International Journal of
Technology and Systems
(IJTS)

The Comprehensibility of a Z Specification Language and Its Implementation in JAVA for Implementing Atomic Read/Write Shared Memory in Mobile Ad hoc Network

Reham.A.Shihata

IPRJB
INTERNATIONAL PEER REVIEWED
JOURNAL AND BOOK PUBLISHING

International Journal of Technology and Systems
ISSN 2518-881X (Online)
Vol.7, Issue 1, No.1, pp 1-19, 2022 www.iprjb.org

The Comprehensibility of a Z Specification Language and Its Implementation in JAVA for Implementing Atomic Read/Write Shared Memory in Mobile Ad hoc Network

Reham.A.Shihata

PhD in Computer Science, EL Menoufia University –Egypt.
Software Consultant .in Egyptian Syndicate of Programmers & Scientists

anwerreham@yahoo.com

Abstract

Comprehensibility is often raised as a problem with formal notations; yet formal methods practitioners dispute this. In a survey, one interview said "formal specifications are no more difficult to understand than code". Measurement of comprehension is necessarily. In this paper, a comprehension of Z specification with that of its implementation in JAVA for atomic object Read/Write shared memory in mobile ad hoc network is performed.

Keywords: Z Specification Language, Formal Specification, GUI, Java Language, Mobile Ad Hoc Network.

International Journal of Technology and Systems
ISSN 2518-881X (Online)
Vol.7, Issue 1, No.1, pp 1-19, 2022 www.iprjb.org

1. Introduction

The Geoquorums approach for implementing atomic rea-d/write shared memory in mobile ad hoc networks, this approach is based on associating abstract atomic objects with certain geographical locations this algorithm for mobile ad hoc networks which uses no pre-existing infrastructure unlike cellular networks that depend on fixed, wired base stations [1] [2][3]. Instead the network is formed by the mobile nodes themselves, which co-operate to route communication from sources to destinations.

Specify an Atomic in Geoquorums Approach Object as a Variable Type[4][5][6]:

V1 a set of legal values (i.e states) for the object

$v0 \in V1$, an initial value (i.e states) for the object

Invocations, a set of invocations

Responses , a set of responses

δ, the transition F_n ,a mapping from: (invocations ×V) $\longrightarrow$ (responses ×V)

Mathematical Notation for Geoquorums Approach:

- I the totally- ordered set of node identifiers.
- $i_0 \in I$, a distinguished node identifier in I that is smaller than all order identifiers in I.
- S, the set of port identifiers, defined as $N^{<0} \times OP \times I$,

  where OP= (get, put, confirm, recon- done).
- O, the totally- ordered, finite set of focal point identifiers.
- T, the set of tags defined as $R^{\geq 0} \times I$.
- U, the set of operation identifiers, defined as $R^{\geq 0} \times S$.
- X, the set of memory locations for each $x \in X$.
  - V_x the set of values for x
  - $v_{0,x} \in V_x$, the initial value of X
- M, a totally-ordered set of configuration names
- $c_0 \in M$, a distinguished configuration in M that is smaller than all other names in M.
- C, totally- ordered set of configuration identifies, as defined as:

  $R^{\geq 0} \times I \times M$
- L, set of locations in the plane, defined as R× R

An Atomic Object is Specified as a Variable type, τ, consists of

International Journal of Technology and Systems
ISSN 2518-881X (Online)
Vol.7, Issue 1, No.1, pp 1-19, 2022

- V, a set of legal values (i.e states) for the object
- $v_0 \in V$ an initial value (i.e states) for the object.
- invocations, a set of invocations.
- responses, a set of responses
- δ, the transition F_n , a mapping from:

$(\text{invocations} \times V) \longrightarrow (\text{responses} \times V)$

That maps every invocation and state to a response and a new state.

Specify a Read/Write object as a Variable Type [7][8][9]

V, as arbitrary set of values for the atomic object

$v_0 \in V$ an arbitrary initial value
invocations= {read} ∪ {write (v): v ϵ V}

responses= {read- ack (v) : v ϵ V) ∪ {write-ack}
δ is defined as:
- δ (read , v)$\longrightarrow$ <read- ack (v), v >
- δ (write (v/), v)$\longrightarrow$ < write- ack, v/>

Formal Sequential Specification for Abstract Read/Write Object[10][11][12]:

State

Value, initially v_0

Operation

Read ( )

Return read-ack (value)
Write (new-value)
Value $\leftarrow$ new- value
return write-ack ( )

Canonical Atomic Object Specification of

type τ = < V, v_0, invocations, responses, δ >, for the set ,φ , of ports
Signature:

Input:

invoke $(inv)_p$,inv ϵ invocations, p ϵ φ, the invocations defined by the variable type τ

Outputs:

Respond $(resp)_p$, resp ϵ responses, p ϵ φ, the responses defined by the variable type τ

Internal:

Perform $(inv, v, resp, v')_p$, inv ϵ invocations, resp ϵ responses, v,v′ ϵ V,

p ϵ φ,

perform the transitions defined by the variable type τ

State:

val ϵ V a value , initially V_0

inv- buffer, a set of pairs ,< inv, p> for invocations, inv ϵ invocations, by port p, p ϵ φ, initially Φ

resp- buffer, a set of pairs ,<resp, p> for responses, resp ϵ responses, to port p, p ϵ φ, initially Φ

Trasitions:

Input invoke $(inv)_p$

Effect:

inv-buffer ←inv-buffer ∪ {<inv, p>}

Output respond $(resp)_p$

Precondition:

< resp, p> ϵ resp-buffer

Effect:

resp-buffer← resp-buffer\ {<resp,p>}

<u>Definition of The Put/Get Variable Type τ</u> <u>[13][14][15]:</u>

put/get variable type τ

State

tag ϵ T, initially, $<0, i_o>$

value ϵ V, initially V_0

config-id ϵ C, initially $<0, i_o, C_0>$

Confirmed- set T, initially Φ

recon ip, a Boolean, initially false

Operations:-

Put (new- tag, new-value, new- config -id)

If (new-tag> tag) then

value← new- value

tag ← new- tag

If (new – config- id > config- id) then

config- id ← new – config- id

recon- ip← true

return put- ack (config- id, recon- ip)

get (new- config- id)

If (new- config- id >config- id) then

config- id ← new – config- id

Recon- ip← true

Confirmed ←(tag ϵ confirmed- set)

return get ack (tag, value, confirmed, config- id, recon- ip)

confirm (new- tag)

confirmed- set ← confirmed- set ∪ {new- tag}

return confirm- ack

recon- done (new- config- id)

If (new- config- id =config- id) then

recon- ip ← false

return recon- done- ack ( )

2. Z Specification for Implementing Atomic Read/Write Shared Memory in Mobile Ad hoc Networks

Objects

Focal point object type Fpo: atomic object Fpo value: N

Op-recon- ip: True/False Op. record: record type Op. phase: phase-Type Op. tag: Variable

Op. value: Variable

Signature Type

International Journal of Technology and Systems
ISSN 2518-881X (Online)
Vol.7, Issue 1, No.1, pp 1-19, 2022

Signature : External type
Signature: Internal type

Operation Manager Client Type OMC: put variable type

OMC: get variable type

Focal Point Emulator Type

< current- port- number, op, i> > 0

Focal Point Emulator Client Type
invoke/ respond interface: interface type < current -port -number, op, i > > 0

Focal Point Emulator Server Type Local broadcast > 0

LBcast Service: Service type **Initialization**

put ?: IP variable type

get- ack- response: IP variable type get ?: IP variable type

put- ack- response: IP variable type tag ?: N↔ order on values

confirm- ack- response: IP variable type config – id: parameter

recon- done- ack : IP variable type

confirmed- set?:

set↔ tags recon- ip

flag?: True/False

recon- done: True/False

Operations

Put- invocation

∃New- tag .int -type > tag. int-type

∃ New-Config- id. Parameter- type > config-id. Parameter-type ∃recon- done= True

put- invocation/ = put-ack- response

Get- invocation

International Journal of Technology and Systems
ISSN 2518-881X (Online)
Vol.7, Issue 1, No.1, pp 1-19, 2022

∃ new- config- id> config-id
get- invocation/= get -ack -response

Confirm -
invocation ∃ new-
tag > tag

∃ confirmed- set = confirm –set ∪ {new- tag?
:N} confirm- invocation= confirm- ack-
response

Recon –done
invocation ∃ recon-
ip= True

∃ new- config- id. parameter = config- id.
Parameter recon- done- invocation/= recon-
done- ack

√ ¬ conf-id=op.recon-conf-id ⟶ recon-
ip=false **Success**

No - Error!: Success
No - Error!: Recon- Ip
No - Error!: Tag

No - Error!: Config – Id
Put-Ack- Response= Okay
Get-Ack Response= Okay

Confirm- ack –response= Okay
Recon- Done- Ack= Okay

((Put- Type , Get -Type) ^ Success)
((Confirm- Type , Recon- Done- Type) ^ Success)

3- JAVA Code for Implementing Atomic Read/ Write Shared Memory in Mobile Ad hoc Networks

```
Import java. Lang. Exception;
Class invariant exception extends Exception {
```

Public invariant exception (int id) {surper (id) ;}

Class put type

int new-tag -
type int tag-
type

int new- config-
id int config- id-
type

Public put type (int: new- tag) throws invariant
exception { if (recon- done== true) {

Invariant exception id =new invariant
exception ("invariant: new-tag must be >
=1")

put- invocation= put –ack- response
system.out.Println (" put- ack- response=
okay") ;} }

Class gettype
{ int config-
id

int new- config –id

Puplic gettype (int: new -config- id) throws
invariantexception { if (new- config- id > config- id){

invariantexception id= new
invariantexception (" invariant: new-
config- id must be > =1"); get -
invocation= get- ack- response}

throw id; }

International Journal of Technology and Systems
ISSN 2518-881X (Online)
Vol.7, Issue 1, No.1, pp 1-19, 2022
www.iprjb.org

get-invocation=get-ack-response}
system.out.Println (" get- ack- response=
okay");} }

Class confirmtype {
Int op.recon- conf-
id Int new- tag

Int new- config-
id Int confirmed-
set

Public confirmtype (int: confirmed- set) throws
invariantexception if (new- tag> && confirmed – set {new-
tag}) {Invariantexception id= new invariantexception

(" invariant: confirmed- set must be
>=1); throw id;}

confirm- invocation = confirm-ack-
response system .out. println ("confirm- ack
–response"); }

Class recon-donetype {

int new- config-id parameter

Public recon-donetype (int: new- config-id) throw
invariantexception if (new- config-id parameter== config-id-
parameter){ invariantexception id= new invariantexception

(" invariant: new- config- id. Parameter
>=1")} throw id;

recon- done invocation= recon- done-
ack }

Public boolean recon –ip as (confirmtype: conf- id
confirm) {boolean recon- ip = true;

boolean op. recon- conf- id = true;

if (conf- id !== op.recon- conf- id) recon- ip= false; return recon- ip;}

}

system. out. println ("confirm- ack- response"); }

Class adhoctype{ boolean ammadh= true; boolean puttype= true; boolean gettype= true;

boolean confirmtype= true; boolean recon -donetype= true; boolean success = true;

Public ammadh type (boolean success) throws invariantexception

if ((puttype | | gettype) && success) | | ((confirmtype | | recon- done type && success)) Invariantexception id= new invariantexception

(" invariant: success must be true"); throw id;)

put-ack-response= okay; get- ack- response= okay;

confirm- ack – response = okay; recon –done –ack= okay; ammadh= okay

system. out. println (" put- ack- response= okay"); system. out. println (" get- ack- response= okay"); system. out. println (" confirm- ack- response= okay"); system. out. println (" recon- done- response= okay"); }

International Journal of Technology and Systems
ISSN 2518-881X (Online)
Vol.7, Issue 1, No.1, pp 1-19, 2022

3. Constructing a Java Code for Implementing Atomic Read /Write Shared Memory in Mobile Ad Hoc Networks

This code will be divided into four phases by using GUI in Java

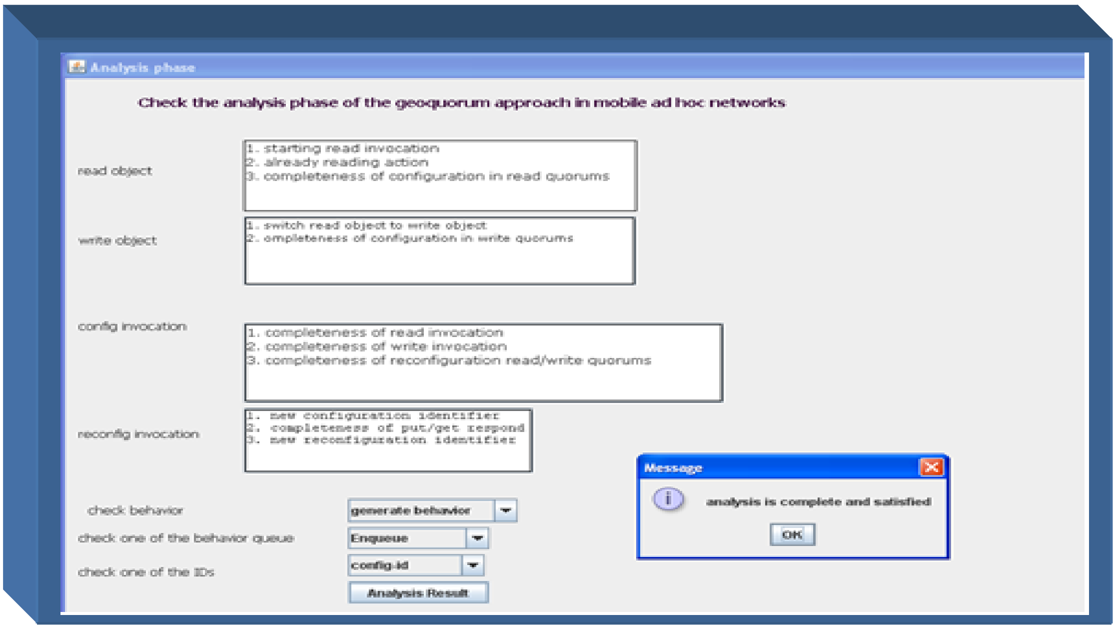

Fig.1 The GUI of Analysis Phase Using Java Code

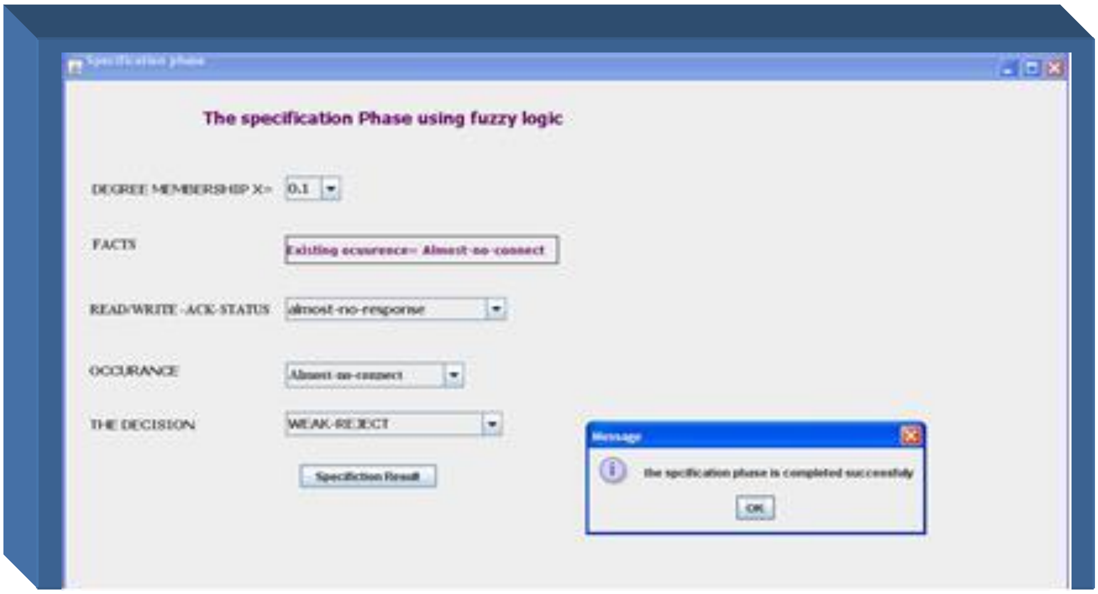

Fig.2 The GUI of Specification Phase Using Java Code

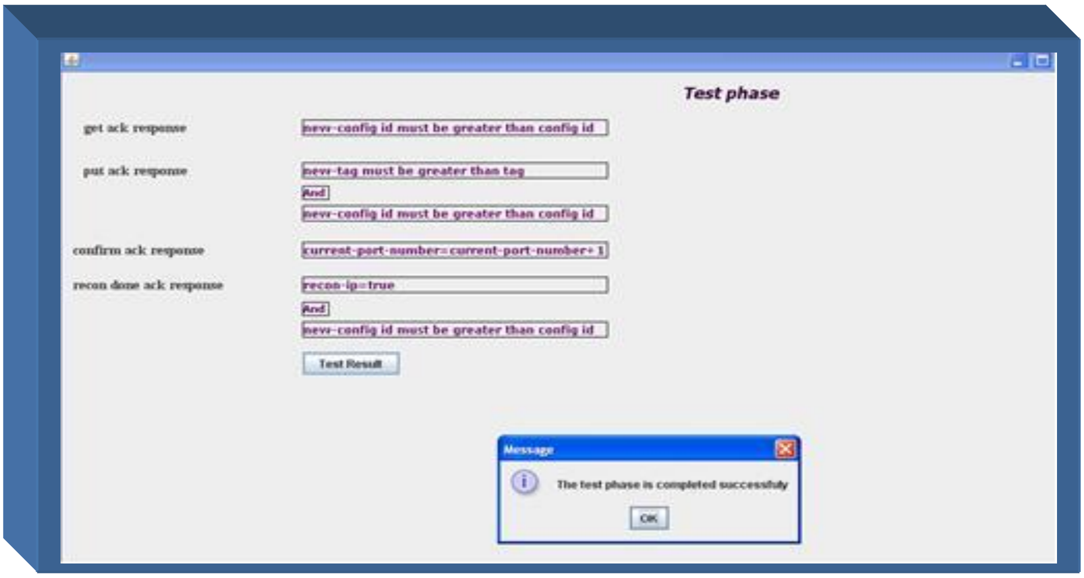

Fig. 3 The GUI of Design Phase Java Code

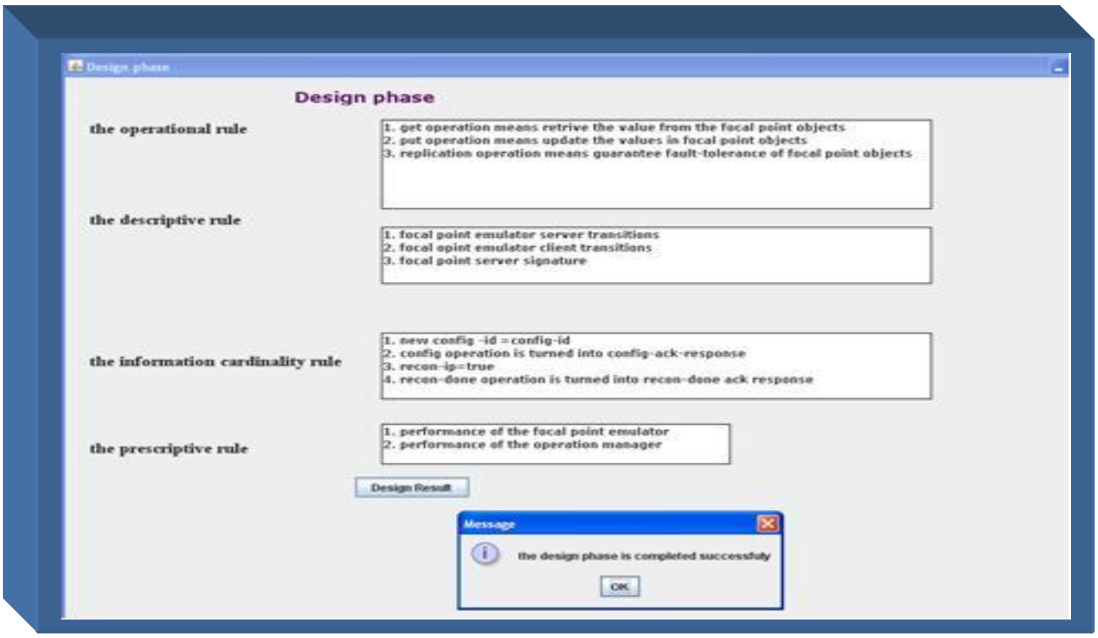

Fig.4 The GUI of Testing Phase Using Java Code

International Journal of Technology and Systems
ISSN 2518-881X (Online)
Vol.7, Issue 1, No.1, pp 1-19, 2022

4. Java Code for the Geoquorum Approach

The final code in for implementing atomic read/write shared memory mobile ad hoc network is done as follow:

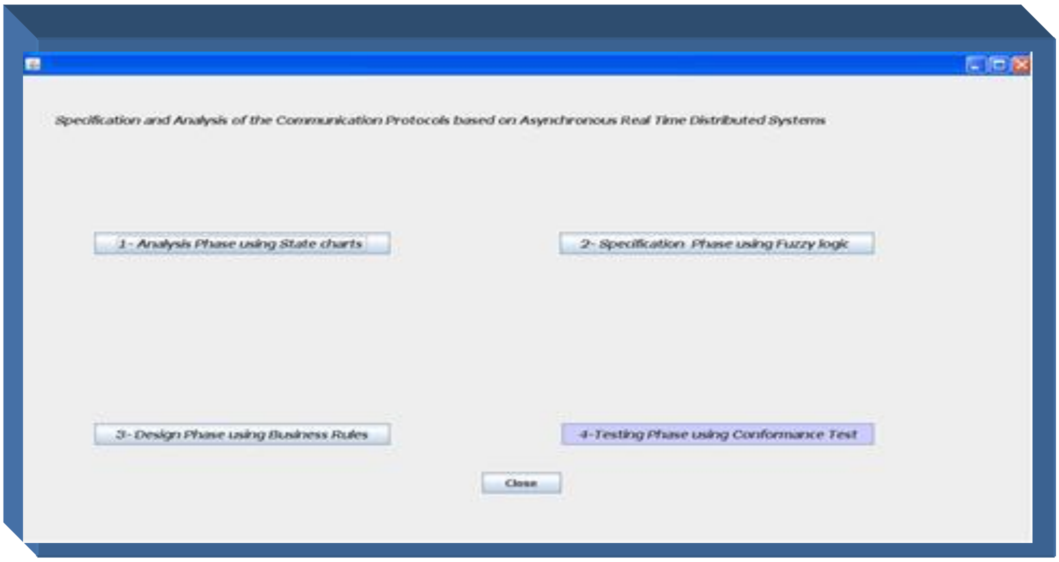

Fig.5 The GUI for The Communication Protocol Based on Asynchronous Real Time Distributed Systems in Java

International Journal of Technology and Systems
ISSN 2518-881X (Online)
Vol.7, Issue 1, No.1, pp 1-19, 2022


```
package       examples;
import java.awt.Frame;

import       java.io.IOException;
import   java.util.logging.Level;
import  java.util.logging.Logger;
import   javax.swing.JOptionPane;
/**
 * @Author Reham
 **/
public class NewJFrame extends javax.swing.JFrame {

    /** Creates new form NewJFrame */
    public NewJFrame() {
        initComponents();
    }
    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this
method is
     * always regenerated by the Form Editor.
     */ @SuppressWarnings("unchecked")

    // <editor-fold defaultstate="collapsed" desc=" Generated
    Code ">
    private void initComponents() {
        buttonGroup1 = new javax.swing.ButtonGroup();
        jLabel1 = new javax.swing.JLabel();
```

```
        jToggleButton1 = jToggleButton2 = jToggleButton3 =
        jToggleButton4 =
new             javax.swing.JToggleButton();             new
javax.swing.JToggleButton();                             new
javax.swing.JToggleButton();
new javax.swing.JToggleButton();

        jButton2 = new javax.swing.JButton();

addComponent(jToggleButton1,javax.swing.GroupLayout.DEFAULT_SIZE,
300,Short.MAX_VALUE).
addComponent(jToggleButton3,javax.swing.GroupLayout.Alignment.LEA
DING,                    javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE) .

addGap(173,173,173).

addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.TRAILING).
addComponent(jToggleButton2,javax.swing.GroupLayout.PREFERRED_SIZ
E,319,javax.swing.GroupLayout.PREFERRED_SIZE).
addComponent(jToggleButton4)).addGap(163,163,163)).

addGroup(layout.createSequentialGroup().

addGap(33, 33, 33).

addComponent(jLabel1,    javax.swing.GroupLayout.PREFERRED_SIZE,
797, javax.swing.GroupLayout.PREFERRED_SIZE).
addContainerGap(220,Short.MAX_VALUE)).addGroup(layout.createSeque
ntialGroup().addGap(467,      467,      467).addComponent(jButton2,
javax.swing.GroupLayout.PREFERRED_SIZE,                        80,
javax.swing.GroupLayout.PREFERRED_SIZE).addContainerGap(503,
Short.MAX_VALUE))  );
layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING).addGroup(layout.createSequentialGroup().addGap(26,26,26.addC
```

```
omponent(jLabel1,          javax.swing.GroupLayout.PREFERRED_SIZE,61,
javax.swing.GroupLayout.PREFERRED_SIZE).addGap(112,     112,     112)
.addGroup(layout.createPar 10     elGroup(javax.swing.GroupLayout.Alig
nment.BASELINE).addComponent(jToggleButton1.addComponent(jToggleB
utton2)).addGap(214,
214,214).addGroup(layout.createParallelGroup(javax.swing.GroupLay
out.Alignment.BASELINE).addComponent(jToggleButton3)

.addComponent(jToggleButton4))

.addGap(35,   35,   35).addComponent(jButton2).addContainerGap(88,
Short.MAX_VALUE))
  );

 pack();

 }// </editor-fold>

private                                                          void
jToggleButton3ActionPerformed(java.awt.event.ActionEvent evt) {

reham1  r1=new  reham1();
//r1.setSize(900,900);
r1.setVisible(true);

    }
private                                                          void
jToggleButton1ActionPerformed(java.awt.event.ActionEvent evt) {

NewJFrame1  f1=new  NewJFrame1();
//f1.setSize(900,900);
f1.setVisible(true);

    }
private                                                          void
jToggleButton4ActionPerformed(java.awt.event.ActionEvent evt) {

NewJFrame2  f2=new  NewJFrame2();
f2.setSize(900,900);
```

```
    //      f2.setState(Frame.MAXIMIZED_BOTH);
f2.setVisible(true);

    }
private                                                    void
jToggleButton2ActionPerformed(java.awt.event.ActionEvent  evt)  {
specification s=new specifi 11ion();

//s.setState(Frame.MAXIMIZED_BOTH);
s.setVisible(true);

    }

private  void  jButton2ActionPerformed(java.awt.event.ActionEvent
evt) {
  this.dispose();
      }
      /**

      * @param args the command line arguments
      */
      public   static   void   main(String   args[])   {
          java.awt.EventQueue.invokeLater(new Runnable() {
              public void run() {
                  new  NewJFrame().setVisible(true);
    }
      });
      }
      //  Variables  declaration  -  do  not  modify
      private  javax.swing.ButtonGroup  buttonGroup1;
      private     javax.swing.JButton     jButton2;
      private javax.swing.JLabel jLabel1;
      private javax.swing.JToggleButton jToggleButton1;
      private javax.swing.JToggleButton jToggleButton2;
      private javax.swing.JToggleButton jToggleButton3;
```

```
    private                    javax.swing.JToggleButton
    jToggleButton4; // End of variables declaration
}
```

Conclusions

In this paper Z specification as a formal specification language and a java code are constructed for the geoquorum approach which is considered a communication protocol based on asynchronous real time distributed systems. This code is a tool to guarantee the accuracy and the validation of some phases of software development lifecycle of this application such as analysis phase, specification phase, design and testing phases. All these phases are illustrated and building by java language.

References

[1] Barmade , M.M. Nashipudinath, "An efficient strategy to detect outlier transactions," International Journal of Soft Computing and Engineering (IJSCE), vol. 6, no. 174-178, p. 3, 2014.

[2] Z. He, X. Xu, J.Z. Huang, S. Deng, "Fp-outlier: frequent pattern based outlier detection," Computer Science and Information Systems, vol. 2, no. 1, pp. 103-118, 2017.

[3] F. Hendrikx, K. Bubendorfer, R. Chard, "Reputation systems: a survey and taxonomy," Journal of Parallel and Distributed Computing, vol. 75, pp. 184-197, 2015.

[4] Dobson, A. J., and A. G. Barnett. , An Introduction to Generalized Linear Models, Chapman and Hall/CRC. ,Taylor & Francis Group, 2018.

[5] A . Manna, A . Sengupta, C. Mazumdar, "A survey of trust models for enterprise information systems," Procedia Comput. Sci., vol. 85, p. 527– 534, 2016.

[6] R. Malaga, "Web-based reputation management systems: problems and suggested solutions," Electronic Commerce Research, p. 403–417, 2016.

[7] G. D'Angelo, S. Rampone, F. Palmieri, "An artificial intelligence-based trust model for pervasive computing," Proceedings of the 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), p. 701–706, 2015.

[8] G. D'Angelo, S. Rampone, F. Palmieri, "Developing a trust model for pervasive computing based on apriori association rules learning and Bayesian classification," Soft Comput., p. 6297–6315, 2017.

[9] J. Weng, C. Miao, A. Goh, "Protecting Online Rating Systems from Unfair Ratings," International Conference on Trust, Privacy and Securityin Digital Business, p. 50–59, 2015.

[10] C .Chiang,J.E.Urban, "Validating Software Specification against User Claims", Proceedings of the Twenty-Third Annual International Computer Software and Applications Conference _OMPSAC(2015),2015,PP:104-109.

[11] DOLEV, S., Gilbert, S.LYNCH, N.A., SHVARTSMAN, A.A., Welch, J.L.: " Geoquorums: Implementing Atomic Memory in Mobile Ad Hoc Networks". In: Proceeding of the 17th International Conference on Distributed Computing, PP. 306-320 (2019).

[12] Haas, Z.J., Liang, B.: "Ad Hoc Mobile Management with Uniform Quorum Systems". IEEE/ACM Transactions on Networking 7(2), PP: 228-240 (2018).

[13] T. Hara, "Location Management of Replication Considering Data Update in Ad Hoc Networks, in : The 20th International Conference AINA, 2016, PP. 753-758 .

[14] T. Hara, A. " Replication Management for Data Sharing In Mobile Ad Hoc Networks", Journal of The Interconnection Networks and communication 7(1) (2019), PP.75-90 .

[15] Y Sawai, M. Shinohara, A. Kanzaki, T. Hara, S. Nishio, " Ensuring Consistency Management Among Replicas Using A Quorum System in Ad Hoc Networks", MDM (2018) , PP.128-132.