Universidad de Costa Rica

Escuela de ciencias de la computación y la informática

Web Services

Fernando Mata Mora

4 de diciembre del 2016

Tabla de contenidos

1.Introducción

En el proyecto se nos pidió implementar un web service que permitiera jugar gato, además de una interfaz gráfica que permitiera jugar.

1.a.Descripción del problema

El propósito del proyecto era crear un servicio web con la lógica para poder llevar a cabo partidas de gato, ya fuera de forma individual como contra la máquina. En el problema la interfaz debía hacer llamados a métodos del servicio web encargado llevar cuenta de la lógica del programa. También dentro de las funcionalidades debía estar la opción de permitir al jugador poder consultar los 10 mejores resultados ordenados por cantidad de segundos.

2.Análisis del problema

Para resolver el problemas se determinó las funciones básicas con las cuales debía contar el programa para lograr llevar a cabo un juego de gato y a partir de ellas se programó la funcionalidad del gato en el servicio web. En la interfaz se programaron únicamente estructuras de control para reflejar el estado del juego que se mantenía en el servidor.

3.Casos de Prueba

Método	Valores de Prueba	Estado	Resultado esperado
newGame	player1=”J1”,player=”J2”	No hay un juego activo	Empieza un juego nuevo normal
newGame	player1=”J1”,player=””	No hay un juego activo	Empieza un juego nuevo contra la máquina.
makeMove	x=”1”,y=”1”,player=”J1”	Un juego iniciado y espacio 1,1 vacío.	Devuelve como resultado dentro del mensaje el valor “COOL” y reserva el espacio para J1.
makeMove	x=”1”,y=”1”,player=”J1”	Un juego iniciado y espacio 1,1 ocupado por el contrincante..	Devuelve como resultado dentro del mensaje el valor “NOT_COOL” y no reserva el espacio para J1.
makeMove	x=”-1”,y=”-1”,player=””	Un juego iniciado y modo solo un jugador.	El cliente solicita un movimiento por parte de la máquina y dentro del mensaje de retorno se encuentran las coordenadas x,y.
makeMove	x=”2”,y=”2”,player=”J1”	Un juego iniciado y las casillas 0,0 y 1,1 ocupadas por J1.	Devuelve como resultado dentro del mensaje el valor “COOL” y el estado del juego, en este caso “WON”.
newScore	player=”J1”,time=”4”	Un juego finalizado.	Agrega el resultado a la base de datos.
leaderboards		Un juego sin iniciar	Devuelve los 10 mejores resultados.

4.Resultados de los casos de prueba

Todos los casos de prueba se realizaron por medio del cliente y en todos los casos se dió el resultado esperado.

5.Código fuente con su respectiva documentación interna.

Client_UI.java

```
/*
```

```
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package soap_client;
import java.util.Date;
import javax.xml.ws.BindingProvider;
import javax.swing.JOptionPane;
import ecci_gato.*;
import java.util.*;
/**
 *
 * @author Fergo
 */
public class Client_UI extends javax.swing.JFrame {

    /**
     * Creates new form Client_UI
     */

    private ECCIGatoPort port;
    private String[][] board;
    private String user_active = "";
    private String user_inactive = "";
    private String jugador1_name;
    private String jugador2_name;
    private Date start_time;
    private boolean cont;

    public Client_UI() {
       initComponents();
       ECCIGato service = new ECCIGato();
       port = service.getECCIGatoPort();
        board = new String[3][3];

((BindingProvider)port).getRequestContext().put(BindingProvider.SESSION_MAINTAIN_PR
OPERTY,true);
    }
    //Metodo para preparar un nuevo juego
    private void set_game(){
        cont = false;
        start_button.setEnabled(true);
            scores_button.setEnabled(true);
            jugador1.setEnabled(true);
            jugador2.setEnabled(false);
            jugador1.setText("");
```

```
            jugador2.setText("");
            two_players_mode.setEnabled(true);

            board11.setEnabled(false);
            board12.setEnabled(false);
            board13.setEnabled(false);
            board21.setEnabled(false);
            board22.setEnabled(false);
            board23.setEnabled(false);
            board31.setEnabled(false);
            board32.setEnabled(false);
            board33.setEnabled(false);

            board11.setText("");
            board12.setText("");
            board13.setText("");
            board21.setText("");
            board22.setText("");
            board23.setText("");
            board31.setText("");
            board32.setText("");
            board33.setText("");
        }
        //Metodo que se encarga de iniciar un juego nuevo
        private void start_game(){
            String new_game_return = "";
            start_time = new Date();
            cont = true;
            try{
                new_game_return = port.newGame(jugador1_name, jugador2_name);
            }
            catch(Exception e){
                JOptionPane.showMessageDialog(this, e, "Error de conexion",
JOptionPane.ERROR_MESSAGE);
            }

            if("START".compareTo(new_game_return) != 0){
                JOptionPane.showMessageDialog(this, "Error al inicializar el juego", "Error",
JOptionPane.ERROR_MESSAGE);
            }
            else{
                start_time = new Date();
                start_button.setEnabled(false);
                scores_button.setEnabled(false);
                jugador1.setEnabled(false);
                jugador2.setEnabled(false);
```

```
        two_players_mode.setEnabled(false);

        board11.setEnabled(true);
        board12.setEnabled(true);
        board13.setEnabled(true);
        board21.setEnabled(true);
        board22.setEnabled(true);
        board23.setEnabled(true);
        board31.setEnabled(true);
        board32.setEnabled(true);
        board33.setEnabled(true);
    }
    user_active = jugador1_name;
    user_inactive = jugador2_name;
}
/**
 * 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() {

    jSeparator2 = new javax.swing.JSeparator();
    jPanel1 = new javax.swing.JPanel();
    jLabel1 = new javax.swing.JLabel();
    start_button = new javax.swing.JButton();
    scores_button = new javax.swing.JButton();
    jugador1 = new java.awt.TextField();
    jugador2 = new java.awt.TextField();
    jLabel2 = new javax.swing.JLabel();
    jLabel3 = new javax.swing.JLabel();
    two_players_mode = new javax.swing.JCheckBox();
    board12 = new javax.swing.JButton();
    board13 = new javax.swing.JButton();
    board11 = new javax.swing.JButton();
    board22 = new javax.swing.JButton();
    board23 = new javax.swing.JButton();
    board21 = new javax.swing.JButton();
    board32 = new javax.swing.JButton();
    board33 = new javax.swing.JButton();
    board31 = new javax.swing.JButton();

    setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
    setResizable(false);
```

```
jLabel1.setFont(new java.awt.Font("Tahoma", 0, 24)); // NOI18N
jLabel1.setText("GATO!");

start_button.setText("Iniciar Juego");
start_button.addMouseListener(new java.awt.event.MouseAdapter() {
    public void mouseClicked(java.awt.event.MouseEvent evt) {
        start_buttonMouseClicked(evt);
    }
});
start_button.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        start_buttonActionPerformed(evt);
    }
});

scores_button.setText("Puntajes Altos");
scores_button.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        scores_buttonActionPerformed(evt);
    }
});

jugador1.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jugador1ActionPerformed(evt);
    }
});

jugador2.setEnabled(false);
jugador2.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jugador2ActionPerformed(evt);
    }
});

jLabel2.setText("Jugador 1");

jLabel3.setText("Jugador 2");

two_players_mode.addChangeListener(new javax.swing.event.ChangeListener() {
    public void stateChanged(javax.swing.event.ChangeEvent evt) {
        two_players_modeStateChanged(evt);
    }
});
two_players_mode.addActionListener(new java.awt.event.ActionListener() {
```

```
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        two_players_modeActionPerformed(evt);
    }
});

board12.setToolTipText("");
board12.setEnabled(false);
board12.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        board12ActionPerformed(evt);
    }
});

board13.setEnabled(false);
board13.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        board13ActionPerformed(evt);
    }
});

board11.setEnabled(false);
board11.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        board11ActionPerformed(evt);
    }
});

board22.setEnabled(false);
board22.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        board22ActionPerformed(evt);
    }
});

board23.setEnabled(false);
board23.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        board23ActionPerformed(evt);
    }
});

board21.setEnabled(false);
board21.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        board21ActionPerformed(evt);
    }
```

```
        });

        board32.setEnabled(false);
        board32.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                board32ActionPerformed(evt);
            }
        });

        board33.setEnabled(false);
        board33.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                board33ActionPerformed(evt);
            }
        });

        board31.setEnabled(false);
        board31.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                board31ActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(
            jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addGap(45, 45, 45)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADIN
G)
                    .addGroup(jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADIN
G)
                            .addComponent(board11, javax.swing.GroupLayout.PREFERRED_SIZE,
107, javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(board21, javax.swing.GroupLayout.PREFERRED_SIZE,
107, javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(board31, javax.swing.GroupLayout.PREFERRED_SIZE,
107, javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
```

```
.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADIN
G)
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addComponent(board12,
javax.swing.GroupLayout.PREFERRED_SIZE, 107,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(board13,
javax.swing.GroupLayout.PREFERRED_SIZE, 107,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addComponent(board22,
javax.swing.GroupLayout.PREFERRED_SIZE, 107,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(board23,
javax.swing.GroupLayout.PREFERRED_SIZE, 107,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addComponent(board32,
javax.swing.GroupLayout.PREFERRED_SIZE, 107,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(board33,
javax.swing.GroupLayout.PREFERRED_SIZE, 107,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addGap(0, 0, Short.MAX_VALUE))
            .addGroup(jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADIN
G)
                    .addComponent(jugador1, javax.swing.GroupLayout.PREFERRED_SIZE,
164, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel2))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADIN
G)
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addComponent(jLabel3)
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                        .addComponent(two_players_mode))
                    .addComponent(jugador2, javax.swing.GroupLayout.PREFERRED_SIZE,
164, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(scores_button))
                .addGap(41, 41, 41))))
        .addGroup(jPanel1Layout.createSequentialGroup()

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADIN
G)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addGap(94, 94, 94)
                .addComponent(start_button))
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addGap(166, 166, 166)
                .addComponent(jLabel1)))
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
    );
    jPanel1Layout.setVerticalGroup(
        jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel1Layout.createSequentialGroup()
            .addGap(20, 20, 20)
            .addComponent(jLabel1)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILIN
G)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADIN
G)
                .addComponent(jLabel3, javax.swing.GroupLayout.Alignment.TRAILING)
                .addComponent(jLabel2))
            .addComponent(two_players_mode))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADIN
G)
            .addComponent(jugador2, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addComponent(jugador1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(24, 24, 24)
```

```
.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELI
NE)
                    .addComponent(start_button)
                    .addComponent(scores_button))))
            .addGap(41, 41, 41)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELI
NE)
                .addComponent(board12, javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(board13, javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(board11, javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELI
NE)
                .addComponent(board21, javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(board22, javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(board23, javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELI
NE)
                .addComponent(board32, javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(board33, javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(board31, javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
            .addContainerGap(62, Short.MAX_VALUE))
        );

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE, 409,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(0, 11, Short.MAX_VALUE))
```

```
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        pack();
    }// </editor-fold>

    //Metodo del boton IniciarJuego que se encarga de crear una nueva partida
    private void start_buttonActionPerformed(java.awt.event.ActionEvent evt) {
        jugador1_name = jugador1.getText();
        jugador2_name = jugador2.getText();
        if("".compareTo(jugador1_name) != 0 && jugador1_name.compareTo(jugador2_name)
!= 0){
            if(!two_players_mode.isSelected()){
                jugador2_name = "";
                start_game();
            }
            else{
                if("".compareTo(jugador2_name) == 0){
                    JOptionPane.showMessageDialog(this, "Por favor escriba un nombre para el
Jugador 2", "Error", JOptionPane.ERROR_MESSAGE);
                }
                else{
                    start_game();
                }
            }
        }
    }
    //Metodo para intercambiar usuarios activos
    private void switch_user(){
        String user = user_active;
        user_active = user_inactive;
        user_inactive = user;
    }
    //Metodo que se conecta con el web service para realizar un nuevo movimiento
    private void makemove(int x, int y){
        String str = port.makeMove(x, y, user_active);
        String[] status = str.split(":");
        String match_state;
```

```
        if("".compareTo(user_active) == 0){
            invalidate_button(Integer.parseInt(status[0]),Integer.parseInt(status[1]),"COMPU");
            match_state = status[2];
        }
        else{
            if("COOL".compareTo(status[0]) == 0){
                invalidate_button(x,y,user_active);
            }
            match_state = status[1];
        }
        if("WON".compareTo(match_state) == 0){
            cont=false;
            if("".compareTo(user_active) != 0 ){
                Date end_time = new Date();
                long time_delta = end_time.getTime() - start_time.getTime();
                int time = (int)(time_delta/1000);
                JOptionPane.showMessageDialog(this, "Ganador: "+user_active +"Tiempo:
"+time+"s", "FIN DEL JUEGO", JOptionPane.INFORMATION_MESSAGE);
                port.newScore(user_active, time);

            }
            else{
                JOptionPane.showMessageDialog(this, "PERDEDOR","FIN DEL JUEGO" ,
JOptionPane.INFORMATION_MESSAGE);
            }
            set_game();
        }
       if("TIE".compareTo(match_state) == 0){
            JOptionPane.showMessageDialog(this, "FIN DEL JUEGO", "Ganador: "+user_active,
JOptionPane.INFORMATION_MESSAGE);
            set_game();
        }
       if("NORMAL".compareTo(match_state) == 0){
           switch_user();
       }

       if("".compareTo(user_active) == 0 && cont){
           int sx = -1;
           int sy = -1;
           makemove(sx,sy);
       }
    }

    //Metodo que se encarga de invalidar uno de los botones ya selecionados cuando se
hace una movida
    private void invalidate_button(int x, int y,String name){
```

```
    if(x == 0){
        if(y == 0){
            board11.setEnabled(false);
            board11.setText(name);
        }
        if(y == 1){
            board12.setEnabled(false);
            board12.setText(name);
        }
        if(y == 2){
            board13.setEnabled(false);
            board13.setText(name);
        }
    }
    if(x == 1){
        if(y == 0){
            board21.setEnabled(false);
            board21.setText(name);
        }
        if(y == 1){
            board22.setEnabled(false);
            board22.setText(name);
        }
        if(y == 2){
            board23.setEnabled(false);
            board23.setText(name);
        }
    }
    if(x == 2){
        if(y == 0){
            board31.setEnabled(false);
            board31.setText(name);
        }
        if(y == 1){
            board32.setEnabled(false);
            board32.setText(name);
        }
        if(y == 2){
            board33.setEnabled(false);
            board33.setText(name);
        }
    }
}
//Boton que se encarga de desplegar los puntajes en pantalla
private void scores_buttonActionPerformed(java.awt.event.ActionEvent evt) {
    String scores = port.leaderboards();
```

```
        String[] scores_split = scores.split(",");
        String resultados = "Nombre Puntaje \n";
        for(int i = 0; i<scores_split.length; ++i){
            String[] element = scores_split[i].split(":");
            resultados= resultados + element[0] +" "+ element[1]+ "s"+ "\n";
        }
        JOptionPane.showMessageDialog(this, resultados, "Resultados",
JOptionPane.PLAIN_MESSAGE);
    }

    private void jugador1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void jugador2ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void two_players_modeActionPerformed(java.awt.event.ActionEvent evt) {
        if(two_players_mode.isSelected()){
            jugador2.setEnabled(true);
        }
        else{
            jugador2.setEnabled(false);
            jugador2.setText("");
        }
    }
    //Cada boton del tablero llama al metodo makemove de con sus respectivas coordenadas
    private void board11ActionPerformed(java.awt.event.ActionEvent evt) {
        makemove(0,0);
    }

    private void board12ActionPerformed(java.awt.event.ActionEvent evt) {
       makemove(0,1);
    }

    private void two_players_modeStateChanged(javax.swing.event.ChangeEvent evt) {

    }

    private void start_buttonMouseClicked(java.awt.event.MouseEvent evt) {

    }

    private void board21ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
```

```
        makemove(1,0);
    }

    private void board13ActionPerformed(java.awt.event.ActionEvent evt) {
        makemove(0,2);
    }

    private void board22ActionPerformed(java.awt.event.ActionEvent evt) {
        makemove(1,1);
    }

    private void board23ActionPerformed(java.awt.event.ActionEvent evt) {
        makemove(1,2);
    }

    private void board31ActionPerformed(java.awt.event.ActionEvent evt) {
        makemove(2,0);
    }

    private void board32ActionPerformed(java.awt.event.ActionEvent evt) {
        makemove(2,1);
    }

    private void board33ActionPerformed(java.awt.event.ActionEvent evt) {
        makemove(2,2);
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and
feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
```

```
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Client_UI.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Client_UI.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Client_UI.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Client_UI.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new Client_UI().setVisible(true);
            }
        });
    }



    // Variables declaration - do not modify
    private javax.swing.JButton board11;
    private javax.swing.JButton board12;
    private javax.swing.JButton board13;
    private javax.swing.JButton board21;
    private javax.swing.JButton board22;
    private javax.swing.JButton board23;
    private javax.swing.JButton board31;
    private javax.swing.JButton board32;
    private javax.swing.JButton board33;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JSeparator jSeparator2;
    private java.awt.TextField jugador1;
    private java.awt.TextField jugador2;
```

```
    private javax.swing.JButton scores_button;
    private javax.swing.JButton start_button;
    private javax.swing.JCheckBox two_players_mode;
    // End of variables declaration
}
```

gato.wsdl

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- Descripción WSDL de la clase Hola Mundo. -->

<definitions name="ECCI_Gato"
             targetNamespace="urn:ECCI_Gato"
             xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
             xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
             xmlns:tns="urn:ECCI_Gato"
             xmlns:xsd="http://www.w3.org/2001/XMLSchema"
             xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
             xmlns="http://schemas.xmlsoap.org/wsdl/">

  <!-- Tipos complejos -->

  <types xmlns="http://schemas.xmlsoap.org/wsdl/">
    <xsd:schema targetNamespace="urn:ECCI_Gato">

      <xsd:element name="newGame">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="player1" type="xsd:string" />
            <xsd:element name="player2" type="xsd:string" nillable="true"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>

      <xsd:element name="newGameReturn">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="newGameResult" type="xsd:string" />
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>

      <xsd:element name="newScore">
        <xsd:complexType>
          <xsd:sequence>
```

```
      <xsd:element name="player" type="xsd:string" />
      <xsd:element name="time" type="xsd:int"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>

<xsd:element name="newScoreReturn">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="newScoreResult" type="xsd:string" />
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>

<xsd:element name="leaderboards">
  <xsd:complexType >
        <xsd:sequence/>
  </xsd:complexType>
</xsd:element>

<xsd:element name="returnLeaderboards">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="leaderboardsResult" type="xsd:string" />
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>


<xsd:element name="makeMove">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="x" type="xsd:int"/>
      <xsd:element name="y" type="xsd:int"/>
      <xsd:element name="player" type="xsd:string" />
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>

<xsd:element name="makeMoveReturn">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="makeMoveResult" type="xsd:string" />
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

```
  </xsd:schema>
</types>

<!-- Mensajes para comunicarse con la clase HolaMundo. -->

<message name="newGameRequest">
  <part name="parameters" element="tns:newGame" />
</message>

<message name="newGameResponse">
  <part name="parameters" element="tns:newGameReturn" />
</message>

<message name="newScoreRequest">
  <part name="parameters" element="tns:newScore" />
</message>

<message name="newScoreResponse">
  <part name="parameters" element="tns:newScoreReturn" />
</message>

<message name="leaderboardsRequest">
  <part name="parameters" element="tns:leaderboards" />
</message>

<message name="leaderboardsResponse">
  <part name="parameters" element="tns:returnLeaderboards" />
</message>


<message name="makeMoveRequest">
  <part name="parameters" element="tns:makeMove" />
</message>

<message name="makeMoveResponse">
  <part name="parameters" element="tns:makeMoveReturn" />
</message>

<!-- Puerto para comunicar con la clase HolaMundo, "ECCI_Gato" -->

<portType name="ECCI_GatoPort">

  <operation name="newGame">
    <input message="tns:newGameRequest" />
    <output message="tns:newGameResponse" />
```

```
  </operation>

  <operation name="leaderboards">
    <input message="tns:leaderboardsRequest" />
    <output message="tns:leaderboardsResponse" />
  </operation>

  <operation name="newScore">
    <input message="tns:newScoreRequest" />
    <output message="tns:newScoreResponse" />
  </operation>

  <operation name="makeMove">
    <input message="tns:makeMoveRequest" />
    <output message="tns:makeMoveResponse" />
  </operation>

</portType>

<!-- Vinculación de los llamados con el transporte - Document, SOAP/Literal over HTTP -->

<binding name="ECCI_GatoBinding" type="tns:ECCI_GatoPort">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />

  <operation name="newGame">
    <soap:operation soapAction="urn:ECCI_Gato#Gato#newGame" style="document" />
    <input>
      <soap:body use="literal" />
    </input>
    <output>
      <soap:body use="literal" />
    </output>
  </operation>

  <operation name="makeMove">
    <soap:operation soapAction="urn:ECCI_Gato#Gato#makeMove" style="document" />
    <input>
      <soap:body use="literal" />
    </input>
    <output>
      <soap:body use="literal" />
    </output>
  </operation>

  <operation name="newScore">
    <soap:operation soapAction="urn:ECCI_Gato#Gato#newScore" style="document" />
```

```
      <input>
        <soap:body use="literal" />
      </input>
      <output>
        <soap:body use="literal" />
      </output>
    </operation>

    <operation name="leaderboards">
      <soap:operation soapAction="urn:ECCI_Gato#Gato#leaderboards" style="document" />
      <input>
        <soap:body use="literal" />
      </input>
      <output>
        <soap:body use="literal" />
      </output>
    </operation>

  </binding>

  <!-- Punto de comunicación con la clase HolaMundo -->

  <service name="ECCI_Gato">
    <documentation />
    <port name="ECCI_GatoPort" binding="tns:ECCI_GatoBinding">
      <soap:address location="http://titanic.ecci.ucr.ac.cr:80/~eb23990/PlayGato/" />
    </port>
  </service>

</definitions>
```

gato.class.php

```
<?php
ini_set('display_errors', 1);
ini_set('display_startup_errors', 1);
error_reporting(E_ALL);
/**
 *
 * Copyright (c) 2005-2015, Braulio José Solano Rojas
 * All rights reserved.
 *
 * Redistribution && use in source && binary forms, with or without modification, are
 * permitted provided that the following conditions are met:
 *
```

```
 *      Redistributions of source code must retain the above copyright notice, this list of
 *      conditions && the following disclaimer.
 *      Redistributions in binary form must reproduce the above copyright notice, this list of
 *      conditions && the following disclaimer in the documentation &&/or other materials
 *      provided with the distribution.
 *      Neither the name of the Solsoft de Costa Rica S.A. nor the names of its contributors
may
 *      be used to endorse or promote products derived from this software without specific
 *      prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS &&
 * CONTRIBUTORS "AS IS" && ANY EXPRESS OR IMPLIED WARRANTIES,
 * INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF
 * MERCHANTABILITY && FITNESS FOR A PARTICULAR PURPOSE ARE
 * DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR
 * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 * SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT
 * NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
 * LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED && ON ANY THEORY OF LIABILITY, WHETHER IN
 * CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
 * OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE,
 * EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 *
 *
 * @version $Id$
 * @copyright 2005-2015
 */



/**
 * HolaMundo Clase que implementa el típico primer ejemplo de programación en todo
lenguaje.
 *
 * @package SoapDiscovery
 * @author Braulio José Solano Rojas
 * @copyright Copyright (c) 2005-2015 Braulio José Solano Rojas
 * @version $Id$
 * @access public
 **/
class Gato {
        private $jugador1 = '';
        private $jugador2 = '';
        public  $game = "";
        private $gato = NULL;
        private $leaderboards;
```

```
        /**
         * HolaMundo::__construct() Constructor de la clase HolaMundo.
         *
         * @param string $nombre
         * @return string
         **/
        public function __construct() {
        }

        /**
         *Metodo que iniciliza un nuevo juego
         *
         * @param string $player1
  * @param string $player2
         * @return string
         **/
        public function newGame($player1,$player2) {
                if(strcmp($player2,"") != 0){
                        $this->jugador2=$player2;
                }
                else{
                        $this->jugador2="COMPU";
                }
                $this->jugador1=$player1;
   $this->gato = array(array("","",""),array("","",""),array("","",""));
   return "START";
        }

        /**
         * Metodo que se encarga de llevar a cabo un movimiento en el tablero y devuelve el
resultado
         *
         * @return string
         **/
        public function makeMove($x,$y,$player) {
                if(strcmp($player,"") == 0){
                        $player = "COMPU";
                }
                $response = '';
                $matriz = "";
                foreach($this->gato as $row){
                        $matriz = $matriz."[";
                        foreach($row as $e){
                                        $matriz = $matriz.$e.",";
                        }
                        $matriz = $matriz."],";
```

```
                }
                error_log($matriz);
                error_log($player);
                error_log("x:".$x);
                error_log("y:".$y);
                if($x < 0 && $y < 0){
                        $response = $this->makeMachineMove();
                }
                else{
                        if(strcmp($this->gato[$x][$y],'') == 0){
        $this->gato[$x][$y] = $player;
        $response = 'COOL';
        }
                        else{
                                $response = 'NOT_COOL';
                        }
                }
                $gato_state =  $this->won($player);
                if(strcmp($gato_state,'WON') == 0){
                        $response= $response.":WON";
                }
                else{
                        $getAvaibleSpaces = count($this->getAvaibleSpaces());
                        if($getAvaibleSpaces == 0){
                                $response = $response.":TIE";
                        }
                        else{
                                $response = $response.":NORMAL";
                        }
                }
                return $response;
        }
        /**
        *Metodo que se encarga de determinar si un jugador ganó.
        **/

        private function won($player){
                if(
                (strcmp($this->gato[0][0],$player) == 0 && strcmp($this->gato[0][1],$player)
== 0 && strcmp($this->gato[0][2],$player) == 0) ||
                (strcmp($this->gato[1][0],$player) == 0 && strcmp($this->gato[1][1],$player)
== 0 && strcmp($this->gato[1][2],$player) == 0) ||
                (strcmp($this->gato[2][0],$player) == 0 && strcmp($this->gato[2][1],$player)
== 0 && strcmp($this->gato[2][2],$player) == 0) ||
                (strcmp($this->gato[0][0],$player) == 0 && strcmp($this->gato[1][0],$player)
== 0 && strcmp($this->gato[2][0],$player) == 0) ||
```

```
                (strcmp($this->gato[0][1],$player) == 0 && strcmp($this->gato[1][1],$player)
== 0 && strcmp($this->gato[2][1],$player) == 0) ||
                (strcmp($this->gato[0][2],$player) == 0 && strcmp($this->gato[1][2],$player)
== 0 && strcmp($this->gato[2][2],$player) == 0) ||
                (strcmp($this->gato[0][0],$player) == 0 && strcmp($this->gato[1][1],$player)
== 0 && strcmp($this->gato[2][2],$player) == 0) ||
                (strcmp($this->gato[0][2],$player) == 0 && strcmp($this->gato[1][1],$player)
== 0 && strcmp($this->gato[2][0],$player) == 0)
                ){
                return "WON";
                }
                else{
                        return "NOT_WON";
                }
        }
        /**
        *Metodo que devuelve los espacio disponibles.
        **/
        private function getAvaibleSpaces(){
                $spaces = array();
                for ($i = 0;$i < count($this->gato[0]);++$i) {
                        for ($j = 0;$j < count($this->gato[0]);++$j) {
                                if(strcmp($this->gato[$i][$j],'') == 0){
                                        //error_log("Free: ".$i.",".$j);
                                        array_push($spaces, array($i,$j));
                                }
                        }
                }
                return $spaces;
        }
        /**
        *Metodo que se encarga de determinar el siguiente movimiento de la maquina.
        **/
        private function makeMachineMove(){
                $spaces = $this->getAvaibleSpaces();
          $space = $spaces[rand()%count($spaces)];
                $this->gato[$space[0]][$space[1]] = $this->jugador2;
                return $space[0].":".$space[1];
        }
        /**
        *Metodo que agrega un nuevo score a la BD
        **/
        public function newScore($player,$time)
        {
                $newScore = new Scores();
                $newScore->nickname = $player;
```

```
                $time_split = explode(":", $time);
                error_log($time);
                $newScore->score = $time;
                $newScore->save();
        }
        /**
         * Metodo que devuelve los resultados de las diferentes partidas
         *
         * @return string
         **/
        public function leaderboards()
        {
                $table = 'eb23990_scores';
                $where = 'score > 0 ORDER BY score';
                $activeRecords = $GLOBALS['db']->GetActiveRecords($table,$where);
                $result = '';
                $i = 0;
                foreach ($activeRecords as $record) {
                        if($i < 10){
                                $result = $result.$record->nickname .':'.$record->score.",";
                                ++$i;
                        }
                }
                return $result;
        }

}




?>
```

6.Bibliografía

PHP documentation
WSDL reference