human errors especially; required data missed and input wrong type of data. The user interfaces of Question Generator are represented in figure 3.1 and 3.2.

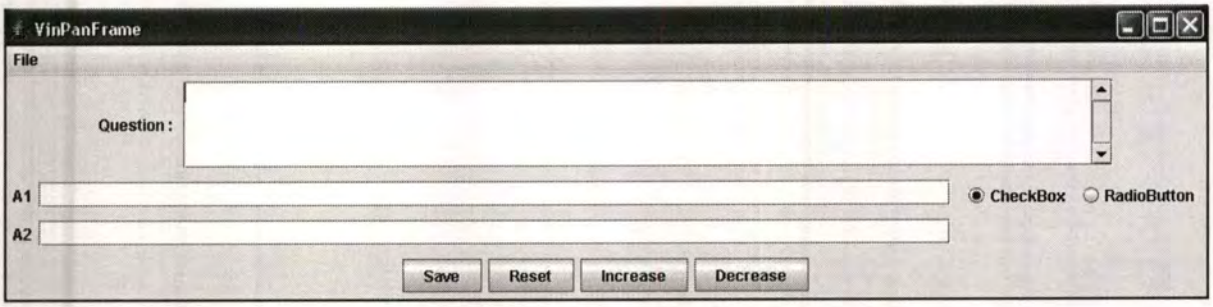


Figure 3.2: Question Generator's default window

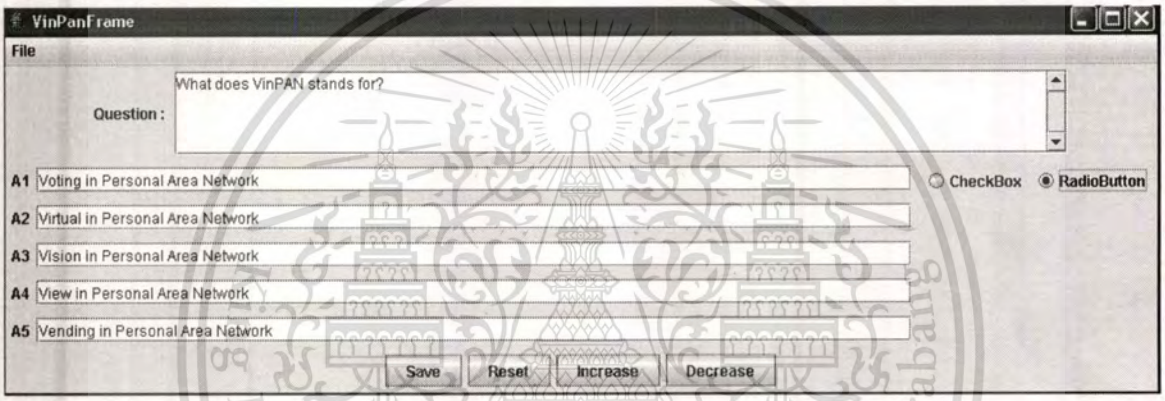


Figure 3.3: Well-input data window

### 3.2.2. The question should generate in XML form.

Since XML is good at describe data and standard, the exported question were designed to use XML instead of normal text file which is non-standard and have nothing to identify the data so, if the user opens the exported file, the data inside is hard to understand.

Each types of the data described by unique XML tags. The tags are represented in table 3.1 and the example of the question in figure 3.4.

Table 3.1: All XML tags in the question file generated by Question Generator.

| XML tags | Description |
|---|---|
| <body> </body> | The main tag of the question consists of many tags and data. It use in transmission of data ( transmit until reached </body> ) |
| <question> </question> | The tag contains the question |
| <choice_number> </choice_number> | The tag defined that how many choices are this question contained |
| <answer n> </answer n> | The tags contain the choices for that question. n is the number of that choice (can be from 2 to 5). |
| <type> </type> | The tag contain the type of the question (single answer, multiple answers) |

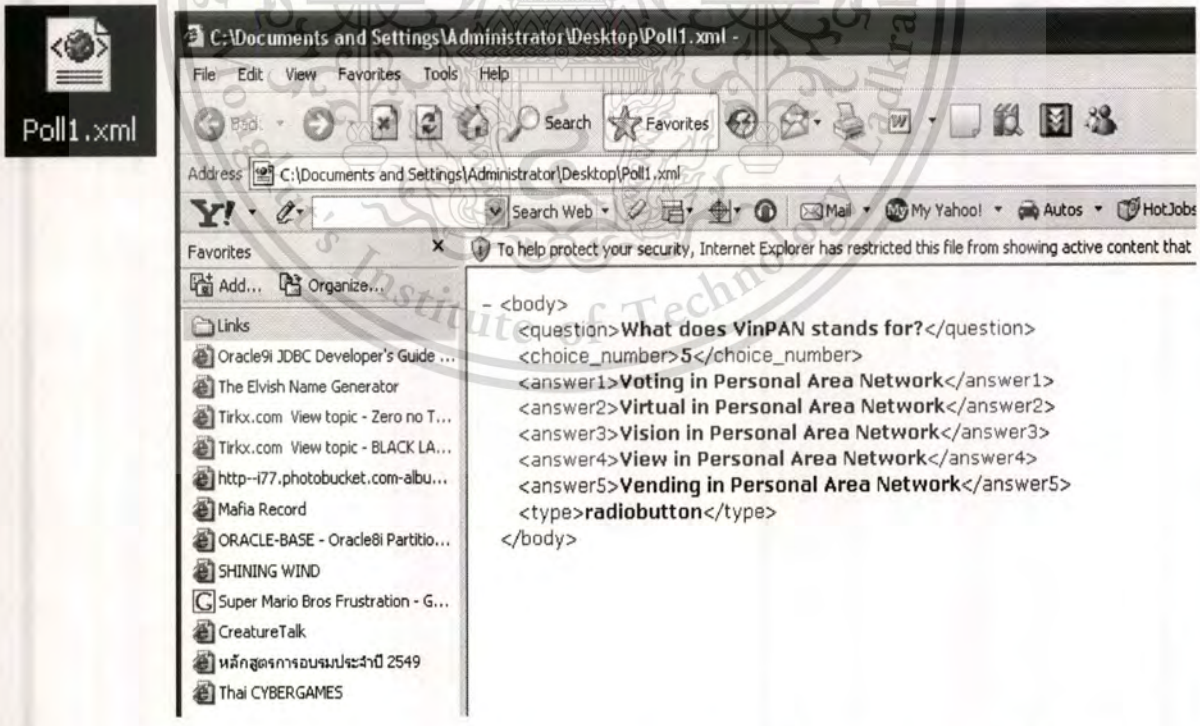


Figure 3.4: export file from Question Generator viewed by Internet Explorer 6.0

## 3.3 VinPANServer

The VinPANServer must succeed these factors:

### 3.3.1. The server must established the connection with clients

The server will start it own service and wait for up to 7 clients to connect. By this way, server may choose whether to accept or decline the clients's connection.

### 3.3.2. The server should embedded security feature to prevent unwanted applications or services to connect to the server

Each of the server and clients are assigned with a UUID which is a set of letters and number. The UUID propose is to distinguish between different applications. The same application should have the same UUID and when the devices try to connect between Bluetooth, it'll first check the UUID. If the UUID of those two devices are not the same, the connection will not establish.

### 3.3.3. The server can read the XML question file and generate the preview of the question

After the XML question has installed on the master phone and the user start the application, the application will read the question and use the XML tags to put the right data at the right place on the phone's screen. User can final check the question before sending to the clients.

### 3.3.4. The server can send the question to all clients connected to it

After all connection established, the server user may press send to start sending the question.

### 3.3.5. The server can display the results from clients in the easy to understand form

Since the purpose of the results from clients is to compare the total result of each choices. Each time the client sends the answer back to server, the results will be calculated in percentage which shows:

- Scores of each choice
- Percentage of scores for each choice
- Total scores of this vote
- Total client participate in this vote (if the question is multi-answer type, one client may answer more than one choice. In this case, total client participate will not equals to total score)

According to the requirements analysis, Class of server can be shown briefly:

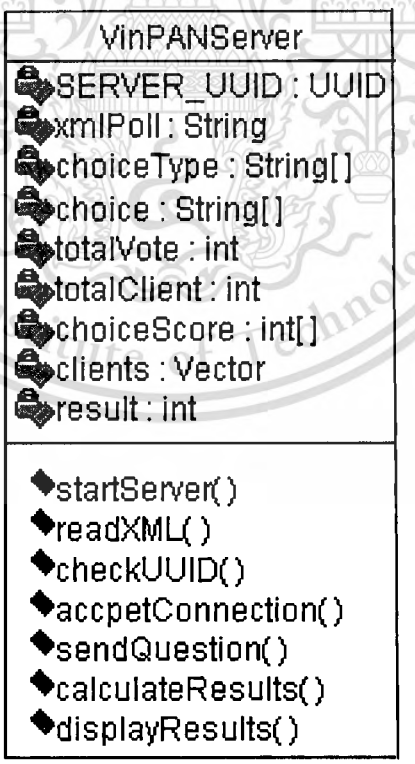


Figure 3.5: VinPANServer Class Diagram

Table 3.2: Description of server class diagram (attributes)

| Attribute Name | Description |
|---|---|
| SERVER_UUID | UUID of VinPAN |
| xmlPoll | Attribute that hold the whole XML question |
| choiceType | Choice type (single-answer, multi-answer) |
| Choice | Contains all choices for the question |
| totalVote | Number of result of answers from clients |
| totalClient | Number of clients participate in vote |
| choiceScore | Contains scores of each choice |
| Clients | Store the name of clients connected with the server |
| Result | Answer from client. Collected the answer one-by-one when received an answer from one of the client |

Table 3.3: Description of server class diagram (methods)

| Method Name | Description |
|---|---|
| startServer | Initialize the server (start service, search for nearby devices) |
| readXML | Read the XML question file and keep the question in attribute xmlPoll |
| checkUUID | Compared UUID between server's and client's |
| acceptConnection | Accept connection from clients that has the same UUID as server's |
| calculateResults | Calculate the results (being called every time an answer has sent from client) |
| displayResults | Display the results on the screen |
| sendQuestion | Send the question to the clients connected to it |

## 3.4 VinPANClient

The VinPANClient must succeed these factors:

### 3.3.1. The client can search and connect to the server that has the highest Quality of Service (QoS)

The client will search for nearby devices and connect to the server that has the highest quality of service for the best performance in transmission. A client may connect to only one server.

### 3.3.2. The client must received the XML question from server and display the correct data in the correct form

After the server had perform the transmission of question, the user interface filled with question will appeared on the screen of every clients with checkbox in front of every choices (for multi-answer) or radio button in front of every choices (for single answer). Also user must be able to interact with the user interface by selecting the choice(s).

### 3.3.3. The client must act like a server by opening the service and accepted the connection from nearby devices

The client will start it own service and wait for up to another 6 clients to connect (count server as one member of its connection).

### 3.3.4. The client may send the answer(s) to the server

Every clients must be able to send the answer(s) to its server (the device that it established the connection with) until the answer reached the server of this vote.

### 3.3.5. The client must be able to forward the question and answers from sender(s) to destination

After server sends the question, the clients will forward the question to every other clients connected to it and after clients send the answers, the client that they established the connection with will forward those answers until reached the server.

### 3.3.6. The clients that forward the answers mustn't see the answers of the senders

For privacy of the participants, the answer shouldn't be displayed on devices other than the server.

### 3.3.7. The clients can send the answer only one time per a vote

After sending question to the server, that client will be redirect to the end page automatically and since the server send the question one time only. Even restart the application, that client will not get any question from server.

According to the requirements analysis, Class of client can be shown briefly:

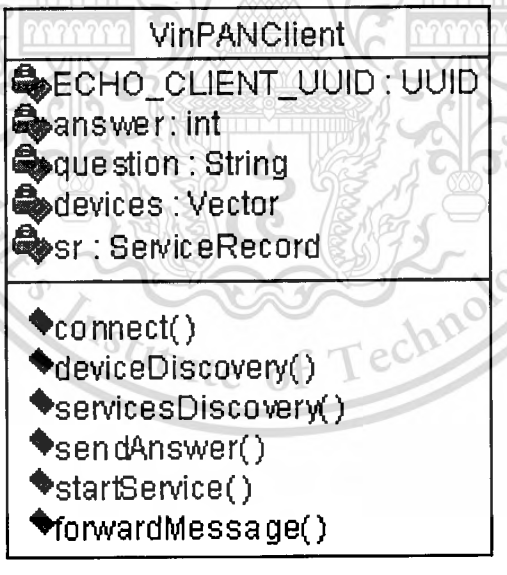


Figure 3.6: VinPANClient Class Diagram

Table 3.4: Description of VinPANClient class diagram (attributes)

| Attribute Name | Description |
|---|---|
| CLIENT_UUID | UUID of VinPAN |
| Answer | The answer of the client |
| Question | The question get from server |
| Devices | List of devices discovered |
| Sr | List of services discovered |

Table 3.5: Description of VinPANClient class diagram (methods)

| Method Name | Description |
|---|---|
| Connect | Connect to the highest quality of service server |
| deviceDiscovery | Search for devices nearby |
| servicesDiscovery | Search for services nearby |
| sendAnswer | Send the answer to the server if connected with client, send the answer to that client |
| startService | Initialize the server-like behavior (start service, search for nearby devices) |
| forwardMessage | Forward answers or question to the server or client which established the connection with it |

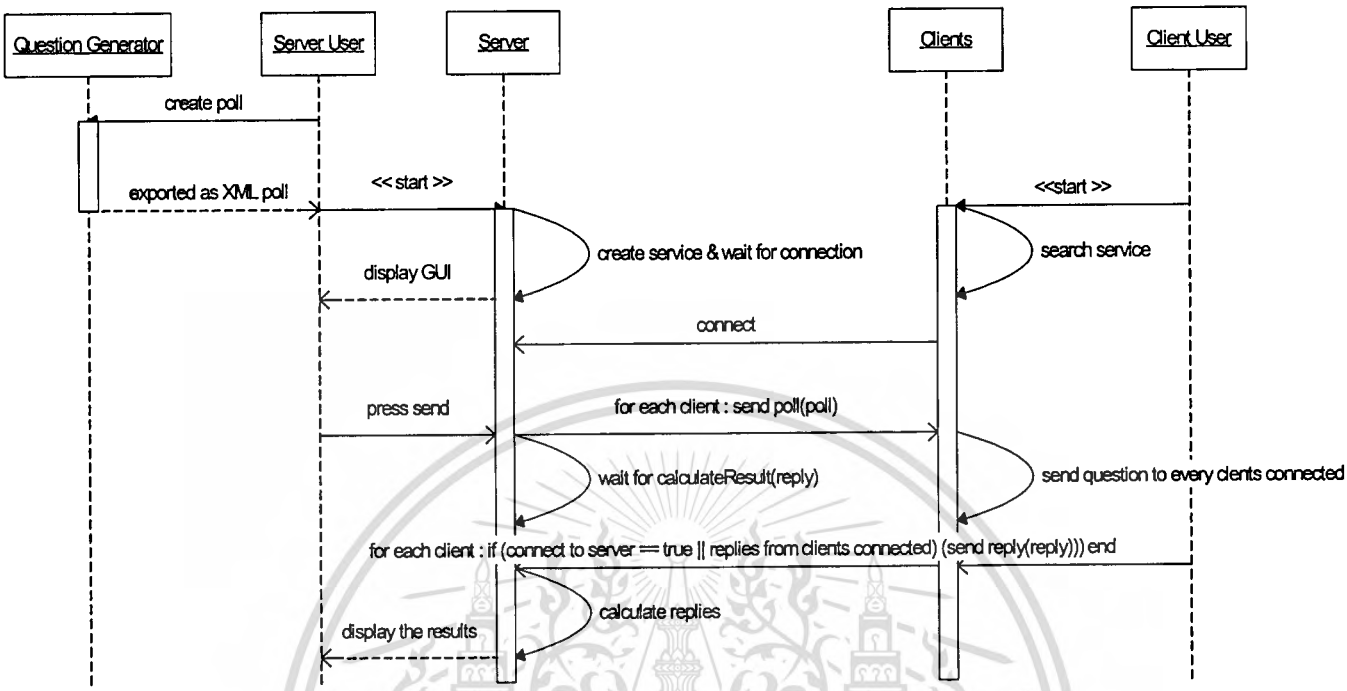


Figure 3.7: VinPANClient Sequence Diagram which shows the overview of our design

# Chapter 4
# Experiments and Results

## Scenario – What situation can VinPAN be used effectively?

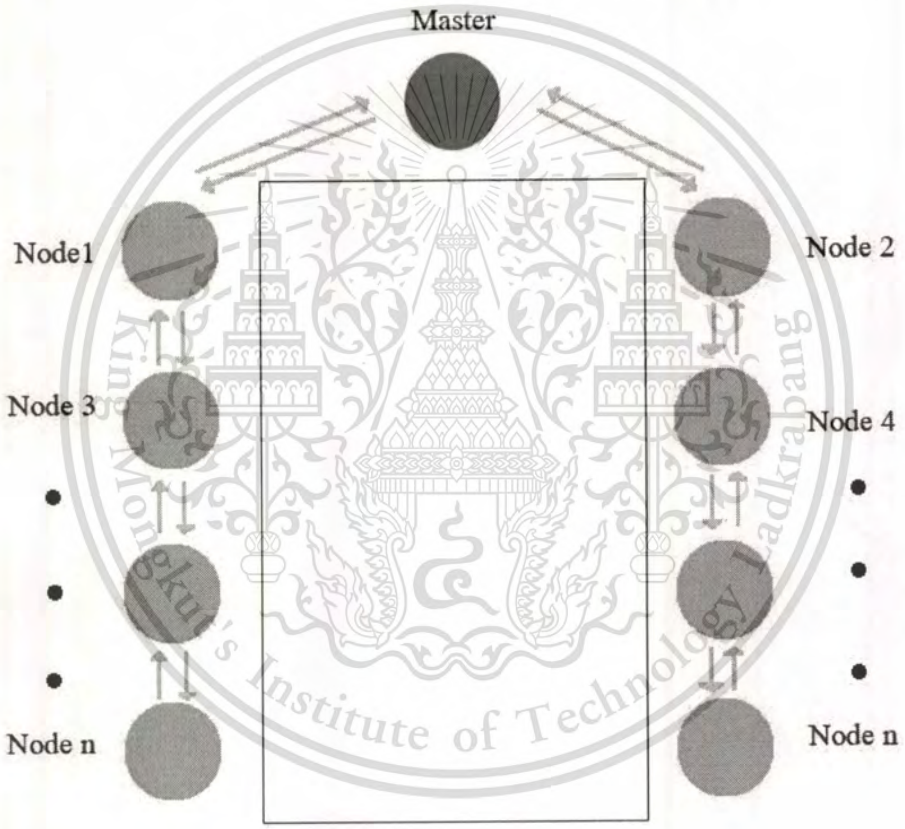


Figure 4.1: Meeting room case

Meeting room case

According to the limitation of VinPAN, this case should be the most reasonable. To imagine of this case, imagine the real meeting room of a company or an organization which is private area. The members of the meeting sit at their location and will not change during the meeting. When the chairman needs the opinion of the members, the answer of each member should be secret because, it may create the conflict among them.

## Experiment and Result

## Experiment 1

**Objective:** To test that the server successfully sends and receives the poll from two clients that connect to the server or not.

**Devices:** Server - Nokia 6680

Client - Nokia 5300

- Nokia 7610

1. Try to send and receive the poll with 3 mobile phones: 1 as server (VinPANServer), 2 as clients (VinPANClient)
2. Run the test and find out that every client get the poll or not.
3. Find out that the server can receives the replies from every clients or not.

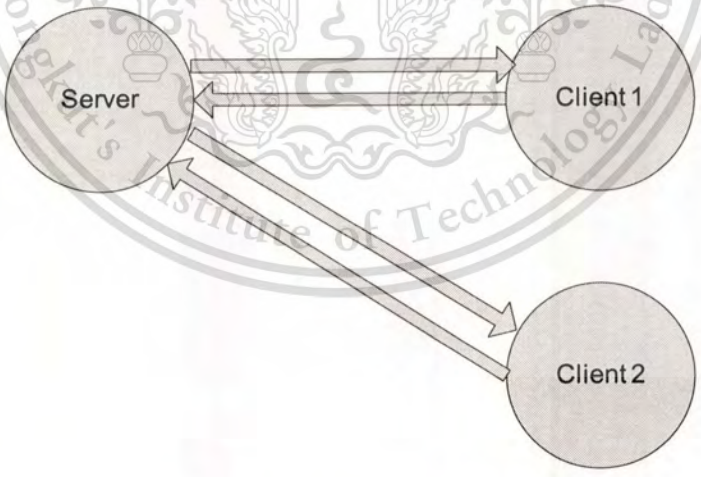


Figure 4.2 Experiment 1

## Experiment 1: Result

Every client got the poll that has been sent from the server. Also the Server successfully receives the answers from every client.

## Experiment 2

**Objective:** To test that VinPANClient(Client 1) successfully act as the server or not. If VinPANClient successfully act as the server, it's mean that VinPAN be able to form network more than seven devices in one network.

**Devices:** Server - Nokia 6680

Client - Nokia 5300

- Nokia 7610

1. Try to send and receive the poll with 3 mobile phones: 1 as server (VinPANServer), 2 as clients (VinPANClient)

2. Let the first client connects to the server and make sure that the second client can not connects to the server directly.

3. Let the second client connect to the first client. Then find out that the server can receive the replies from every client or not.

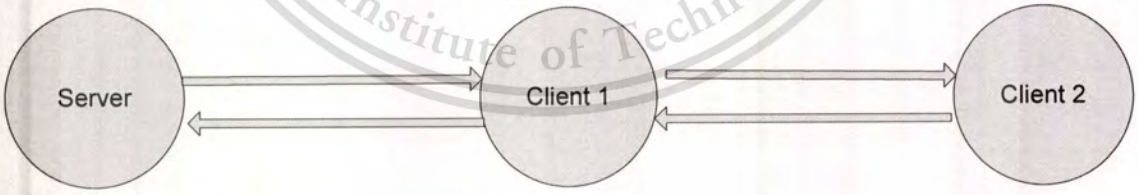


Figure 4.3 Experiment 2

## Experiment 2: Result

The second client successfully connects to the first client. Also the server can be able sends and receives the poll from every client.

## Experiment 3

After we had two experiments from the above, we went to Thanaruk accounting office to ask for their cooperate in testing our program.

The reason that we went to the office was we only had three mobile phones to test. So we couldn't guarantee that our program can work perfectly if they had more than seven devices or not.

At the office there is one chairman and twenty employees but the devices that can use our program are fifteen devices.

**Objective:** To test that the server successfully sends and receives the poll from all clients that connect to the server or not.

**Devices:** Sever – Nokia 3250
Clients – Nokia series that are capable to use VinPAN (MIDP 2.0, CLDC 1.1)

1. Let the chairman act as the server and sends the polls to the clients in his room on the second floor.
2. Find out those fifteen clients on the first floor can receive the poll and send the answers back to the server successfully or not.

**Experiment 3: Result**

The server successfully sends and receives the polls from every client. It means that the clients also act as the server and send the polls to other clients around them.

# Chapter 5
# Conclusion

The objectives of this project are using the Bluetooth of mobile phone for send and receive the polls. User that uses the server program can send the poll to every client and receive result of polls. After receive the question, every clients will be able to send the result back to the server. Every client will act as client server to start service and connect to other clients nearby and forward question and answers for them. The server will display the result in percentage.

VinPAN was developed successfully and become the high quality voting system for the conference room model ( every nodes are at fixed location, the distance between each node is less than 10 meters, transmission speed is not the issue). The project had been developed according to the project plan, almost on schedule, and complete the software development life cycle. VinPAN is small and light-weighted application in both memory (11 kb) and performance view.

## Limitation of VinPAN

These are few limitation of using VinPAN :

- Every node in the Scatternet should be in the fixed location. Since the connection pattern will be changed if the device move out of the signal range of its Server or ClientServer.
- The distance between each node should be less than 10 meters or the system will no longer reliable.
- The situation shouldn't strict about the transmission time.
- The voting should be done in the private room if the polling needs to be done on the specific subject (this problem can be fixed, see Chapter 3 optional requirements).
- There should be no interrupt frequencies that make transmission lost.

## Further development

For further developed in the future, VinPAN can be modified or upgraded in various ways likes adding some feature with on secure poll by embedded VinPAN with symmetric cryptography encrypted every messages to provided confidentiality and server can change the UUID for secret or specific group vote, add feature of storing more than one question on a mobile phone and can store results as file, or even improved the all GUI of VinPAN and in the result page to vary presentations likes graph and feature that compared the result with the old replays.

If developers can think, innovate, and create, the invention of new technology including mobile phone, Bluetooth, and VinPAN will never stop and someday it will beyond our imagination

# References

[1] Rajarshi Roy, Mukesh Kumar, Navin K. Sharma, and Shamik Surai, Member. "Bottom-up Construction of Bluetooth technology under a traffic-Aware Scheduling Scheme"

[2] Yang-Ick Joo, Tae-Jin Lee, Doo Seop Eom, Yeonwoo Lee, and Kyun Hyon Tchah. "Power-Efficient and QoS-Aware Scheduling in Bluetooth Scatternet for Wireless PANs"

[3] Josef Hallberg, Marcus Nilsson, Kare Synnes. "Positioning with Bluetooth"

[4] http://java.sun.com/javame/reference/apis.jsp

[5] Exploiting an Efficient Scheduling Algorithm for Bluetooth Scatternet

[6] MAC Scheduling for Fixed Broadband Wireless Access Systems

[7] http://www.siamphone.com/catalog/nokia/nokiaMAIN.html

[8] http://java.sun.com/j2se/1.5.0/docs/api/

[9] A Priority-Based MAC Scheduling Algorithm for Enhancing QoS Support in Bluetooth Piconet

[10] http://www.mobileinfo.com/Bluetooth/FAQ.htm

[11] http://discussion.forum.nokia.com/forum/forumdisplay.php?f=38

[12] http://www.siamphone.com/catalog/samsung/main.htm

[13] http://dict.longdo.com/

[14] http://en.wikipedia.org/wiki/Bluetooth

[15] http://en.wikipedia.org/wiki/Personal_area_network

[16] http://en.wikipedia.org/wiki/IEEE_802.15

[17] http://www.bluetooth.com/Bluetooth/Technology/Works/

[18] http://web.archive.org/web/20060519034246/www.niksula.hut.fi/~jiitv/bluesec.html

[19] http://www.wirelessnetdesignline.com/howto/showArticle.jhtml?articleID=180201430

[20] http://www.newswireless.net/index.cfm/article/629

[21] http://www.thebunker.net/resources/bluetooth

[22] http://www.theregister.co.uk/2004/06/15/symbian_virus/

[23] http://www.securenetwork.it/ricerca/whitepaper/download/bluebag_brochure.pdf

[24] http://www.eclipseme.org/

[25] http://java.sun.com/products/sjwtoolkit/

# Appendix

## List of some devices that are capable to use VinPAN

### BlackBerry

- BlackBerry 7520: BlackBerry7520/4.0.0 Profile/MIDP-2.0 Configuration/CLDC-1.1
- BlackBerry 8700: BlackBerry8700/4.1.0 Profile/MIDP-2.0 Configuration/CLDC-1.1 VendorID/100
- BlackBerry 8100: Mozilla/4.0 BlackBerry8100/4.2.0 Profile/MIDP-2.0 Configuration/CLDC-1.1 VendorID/100

### BenQ-Siemens

- S68: SIE-S68/36 UP.Browser/7.1.0.e.18 (GUI) MMP/2.0 Profile/MIDP-2.0 Configuration/CLDC-1.1
- EF81: SIE-EF81/58 UP.Browser/7.0.0.1.181 (GUI) MMP/2.0 Profile/MIDP-2.0 Configuration/CLDC-1.1

### Motorola

- Motorola L6:MOT-L6/0A.52.2BR MIB/2.2.1 Profile/MIDP-2.0 Configuration/CLDC-1.1

### Nokia

- Nokia 6280:Nokia6280/2.0 (03.60) Profile/MIDP-2.0 Configuration/CLDC-1.1
- Nokia N70:NokiaN70-1/5.0616.2.0.3 Series60/2.8 Profile/MIDP-2.0 Configuration/CLDC-1.1
- Nokia N80:NokiaN80-1/3.0(4.0632.0.10) Series60/3.0 Profile/MIDP-2.0 Configuration/CLDC-1.1

### Samsung

- Samsung D600: "SAMSUNG-SGH-D600/1.0 Profile/MIDP-2.0 Configuration/CLDC-1.1 UP.Browser/6.2.3.3.c.1.101 (GUI) MMP/2.0"

**SonyEricsson**

- SonyEricsson K510i: "SonyEricssonK510i/R4CJ Browser/NetFront/3.3 Profile/MIDP-2.0 Configuration/CLDC-1.1"
- SonyEricsson K610i: "SonyEricssonK610i/R1CB Browser/NetFront/3.3 Profile/MIDP-2.0 Configuration/CLDC-1.1"
- SonyEricsson K700: "SonyEricssonK700/R1A Profile/MIDP-1.0 MIDP-2.0 Configuration/CLDC-1.1"
- SonyEricsson W800i: "SonyEricssonW800i/R1AA Browser/SEMC-Browser/4.2 Profile/MIDP-2.0 Configuration/CLDC-1.1"
- SonyEricsson W900i: "SonyEricssonW900i/R5AH Browser/NetFront/3.3 Profile/MIDP-2.0 Configuration/CLDC-1.1"

**LG Electronics**

- LG U880: "LG/U880/v1.0"
- LG B2050: "LG-B2050 MIC/WAP2.0 MIDP-2.0/CLDC-1.0"
- LG C1100: "LG-C1100 MIC/WAP2.0 MIDP-2.0/CLDC-1.0"
- LG CU8080: "LGE-CU8080/1.0 UP.Browser/4.1.26l"
- LG G1800: "LG-G1800 MIC/WAP2.0 MIDP-2.0/CLDC-1.0"
- LG G210: "LG-G210/SW100/WAP2.0 Profile/MIDP-2.0 Configuration/CLDC-1.0"
- LG G220: "LG-G220/V100/WAP2.0 Profile/MIDP-2.0 Configuration/CLDC-1.0"
- LG G232: "LG-G232/V100/WAP2.0 Profile/MIDP-2.0 Configuration/CLDC-1.0"
- LG G262: "LG-G262/V100/WAP2.0 Profile/MIDP-2.0 Configuration/CLDC-1.0"
- LG G5200: "LG-G5200 AU/4.10"
- LG G5600: "LG-G5600 MIC/WAP2.0 MIDP-2.0/CLDC-1.0"
- LG G610: "LG-G610 V100 AU/4.10 Profile/MIDP-1.0 Configuration/CLDC-1.0"
- LG G622: "LG-G622/V100/WAP2.0 Profile/MIDP-2.0 Configuration/CLDC-1.0"
- LG G650: "LG-G650 V100 AU/4.10 Profile/MIDP-1.0 Configuration/CLDC-1.0"
- LG G660: "LG-G660/V100/WAP2.0 Profile/MIDP-2.0 Configuration/CLDC-1.0"
- LG G672: "LG-G672/V100/WAP2.0 Profile/MIDP-2.0 Configuration/CLDC-1.0"

- LG G682: "LG-G682 /V100/WAP2.0 Profile/MIDP-2.0 Configuration/CLDC-1.0"
- LG G688: "LG-G688 MIC/V100/WAP2.0 MIDP-2.0/CLDC-1.0"
- LG G7000: "LG-G7000 AU/4.10"
- LG G7050: "LG-G7050 UP.Browser/6.2.2 (GUI) MMP/1.0 Profile/MIDP-1.0 Configuration/CLDC-1.0"
- LG G7100: "LG-G7100 AU/4.10 Profile/MIDP-1.0 Configuration/CLDC-1.0"
- LG G7200: "LG-G7200 UP.Browser/6.2.2 (GUI) MMP/1.0 Profile/MIDP-1.0 Configuration/CLDC-1.0"
- LG G822: "LG-G822/SW100/WAP2.0 Profile/MIDP-2.0 Configuration/CLDC-1.0"
- LG G850: "LG-G850 V100 UP.Browser/6.2.2 (GUI) MMP/1.0 Profile/MIDP-1.0 Configuration/CLDC-1.0"
- LG G920: "LG-G920/V122/WAP2.0 Profile/MIDP-1.0 Configuration/CLDC-1.0"
- LG G922: "LG-G922 Obigo/WAP2.0 MIDP-2.0/CLDC-1.1"
- LG G932: "LG-G932 UP.Browser/6.2.3(GUI)MMP/1.0 Profile/MIDP-2.0 Configuration/CLDC-1.1"
- LG L1100: "LG-L1100 UP.Browser/6.2.2 (GUI) MMP/1.0 Profile/MIDP-1.0 Configuration/CLDC-1.0"
- LG T5100: "LG-T5100 UP.Browser/6.2.3 (GUI) MMP/1.0 Profile/MIDP-1.0 Configuration/CLDC-1.0"

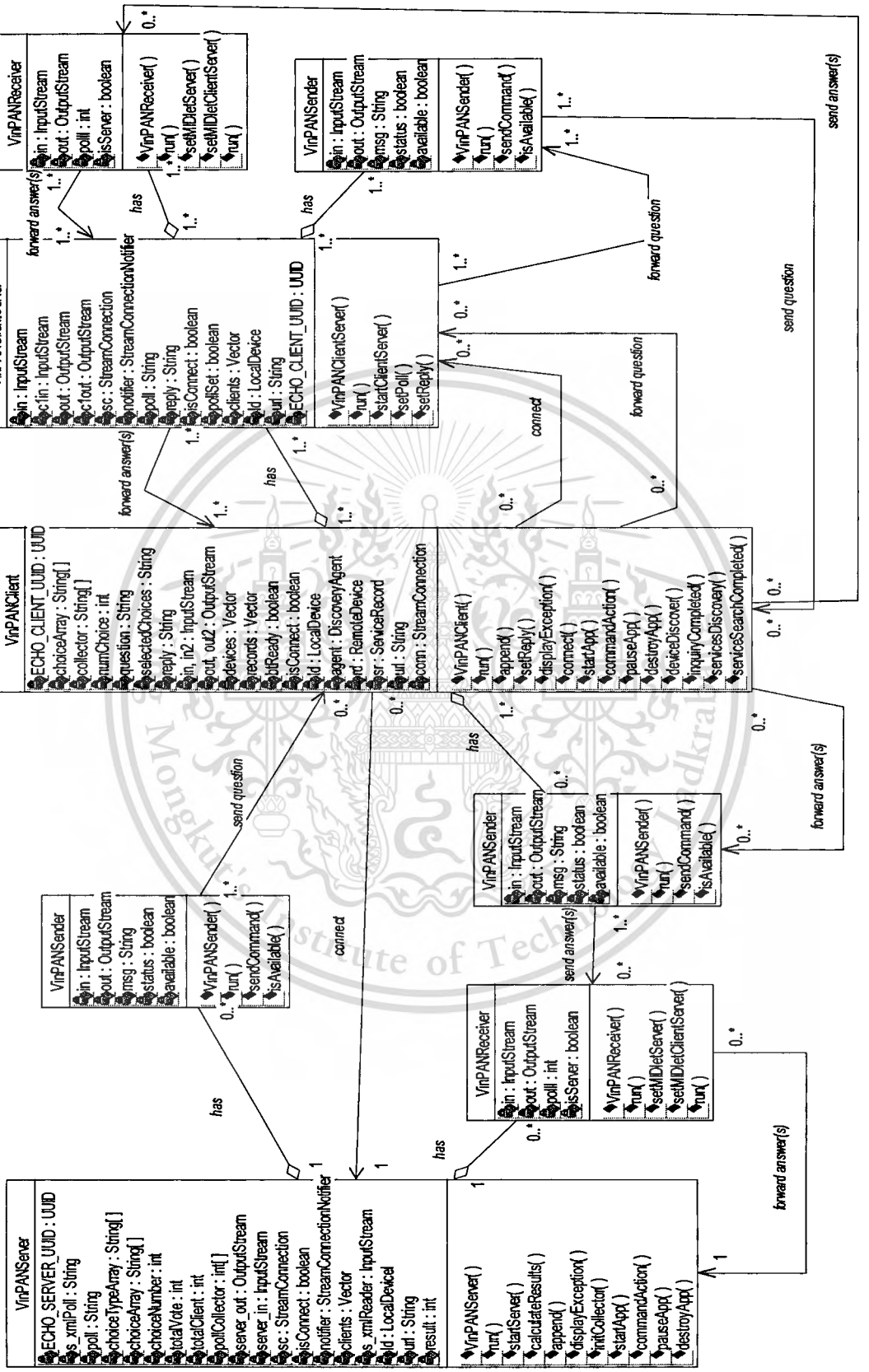


Figure A.1: Class Diagram (Server – ClientServer Model)

## Attributes Description (Figure A.1)

| Attribute Name | Type | Description |
|---|---|---|
| ECHO_SERVER_UUID | UUID | Server's UUID |
| S_xmlPoll | String | Contained xml poll from /res |
| choiceTypeArray | String[ ] | Array contained choice type: radiobutton,checkbox (between <type> and </type>) |
| choiceArray | String[ ] | Array contained poll choices (between <choice1-5>and</choice1-5>) |
| choiceNumber | int | Number of choice for server (between <choice_number> and </choice_number> |
| Sc | StreamConnection | Start the service and accept the connection from clients |
| isConnect | boolean | Check the connectivity of server (if already start service, set to true) |
| notifier | StreamConnectionNotifier | Use for creating the url |
| clients | Vector | Store the name of clients that connected to the server |
| s_xmlReader | InputStream | InputStream for reading the poll from /res |
| ld | LocalDevice | Device that found during the device |

| | | |
|---|---|---|
| | | search |
| url | String | Server's url for start service |
| in | InputStream | InputStream for the system |
| out | OutputStream | OutputStream for the system |
| msg | String | Poll question that passed from server to ClientHandler |
| status | Boolean | Status of the server, ready to send the poll or not (if the poll already sent, set to true) |
| reply | String | Answer from clients |
| ECHO_CLIENT_UUID | UUID | Client's UUID which is same as Server's |
| collector | String[ ] | Contained the answer of client after check the correctness (1 for selected, otherwise, 0) |
| numChoice | Int | Number of choice for client (between <choice_number> and </choice_number> |
| question | String | Poll's question part (between <question> and </question>) |
| selectedChoices | String | Contained the concatenated answer of client after check the correctness |
| Devices | Vector | List of devices found |
| Records | Vector | List of devices connected to the client |
| btReady | boolean | Check the |

| | | |
|---|---|---|
| | | connectivity of client (if already connected, set to true) |
| Agent | DiscoveryAgent | Search for the device near by |
| Rd | RemoteDevice | Handle the transmitting protocol for connected devices |
| sr | ServiceRecord | Record the server device |
| conn | StreamConnection | Open the connection |
| poll | String | Contained answer from each clients. Use in server to calculate the results |
| pollCollector | Int[ ] | Array contained the score of each choice |
| available | boolean | Check the availability of the VinPANSender (if poll from server has set, set to true) |
| isServer | boolean | The derived class type : server, clientserver (if server, set to true) |
| pollSet | boolean | Status of clientserver (if the poll is already set, set to true) |
| c1in | InputStream | InputStream |

| | | |
|---|---|---|
| | | from clientserver to clients that connected to it |
| clout | OutputStream | OutputStream from clientserver to clients that connected to it |

## Methods Description (Figure A.1)

| Method name | Parameters | Description/Return types |
|---|---|---|
| **VinPANServer** | | **Main server** |
| VinPANServer | - | Server's constructor handle initialization of variables, creation of GUI, and read the poll from xml file |
| run | - | Check the connectivity of server, start service, establish the connection, and collect the clients devices. |
| startServer | - | Initialize the server (create service and wait for accept the connection from client) |
| calcualteResults | String poll | Receive the replies from clients, calculate the results in percentage, and |

| | | |
|---|---|---|
| | | display on screen. |
| append | String text | Concatenate the text |
| displayException | String e | Handle the exception display |
| initCollector | - | Initiate the answer's container |
| startApp | - | Start the MIDlet process |
| commandAction | Command cmd, Displayable d | Handle the commands (exit, send), set main page to results receiver |
| pauseApp | - | Handle the pause application behavior |
| destroyApp | Boolean unconditional | Handle the deletion of application |
| **VinPANClient** | | **Main client** |
| VinPANClient | - | Client's constructor handle initialization of variables |
| Run | - | Check the connectivity of client, create the GUI from poll that sent from server, and after that, wait for the replies from other clients connected to it |
| append | String text | Concatenate the text |
| setReply | String reply | Forward reply to the clientserver |
| displayException | String e | Handle the exception display |
| connect | - | Start client, search for devices near by, and connect to server |
| startApp | - | Start the MIDlet process |
| commandAction | Command cmd, Displayable d | Handle the commands (exit, |

| | | |
|---|---|---|
| | | send), set main page to end page |
| pauseApp | - | Handle the pause application behavior |
| destroyApp | Boolean unconditional | Handle the deletion of application |
| deviceDiscovered | RemoteDevices btDevice, DeviceClass dc | Add the device into the list of devices found |
| inquiryCompleted | Int code | Synchronized method of deviceDiscovered |
| servicesDiscovered | Int id, ServiceRecord[] r | Add the service devices (server) into the list of service devices found |
| servicesSearchCompleted | Int id, int code | Synchronized method of servicesDiscovered |
| **VinPANSender** | | **Sending Protocal** |
| VinPANSender | InputStream in, OutputStream out | Sender's constructor handle the protocol for transmitting data |
| run | - | Sending data process |
| sendCommand | String s | Start the sending process |
| isAvailable | - | Check the availability of the sender |
| **VinPANReceiver** | | **Receiving Protocal** |
| VinPANReceiver | InputStream in, OutputStream out, String receiverType | Receiver's constructor handle checking the derived class type and initiate variables |
| Run() | - | Start and control the receiving process |
| setMIDletServer | VinPANServer r_server | Let the receiver control the server MIDlet |

| | | |
|---|---|---|
| setMIDletClientServer | VinPANClientServer r_clientserver | Let the receiver control the clientserver MIDlet |
| **VinPANClientServer** | | Server's function in client |
| VinPANClientServer | InputsStream in, OutputStream out | Initiate the variables and start the server |
| run | - | Check the connectivity of server, start service, establish the connection, and collect the clients devices |
| startClientServer | - | Search for device near by, start service, and wait for connection from clients |
| setReply | String reply | Send replies to server |
| setPoll | String poll | Send question to clients connected |

## Emulator test

The program is written in Java and running in the built-in emulator phone of Java ME. The results are captured in the following figures:

1. After lunch the VinPAN, The user can select to act as server or receiver. (Figure A.2)
2. After select Lunch button, VinPANServer, the application will ask user to confirm to start the server. (Figure A.3)

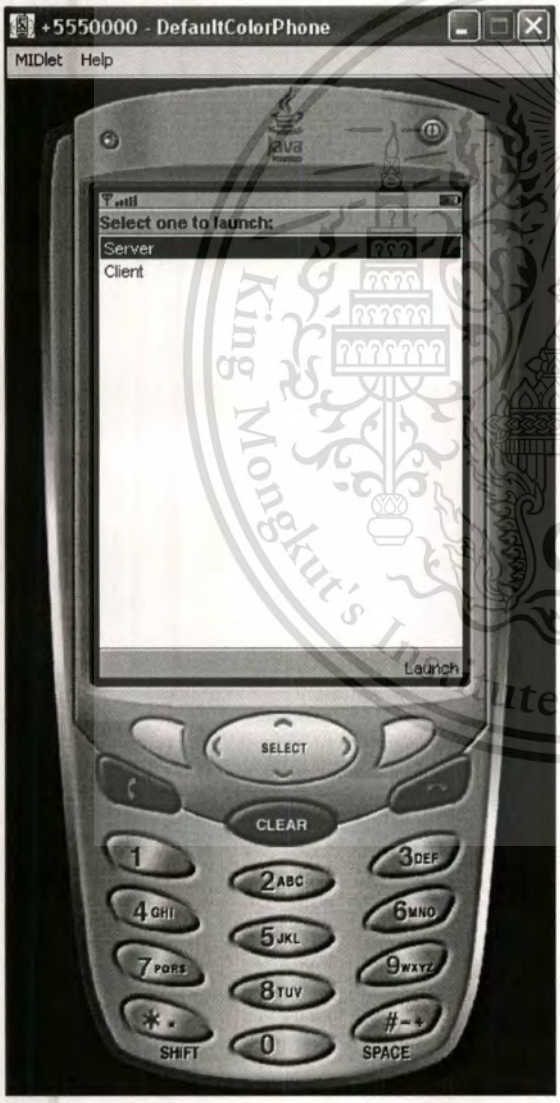


Figure A.2: VinPAN first page.

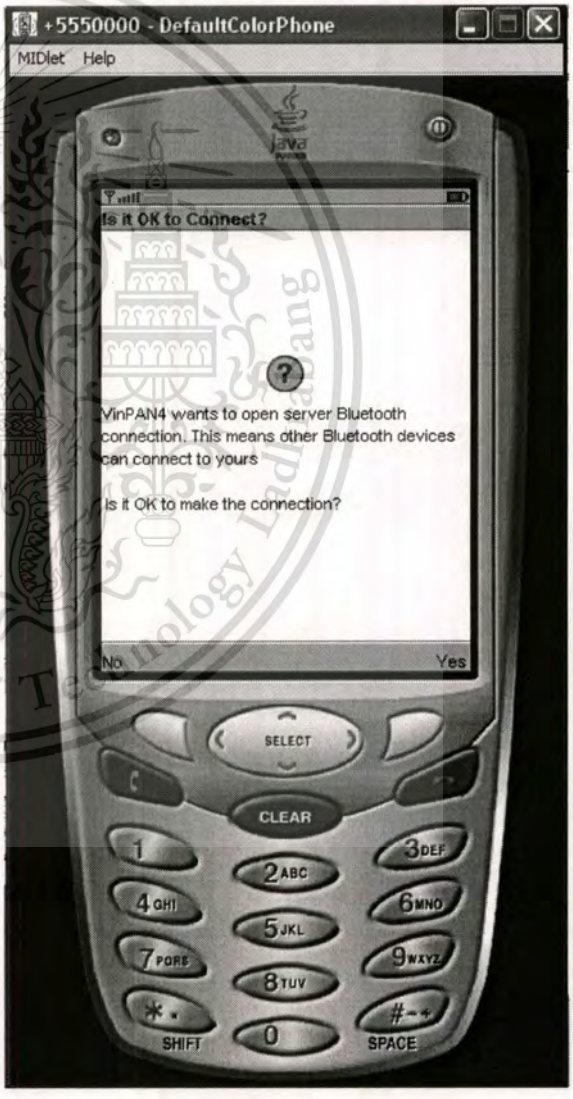


Figure A.3: After select Lunch button

3. After the confirmation, it'll get the inputstream from the poll.xml and create the GUI and put the separated data to right place. (Figure A.4)
4. Assume that a client want to open the connection, the application will display the warning message and ask user to confirm the connection. (Figure A.5)

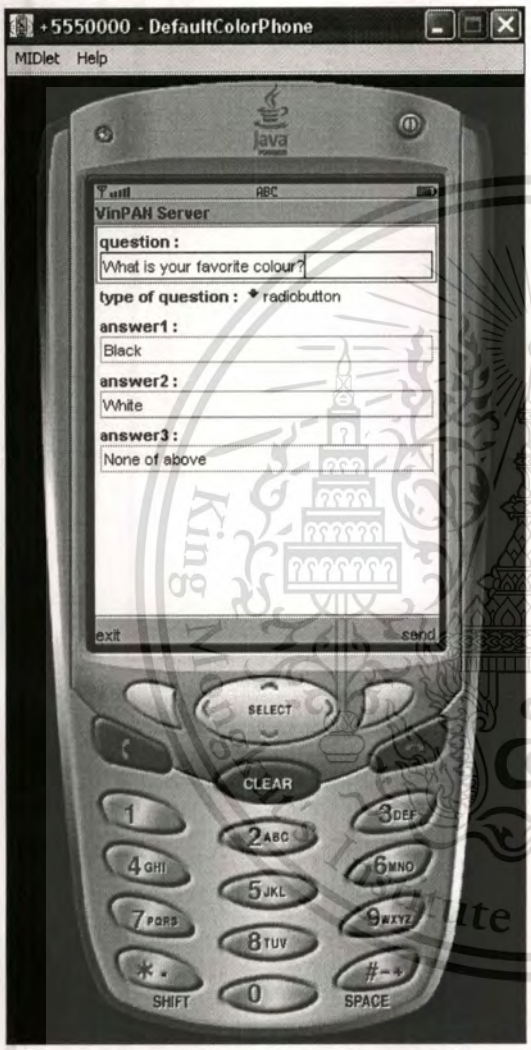


Figure A.4: Question from poll.xml

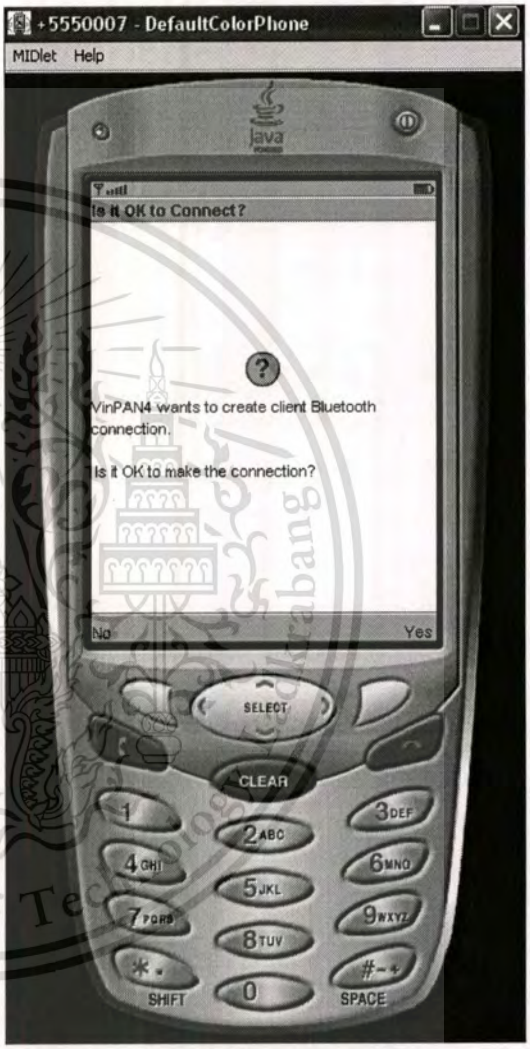


Figure A.5: Warning Message

5. After the confirmation, it'll change to the standby window to wait for the poll from server. At this stage, the send button can't be performing. (Figure A.6)

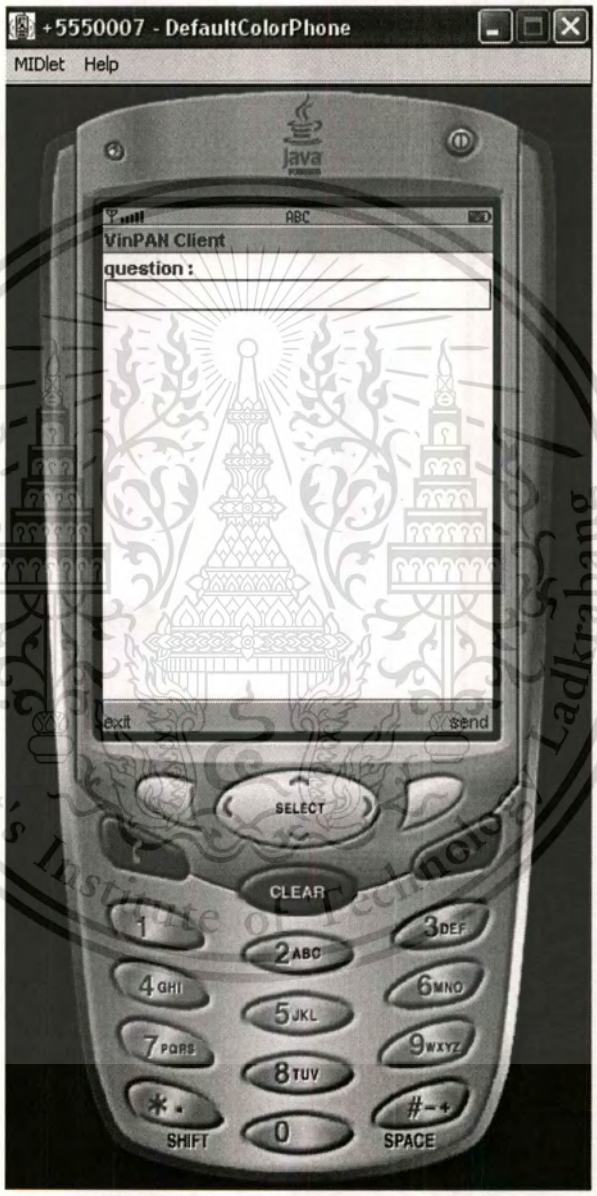


Figure A.6: Standby window

6. After the server send poll, the server will change the main window to "Feedback Receiver Processing…" window which will wait for the reply from the client. At the Client side, after receive the poll from server, the program will generate the component automatically according to the poll. At this point, user can answer the poll from their mobile phone. (Figure A.7)


Figure A.7: After the server send poll

7. After server received the polls from any clients, it'll display the result in server's mobile phone. (Figure A.8)

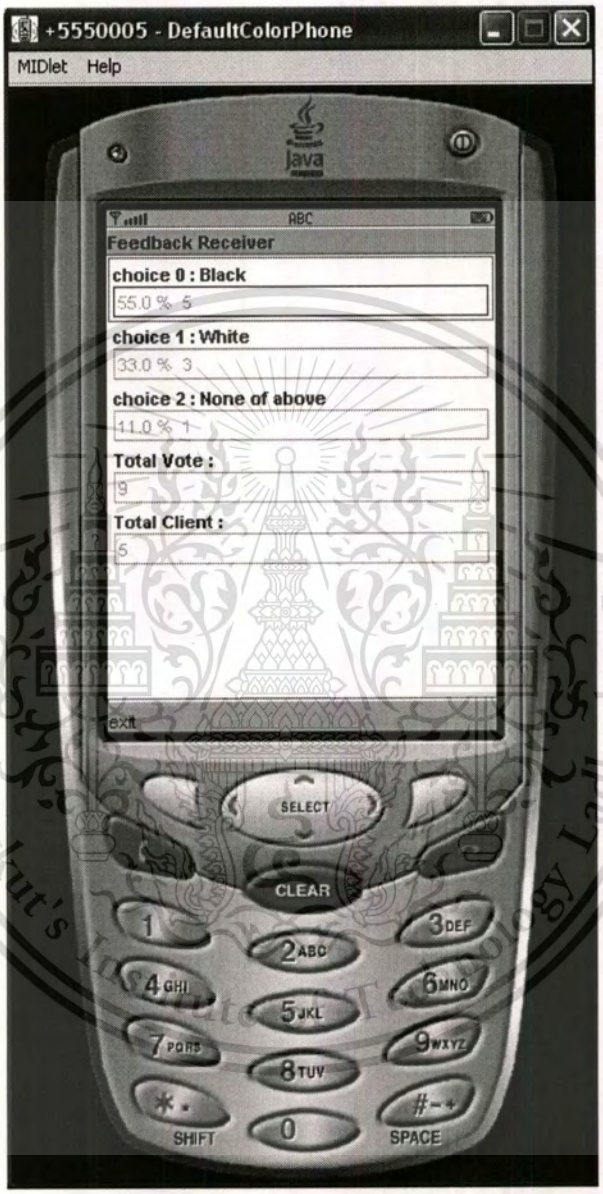


Figure A.8: Result