Software de Comunicaciones
I.T.T. Especialidad Telemática
Escuela Politécnica Superior
Universidad Carlos III de Madrid

Duración de los problemas: 1 hora 30 minutos.
Duración total del examen: 2 horas 30 minutos.

NORMAS:
No se permiten ni libros ni apuntes durante el examen.

Nombre: .....
Apellidos: .....

Problemas (5 puntos)

Problema 1: Cotizaciones (3 puntos)

La empresa *PROFIT S.A.* lo ha contratado para realizar una aplicación empresarial que permita gestionar de forma eficiente las cotizaciones en la bolsa, denominada *Market Manager*. Para ello, el primer paso es desarrollar un componente (bean de sesión sin estado llamado `cotizacionEJB`) con las siguientes características:

1. Posee un único método de negocio con la siguiente estructura:

   `long obtenerCotizacion(Date fecha)`. Este método devuelve el valor de cotización de una acción determinada en el momento indicado, el cual se pasa como parámetro (fecha).

2. Utiliza un EJB llamado `HistorialAccion` del que sabe que:

   a) Tiene un `Date` por clave primaria, la cual es compatible con la del método `obtenerCotizacion`.

   b) Posee un método de negocio `long valor()` que devuelve la cotización de dicha acción en la fecha indicada.

Teniendo toda esta información en cuenta **se le pide** que:

1. (0,5 ptos) Defina la vista cliente de `cotizacionEJB`.

   Solución:

```
//Interfaz home
public interface CotizacionHome extends EJBHome{
  public CotizacionRemote create() throws RemoteException, CreateException;
}
```

```
//Interfaz remote
public interface CotizacionRemote extends EJBObject{
  public long obtenerCotizacion(Date fecha) throws RemoteException;
}
```

2. (1,5 ptos) Diseñe la clase principal que implemente dicha vista cliente.

   Solución:

```
public class  CotizacionBean implements SessionBean {
  //Métodos de la plantilla del session bean.
  public void ejbCreate()  throws CreateException {};
  void ejbRemove(){};
  void ejbPassivate(){};
  void ejbActivate(){};
  void setSessionContext (Session Context ctx){};

 //método de negocio
  public long obtenerCotizacion(Date fecha) {
    HistorialAccionRemote har = null;
    try {
      Context ctx = new InitialContext();
      Object obj =ctx.lookup("java:comp/env/ejb/HistorialAccionHome");
      HistorialAccionHome home=(HistorialAccion)
                          PortableRemoteObject.narrow(obj, HistorialAccionHome.class);
      har = home.findByPrimaryKey(fecha);
    }  catch(FinderException e){ //do something  }
    return har.valor();
   }
}//fin de la clase
```

3. (0,7 ptos) Esboce el descriptor XML de dicha vista cliente.

   Solución:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE ejb-jar PUBLIC "-//Sun Microsystems, Inc.//DTD Enterprise JavaBeans 2.0//EN"
   "http://java.sun.com/dtd/ejb-jar_2_0.dtd">
<ejb-jar>
 <enterprise-beans>
 <session>
   <ejb-name>CotizacionEJB</ejb-name>
   <home>swc.CotizacionHome</home>
   <remote>swc.CotizacionRemote</remote>
   <ejb-class>swc.CotizacionBean</ejb-class>
   <session-type>Stateless</session-type>
   <transaction-type>Container</transaction-type>
   <ejb-ref>
     <ejb-ref-name>ejb/HistorialAcccionHome</ejb-ref-name>
     <ejb-ref-type>Entity</ejb-ref-type>
     <home>swc.HistorialAccionHome</home>
     <remote>swc.HistorialAccionRemote</remote>
   </ejb-ref>
   <security-identity><use-caller-identity/></security-identity>
```

```
 </session>
 </enterprise-beans>
 <assembly-descriptor> ... </assembly-descriptor>
</ejb-jar>
```

4. (0,3 ptos) Describa la tabla mínima de la base de datos necesaria para dar persistencia a `HistorialAccion`.

   Solución:

   ```
   Campo  -   tipo
   fecha  -  Date (PK)
   valor  -   long
   ```

Problema 2: AuthzMIDP (2 puntos)

Para que los gestores de bolsa puedan acceder desde su móvil a la aplicación empresarial *Market Manager* requieren una autorización previa, por tanto, **se le pide que desarrolle la aplicación MIDP** con la siguiente funcionalidad (1,7 puntos):

- Al abrir la aplicación se debe solicitar el código de acceso único para cada gestor y enviarlo al servidor para su validación. La URL del servidor se obtiene como atributo del MIDlet (`getAppProperty('servidor')`).
- Si el código es correcto, se le muestra al usuario una imagen de bienvenida utilizando la API de bajo nivel. Si no es correcto, entonces se termina la aplicación.
- Cuando la autorización es exitosa, el usuario podrá salir de la aplicación pulsando la tecla 5 (o acción `FIRE`).
- Antes de finalizar la aplicación, se debe almacenar de forma persistente la fecha (`new Date()`) en que se accedió o se intentó acceder, para asuntos de auditoría.

Solución:

```
public class AuthnMIDP extends MIDlet implements CommandListener {
  private Display display;
  static final Command ok = new Command("Enviar", Command.OK, 0);
  private Canvas micanvas;
  private TextBox main = new TextBox("Introduzca código de acceso:", "", 10, TextField.PASSWORD);

  public AuthnMIDP () {
     display=Display.getDisplay(this);
      main.addCommand(ok);
      main.setCommandListener(this);
   } //fin del constructor

   protected void startApp(  ) {
      display.setCurrent(main);
   }

   protected void pauseApp(  ) {}
```

```
 protected void destroyApp(boolean incondicional) {
   display = null;
   main = null;
   micanvas = null;
   ok = null;
  }

 public void commandAction(Command c, Displayable d) {
   if (c == ok) {
    Thread hilo = new Thread() {
      public void run() {
        verificarCodigo();
       }
     };
     hilo.start();
   }
 }

public void verificarCodigo() {
    InputStream is = null;
    OutputStream os = null;
     HttpsConnection c = null;
     try {
        int ch = 0;
        c = (HttpsConnection)Connector.open(getAppProperty(servidor));
        c.setRequestMethod(HttpConnection.POST);
        c.setRequestProperty("User-Agent", "Profile/MIDP-2.0 Configuration/CLDC-1.1");
         if ((main.getString()!=null) || (main.getString()!="")) {
          String formData = "token="+main.getString();
           byte[] data = formData.getBytes();
           c.setRequestProperty("Content-Length", Integer.toString(data.length));
           os = c.openOutputStream();
           os.write(data);
           os.flush();
           is = c.openInputStream();
           if ((ch = is.read()) != -1) {
             if (ch==0) { //Creamos pantalla de authn exitosa
               micanvas = new Canvas(this) {
                 public void paint (Graphics g){
                   g.setColor(255,255,255);
                   g.fillRect(0,0,getWidth(),getHeight());
                   try {
                     Image bienvenida = new Image("/welcome.png");
                      if (bienvenida != null)
                        g.drawImage(bienvenida,width/2,height/2,0);
                    } catch (Exception ex) { //do something  }
                  }//fin método paint

                  public void keyPressed(int keyCode) {
                    if (keyCode==NUM_5) {
                      this.destroyApp(true);
                      this.notifyDestroyed();
                     }
```

```
                    } //fin del método keyPressed
                  };
                  display.setCurrent(micanvas);
                } else {
                  destroyApp(true);
                  notifyDestroyed();
                }
              }
      } finally {
        if (is != null) is.close();
        if (os != null) os.close();
        if (c != null) c.close();
      }  catch (IOException ioe) { ioe.printStackTrace(); }
  }
}
```

Además, **debe escribir el descriptor de la aplicación**, teniendo en cuenta el atributo definido por el desarrollador (0,3 puntos).

Solución:

```
MIDlet-Name: AuthnMIDP
MIDlet-Version: 1.0
MIDlet-Vendor: swc
MIDlet-Jar-URL: authen.jar
MIDlet-Jar-Size: 100KB
servidor:http://www.it.uc3m.es/authnserver
```