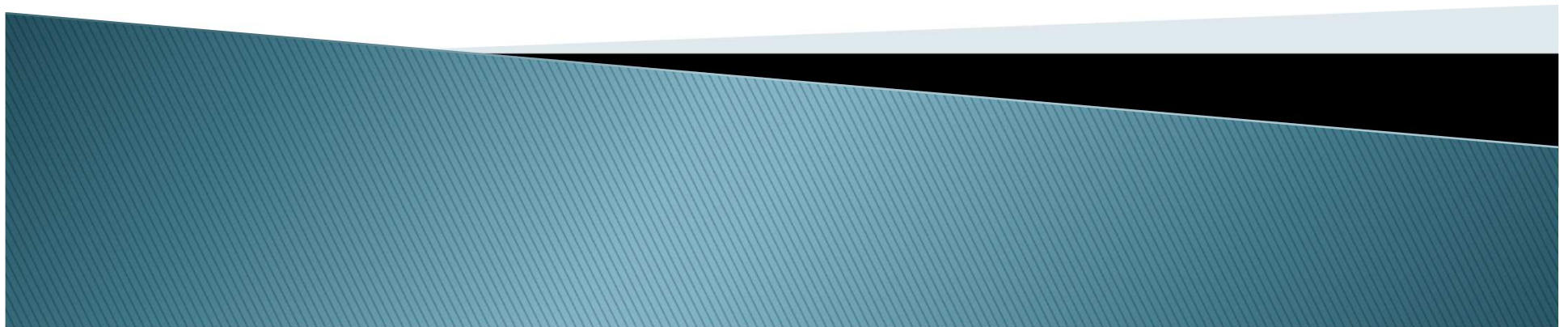


3 MAILAKO ARKITEKTURA

SOFTWARE INGENIARITZA



EDUKIAK

- ▶ Helburua
- ▶ Motibazioa
- ▶ Proposamena
- ▶ Maila bakarreko aplikazioak
- ▶ Bezero-zerbitzari arkitektura
- ▶ 3 mailako aplikazioak



Definizioa

- ▶ Softwareren Arkitekturak maila altuko software egituren diseinu eta inplementazioarekin erlazioa dauka.
- ▶ Sistema baten funtzionalitatea eta jarduera irizpideak betetzeko osagai arkitektonikoen egituraketaren emaitza da.
- ▶ Funtzionalak ez diren irizpideak betetzea ere ahalbidetzen du: fidagarritasuna, eskalagarritasuna, eramangarritasuna eta erabilgarritasuna.



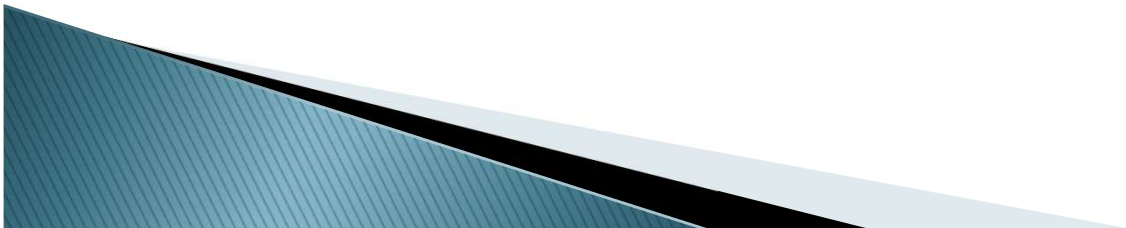
Definizioa

- ▶ Softwareren arkitekturak, modu abstraktu baten, konputazio atazaren bat burutzen duten osagaiak, beraien interfazak eta elkarreko komunikazioa definitzen ditu.
- ▶ Edozein softwareren arkitekturak arkitektura fisiko baten gainean inplementagarria eza behar da, hau da, ataza bakoitza zein konputagailua burutuko den ezarri behar da.



Motibazioa

- ▶ Garapenaren partaideen arteko komunikazioa erraztu.
- ▶ Ondoren jarraitzen duen garapen prozesuan eragin handia izan ahal duten diseinu erabakiak azpimarratzen ditu.
- ▶ Sistema nola egituratzen den eta bere osagaiek elkarrekin nola egiten duten lan erakusten du.
- ▶ Aplikazio malgu, eskalagarri eta berrerabilgarrien garapena erraztu.



Proposamena

Ez da softwarearen arkitektura berri bat sortu behar eraiki nahi den sistema bakoitzarentzat. Ohikoa, ezaguna den eta bere abantaila edo eragozpenen arabera kasu konkretu bakoitzari hobeena moldatzen den arkitektura bat aukeratzea da.



Proposamena

Arkitekturarik unibertsalenak ondorengoak dira:

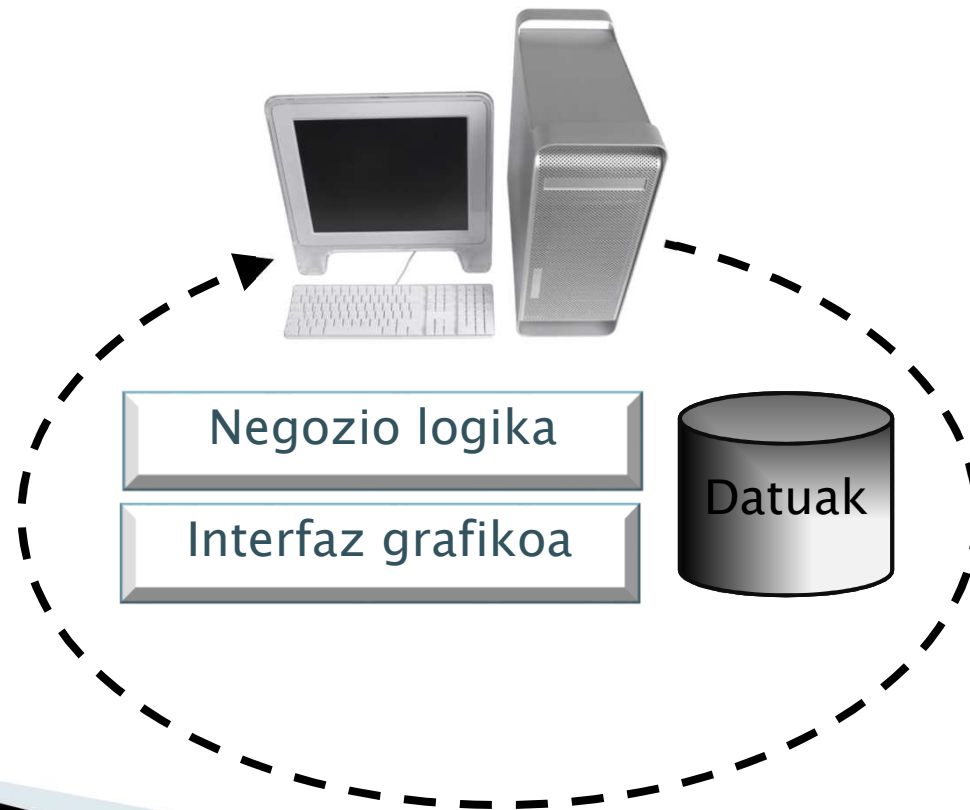
- ▶ Maila bakarrekoak: softwarea akloplamendu oso sendoa duten multzo funtzionaletan egituratzen da.
- ▶ Bezero-zerbitzaria: softwarea bi zati independientetan banatzen da.
- ▶ Hiru maila: softwarea hiru maila ezberdinetan banatzen da.

Aplikazioa mailetan banatzeak guztiz berriro idatzi behar gabe garatzaileek zati batzuk aldatzea posible egiten du.



Maila bakarreko aplikazioak

- ▶ Bai aplikazio datuak, bai negozio logika eta baita interfaz grafikoa bateratuta daude.



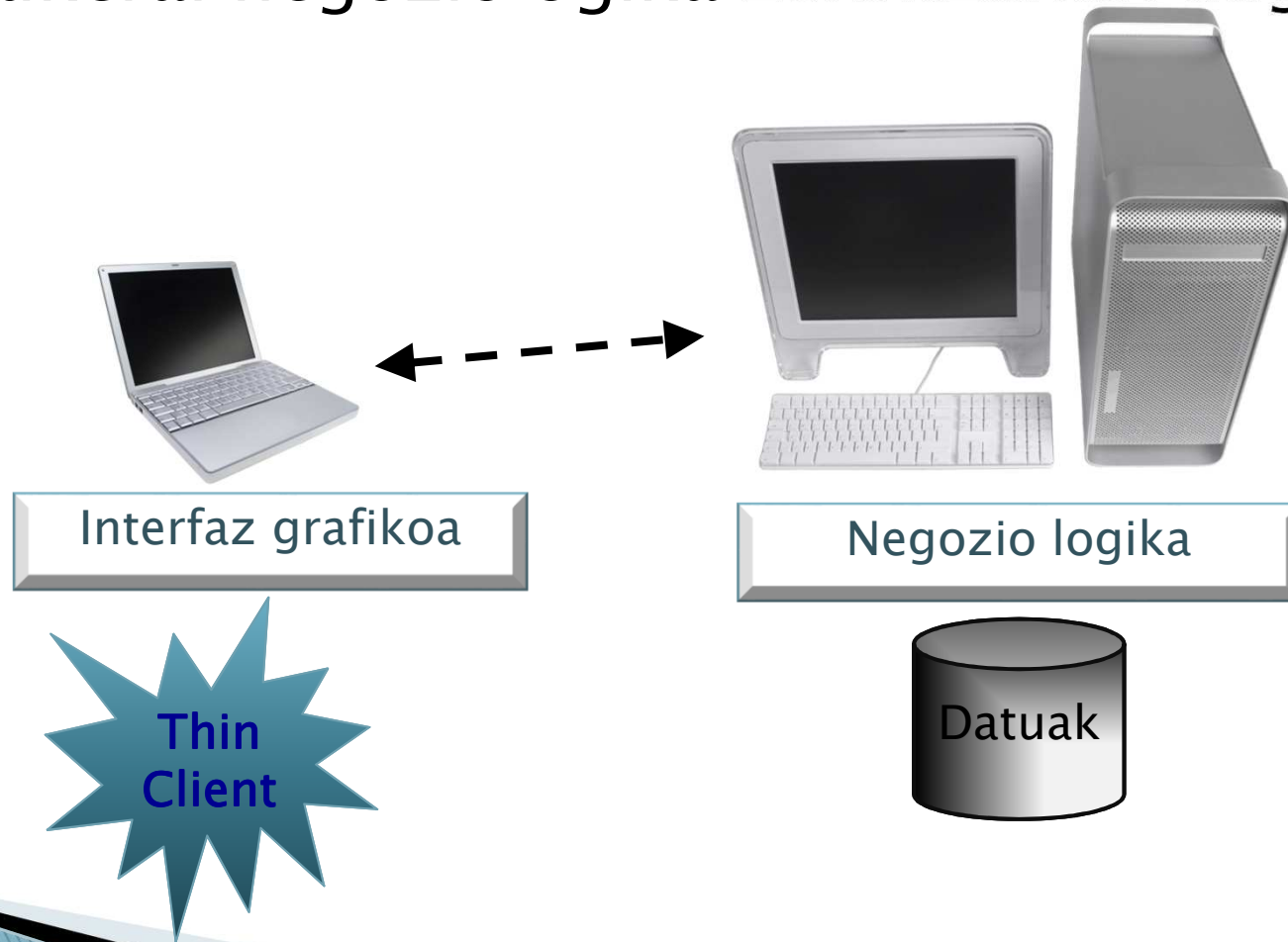
Bezero-zerbitzari aplikazioak

- ▶ Softwareak konputazio karga bi zati independentetan banatzen du, baina funtzio banaketa argirik izan gabe.
- ▶ Negozio logikaren kokapenaren arabera hiru aukera daude.



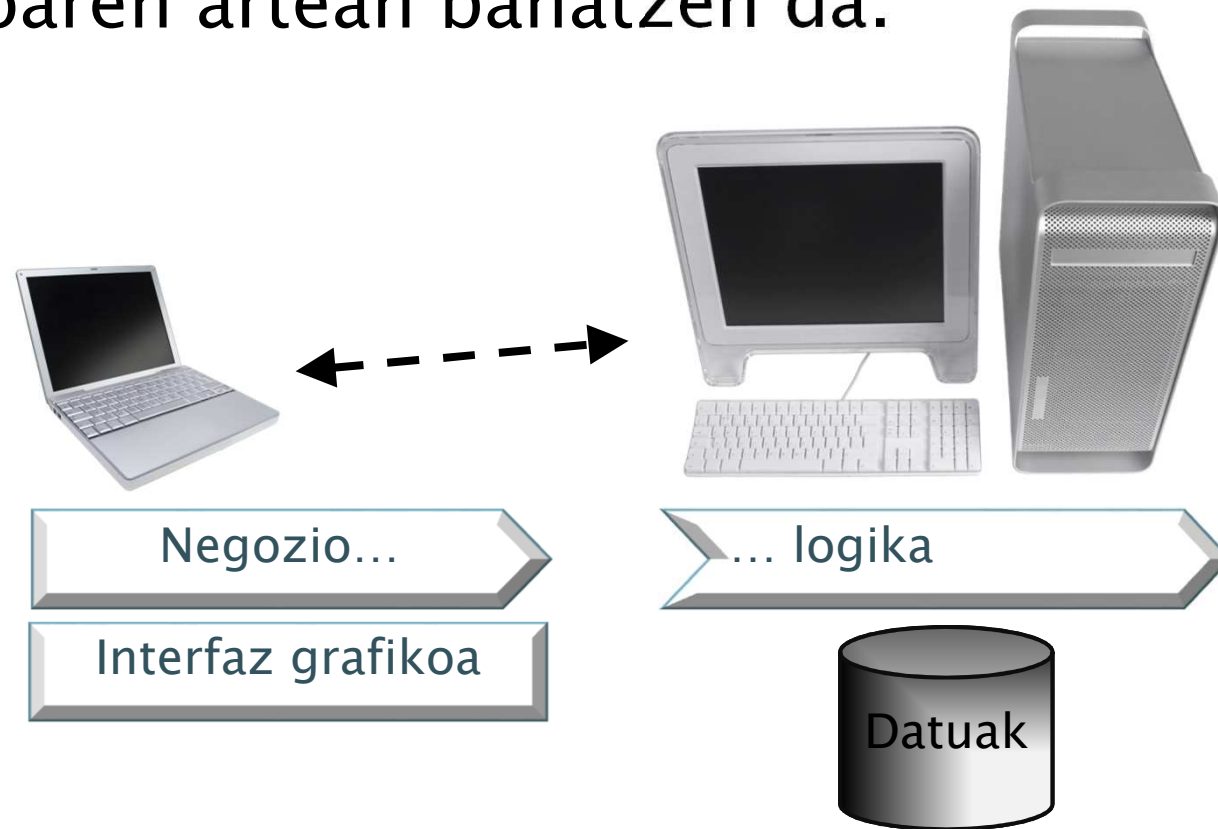
Bezzero-zerbitzari aplikazioak

- ▶ 1. aukera: negoziologika zerbitzarian dago.



Bezero-zerbitzari aplikazioak

- ▶ 2. aukera: negozio logika zerbitzari eta bezeroaren artean banatzen da.

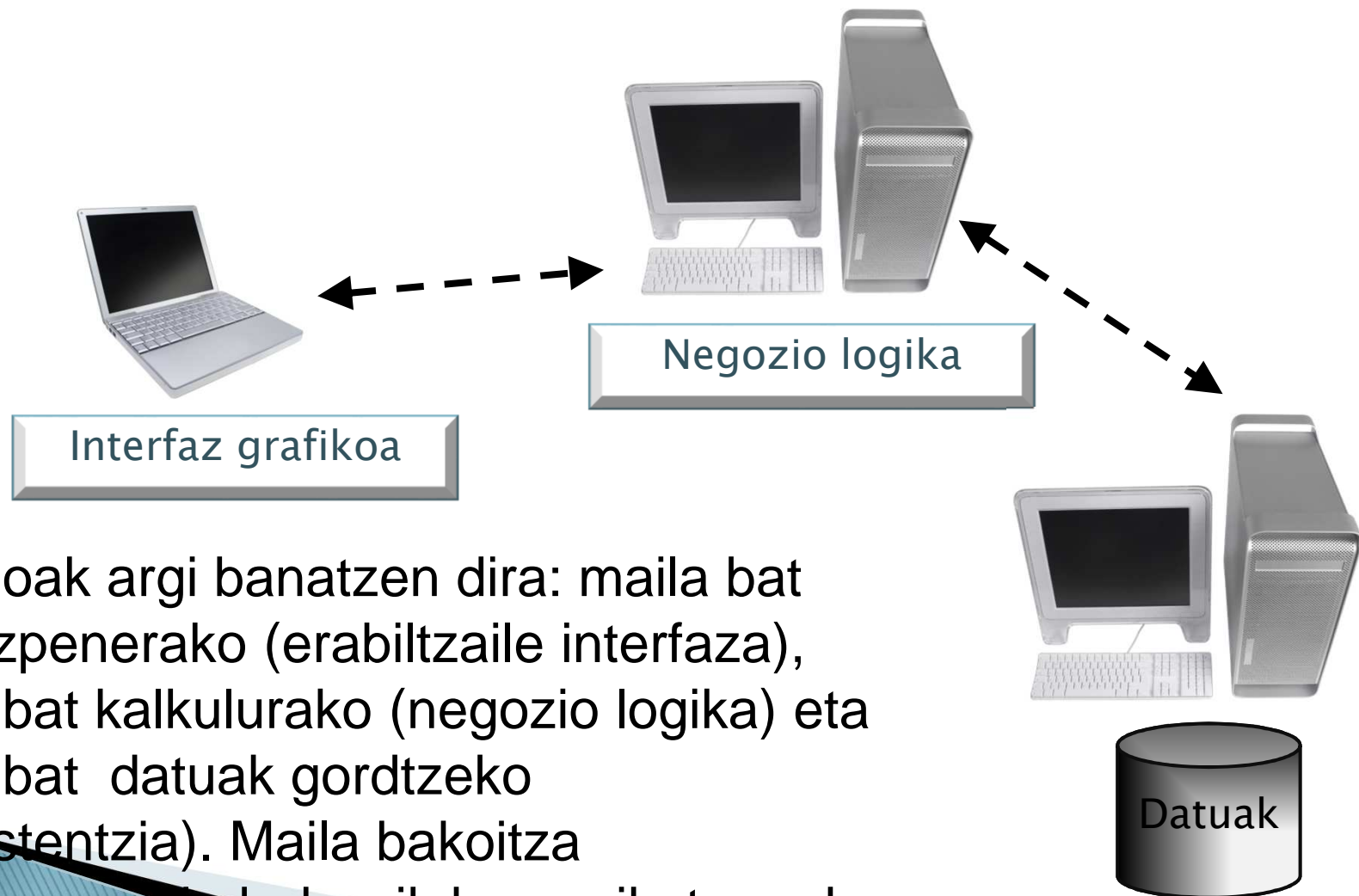


Bezzero-zerbitzari aplikazioak

- ▶ 3. aukera: negozio logika bezeroan dago.

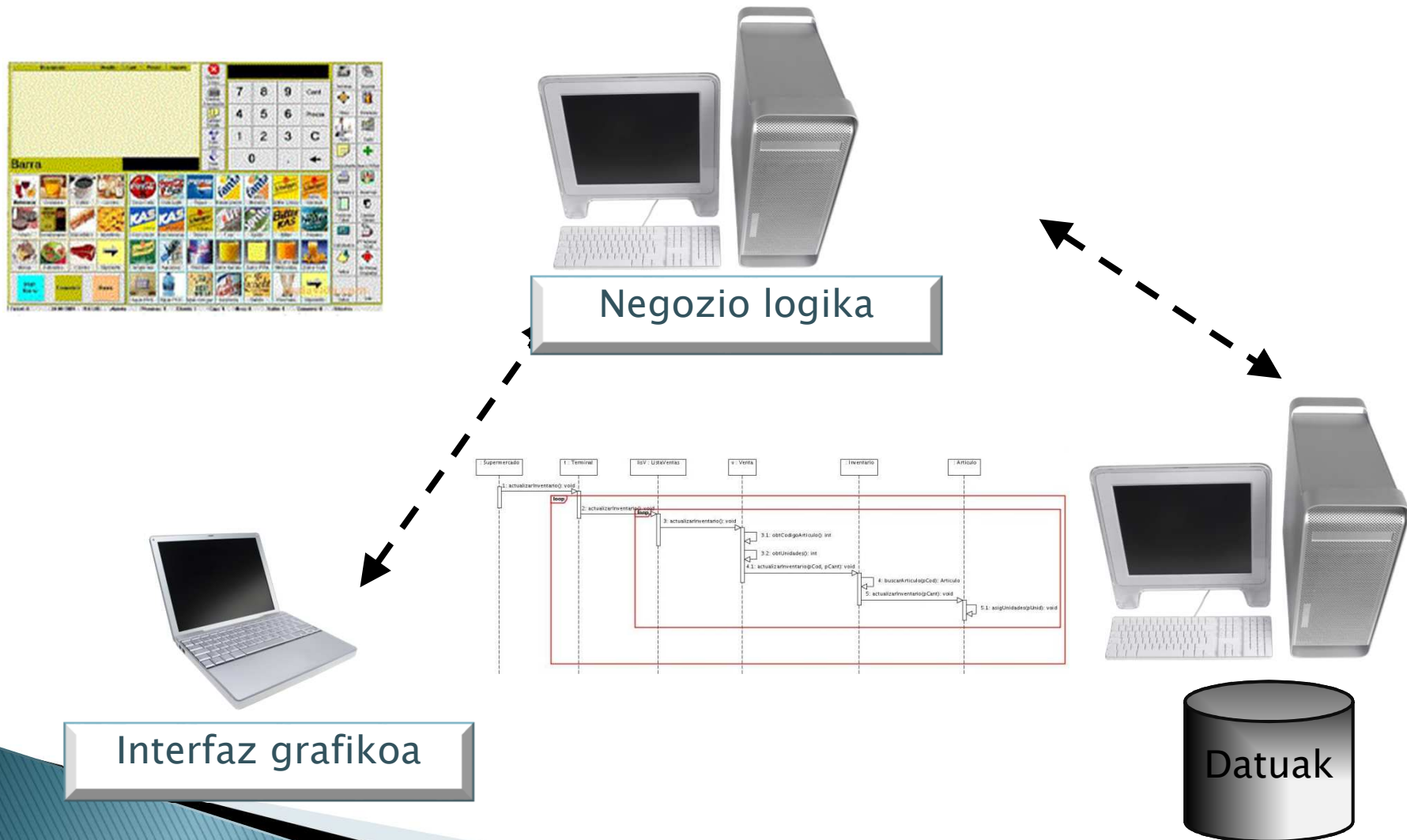


Hiru mailako aplikazioak



Funtzioak argi banatzen dira: maila bat aurkezpenerako (erabiltzaile interfaza), beste bat kalkulurako (negozio logika) eta beste bat datuak gordzeko (persistenzia). Maila bakoitza hurrengoarekin bakarrik komunikatzen da.

Hiru mailako aplikazioak



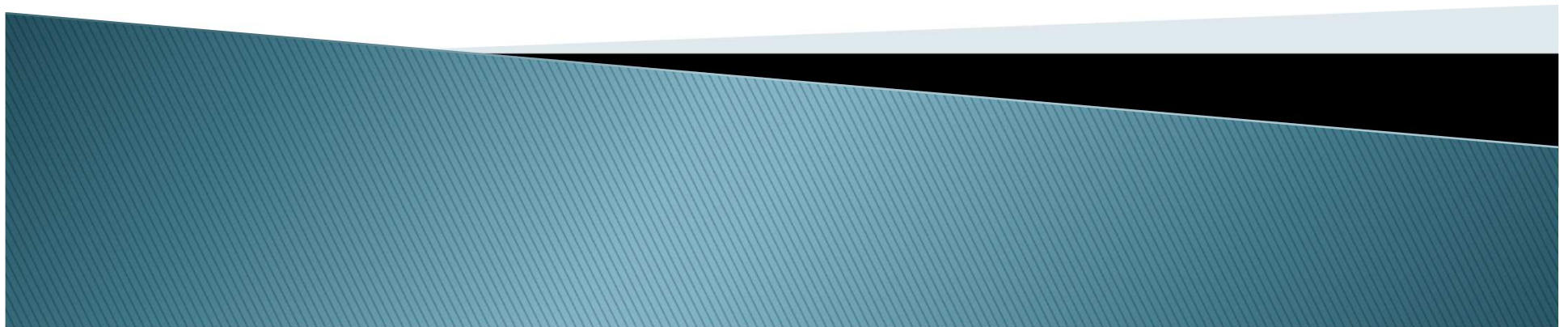
Hiru mailako aplikazioak

▶ Abantailak

- Pisu gutxiko bezeroak
- DBKS-z migratzeko gaitasuna (datu persistentzia)
- Eskalagarritasuna
- Ez dago plataformaren menpe
- Eguneratzeko erraztasuna



Adibidea: Txartel erreserba sistema



Sarrera

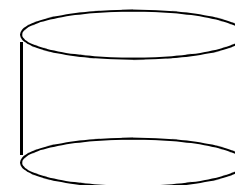
Bilete eskaera

Izena: Kepa Lasa

Esleipena Erreferentzia:0

Bilete eskatu

BILLETES : Tabla			
	NUMERO	ESTADO	NOMBRE
	0	OCUPADO	Kepa Lasa
	1	LIBRE	
	2	LIBRE	



DATU BASEA

```

public class BileteakEskatu3Maila extends JFrame {
// Nota: EZ DAGO OSORIK !!
    JLabel jLabel1 = new JLabel("Izena:");
    JButton jButton1 = new JButton("Bilete eskatu");
    public BileteakEskatu3Maila () {
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        conexion=DriverManager.getConnection("jdbc:odbc:Billetes"); }
    void jButton1_actionPerformed(ActionEvent e) {
        ResultSet rs = sentencia.executeQuery("SELECT NUMERO FROM"+
            " BILLETES WHERE ESTADO='LIBRE'");
        if (rs.next()) {
            int act = sentencia.executeUpdate("UPDATE BILLETES"+
                " SET ESTADO='OCUPADO', NOMBRE = "+jTextField1.getText()+
                " WHERE NUMERO="+rs.getString("NUMERO")+
                " AND ESTADO='LIBRE'");
            if (act>0) JTextArea1.append("Esleipena Erreferentzia: "+n+"\n");
            else JTextArea1.append("Errorea biletea esleitzerakoan"); }}

    public static void main (String []arg) {
        BileteakEskatu3Maila b = new BileteakEskatu3Maila ();
        b.setVisible(true);}}

```

```
public class BileteakEskatu3Maila extends JFrame { // Nota: EZ DAGO OSORIK !!
```

```
    JLabel jLabel1 = new JLabel("Izena:");
```

```
    JButton jButton1 = new JButton("Bilete eskatu");
```

```
public BileteakEskatu3Maila () {
```

```
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```

```
    konexio=DriverManager.getConnection("jdbc:odbc:Billetes"); }
```

```
    Statement galdera=konexio.createStatement();
```

```
void jButton1_actionPerformed(ActionEvent e) {
```

```
    ResultSet rs = galdera.executeQuery("SELECT NUMERO FROM "+  
        " BILLETES WHERE ESTADO='LIBRE'");
```

```
    if (rs.next()) {
```

```
        int act = sentencia.executeUpdate("UPDATE BILLETES"+
```

```
        " SET ESTADO='OCUPADO', NOMBRE = "+jTextField1.getText()+
```

```
        " WHERE NUMERO="+rs.getString("NUMERO")+
```

```
        " AND ESTADO='LIBRE'");
```

```
        if (act>0) jTextArea1.append("Esleipena Erreferentzia: "+n+"\n");
```

```
        else jTextArea1.append("Errorea biletea esleitzerakoan"); }
```

```
public static void main (String []arg) {
```

```
    BileteakEskatu3Maila b = new BileteakEskatu3Maila ();
```

AURKEZPENA

```
public class BileteakEskatu3Maila extends JFrame { // Nota: EZ DAGO OSORIK !!
```

```
    JLabel jLabel1 = new JLabel("Izena:");  
    JButton jButton1 = new JButton("Bilete eskatu");
```

DATU

```
public BileteakEskatu3Maila () {  
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");  
    konexio=DriverManager.getConnection("jdbc:odbc:Billetes"); }
```

ATZIPENA

```
    Statement galdera=konexio.createStatement();  
    void jButton1_actionPerformed(ActionEvent e) {
```

```
        ResultSet rs = galdera.executeQuery("SELECT NUMERO FROM "+  
            " BILLETES WHERE ESTADO='LIBRE'");
```

```
        if (rs.next()) {  
            int act = sentencia.executeUpdate("UPDATE BILLETES"+  
                " SET ESTADO='OCUPADO', NOMBRE = "+jTextField1.getText()+  
                " WHERE NUMERO="+rs.getString("NUMERO")+  
                " AND ESTADO='LIBRE'");
```

```
            if (act>0) jTextArea1.append("Esleipena Erreferentzia: "+n+"\n");  
            else jTextArea1.append("Errorea biletea esleitzerakoan"); }}
```

```
public static void main (String []arg) {  
    BileteakEskatu3Maila b = new BileteakEskatu3Maila ();
```

NEGOZIOAREN LOGIKA

```
public class BileteakEskatu3Maila extends JFrame { // Nota: EZ DAGO OSORIK !!
```

```
    JLabel jLabel1 = new JLabel("Izena:");
```

```
    JButton jButton1 = new JButton("Bilete eskatu");
```

```
public BileteakEskatu3Maila () {
```

```
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```

```
    konexio=DriverManager.getConnection("jdbc:odbc:Billetes"); }
```

```
    Statement galdera=konexio.createStatement();
```

```
void jButton1_actionPerformed(ActionEvent e) {
```

```
    ResultSet rs = galdera.executeQuery("SELECT NUMERO FROM "+  
        " BILLETES WHERE ESTADO='LIBRE'");
```

```
    if (rs.next()) {
```

```
        int act = sentencia.executeUpdate("UPDATE BILLETES"+
```

```
        " SET ESTADO='OCUPADO', NOMBRE = "+jTextField1.getText()+
```

```
        " WHERE NUMERO="+rs.getString("NUMERO")+
```

```
        " AND ESTADO='LIBRE'");
```

```
        if (act>0) jTextArea1.append("Esleipena Erreferentzia: "+n+"\n");
```

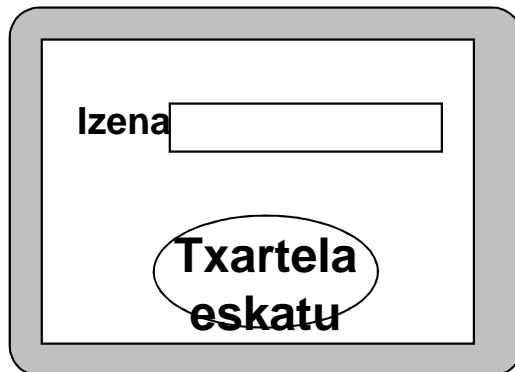
```
        else jTextArea1.append("Errorea billetea esleitzerakoan"); }}
```

```
public static void main (String []arg) {
```

```
    BileteakEskatu3Maila b = new BileteakEskatu3Maila ();
```

Mailatako softwarearen arkitektura logikoa

Aurkezpen maila



Izena

Txartela eskatu

Erabiltzailearen interfaze grafikoa:

Frame edo Applet

Negozioaren logikaren maila

```
public class TK
    implements TxartelenKudeatzailea{
    ...
    public int getTxartela (String iz){
    ...
    }
```

Negozioaren eragiketa propioak dituzten klaseak

- hasieratu
- getTxartela

Hemen negozioaren arauak aplikatu daitezke
(erositako 10 txartelengatik bat oparitzen da, etab...)

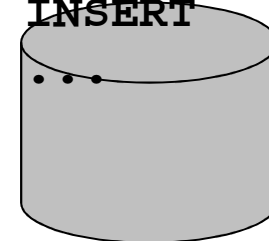
Datuen maila

SELECT

...

INSERT

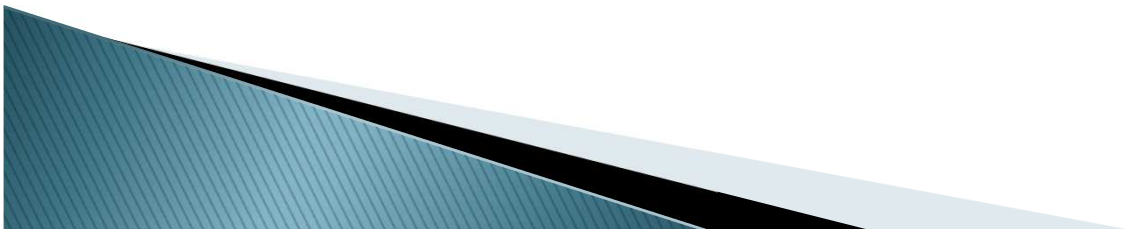
...



XML, datu-basea...

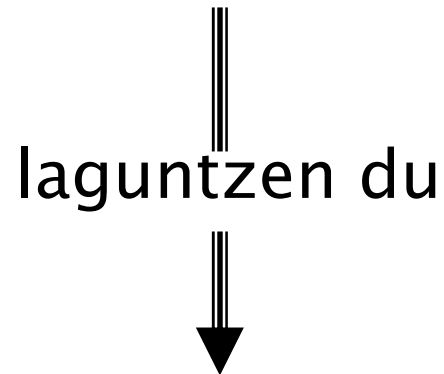
Mailatako softwarearen arkitektura logikoa

- ▶ **Aurkezpen maila**
 - Erabiltzaile interfazeak eta interakzioak inplementatzen dituzten osagaiak.
- ▶ **Negozioaren logikaren maila**
 - Negozioaren arazoak ebazten dituzten osagaiak.
 - Negozioaren arau propioak inplementatzen dituzten zerbitzu eta eragiketaz osatua dago.
- ▶ **Datuen maila**
 - Persistentzia lortzeko negozioaren logika mailak erabilia.
 - XML bat, datu-base bat, fitxategi bat, etab. da.

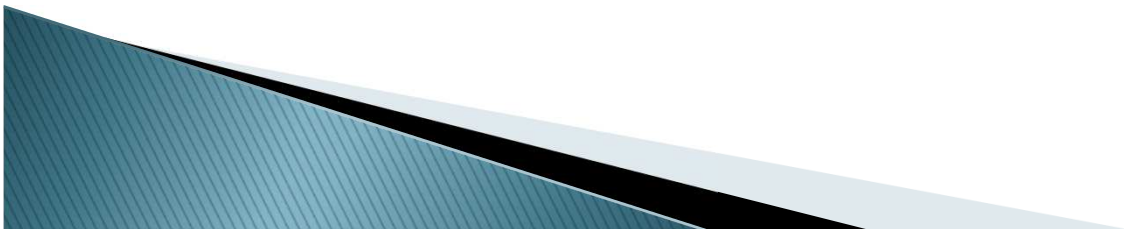


Mailatako softwarearen arkitektura logikoa

- ▶ Abantaila: Maila bat besteetatik isolatzea
 - Maila batean aldaketak egin daitezke, beste mailetan aldaketa gutxi eginik.
- ▶ Maila anitzeko softwarearen arkitektura logikoak



softwarearen hedagarritasuna eta berrerabilpena



Mailatako softwarearen arkitektura logikoa

▶ Aurkezpen maila

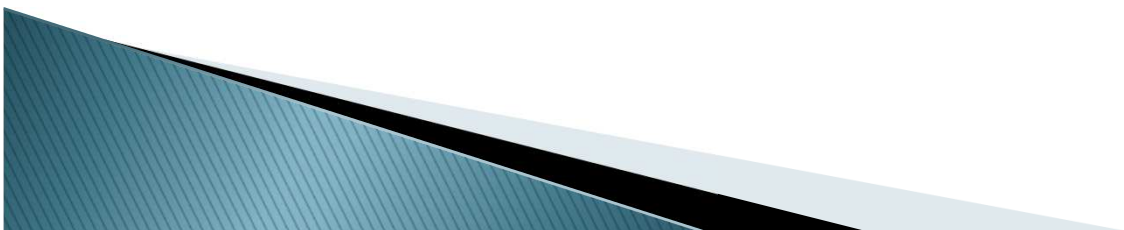
- Klaseen bidez eraikitzen da (Frame-ak edota Applet-ak).

▶ Negozioaren logikaren maila

- Negozioaren logikaren zerbitzu edota eragiketak dituzten Java klaseen bidez eraikitzen da.

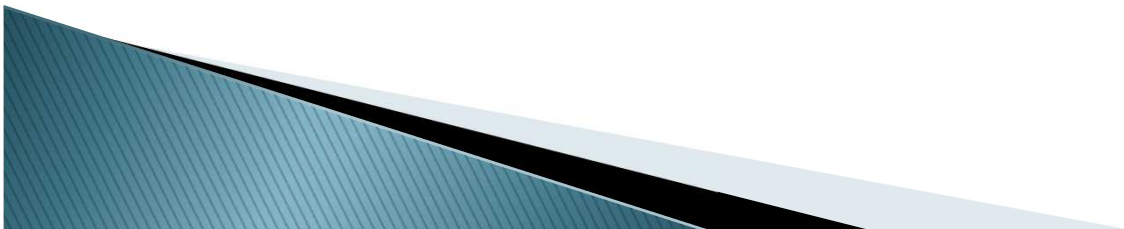
▶ Datuen maila

- XML fitxategia, datu-basea, testu fitxategia...



Mailatako softwarearen arkitektura fisikoa

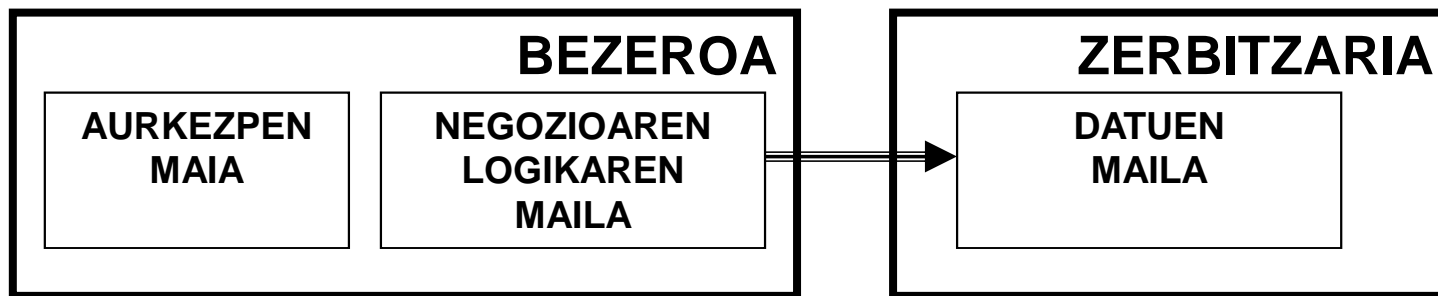
- ▶ Mailatako banaketa fisikorako aukerak:
 - 2 mailatako arkitektura
 - mailetako bi, nodo batean biltzen dira. Bi aukera:
 - negozioaren logika eta aurkezpena biltzen da, edo
 - negozioaren logikaren zati bat datuekin biltzen da.
 - 3 mailatako (edo gehiagoko) arkitektura
 - maila bakoitza, gutxienez, nodo ezberdin batean



Bi mailatako arkitektura fisikoa:

bezera gizona / zerbitzari mehea

- ▶ Aurkezpen maila eta negozioaren logikaren maila nodo batean biltzen dira.
- ▶ Beste nodoan datuen maila gelditzen da.



- Bezera eta Zerbitzariaren arteko komunikazioa SQLz
- APIak behar dira (adibidez JDBC edota ODBC)
- Bezera guztietan DBaren DRIVER-ak instalatu behar dira (edo XML-arekin lan egiteko libreriak)

BEZEROA

```
public class BileteakEskatu2MailaBezeroGizena extends
JFrame {
    BileteakEskatu2MailaBezeroGizena bileteKud;
    void jButton1_actionPerformed(ActionEvent e) {
        int ema =
        bileteKud.getBilete(jTextField1.getText()).getNum();
        if (ema<0) jTextArea1.append("Errorea bilete
esleitzerakoan");
        else jTextArea1.append("Esleipena. Erref: "+ema+"\n");} }
```

Aurkezpena

```
public class BileteKudDB
    implements BileteKud2MailaBezeroGizena
{ public BileteKudDB () {
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```

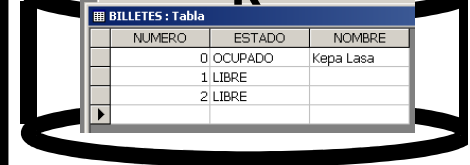
Negozio Logika

```
konexioa=DriverManager.getConnection("jdbc:odbc:Billetes
");}
    public Bilete getBilete(String izena)
    {ResultSet rs = sent.executeQuery("SELECT
NUMERO...");
    int act = sent.executeUpdate("UPDATE BILLETES ...");
    if (act>0) return new Bilete(n,izena); // Bilete esleituta
    else return new Bilete(n,""); // Ez zegoen librerik}}
```

DB

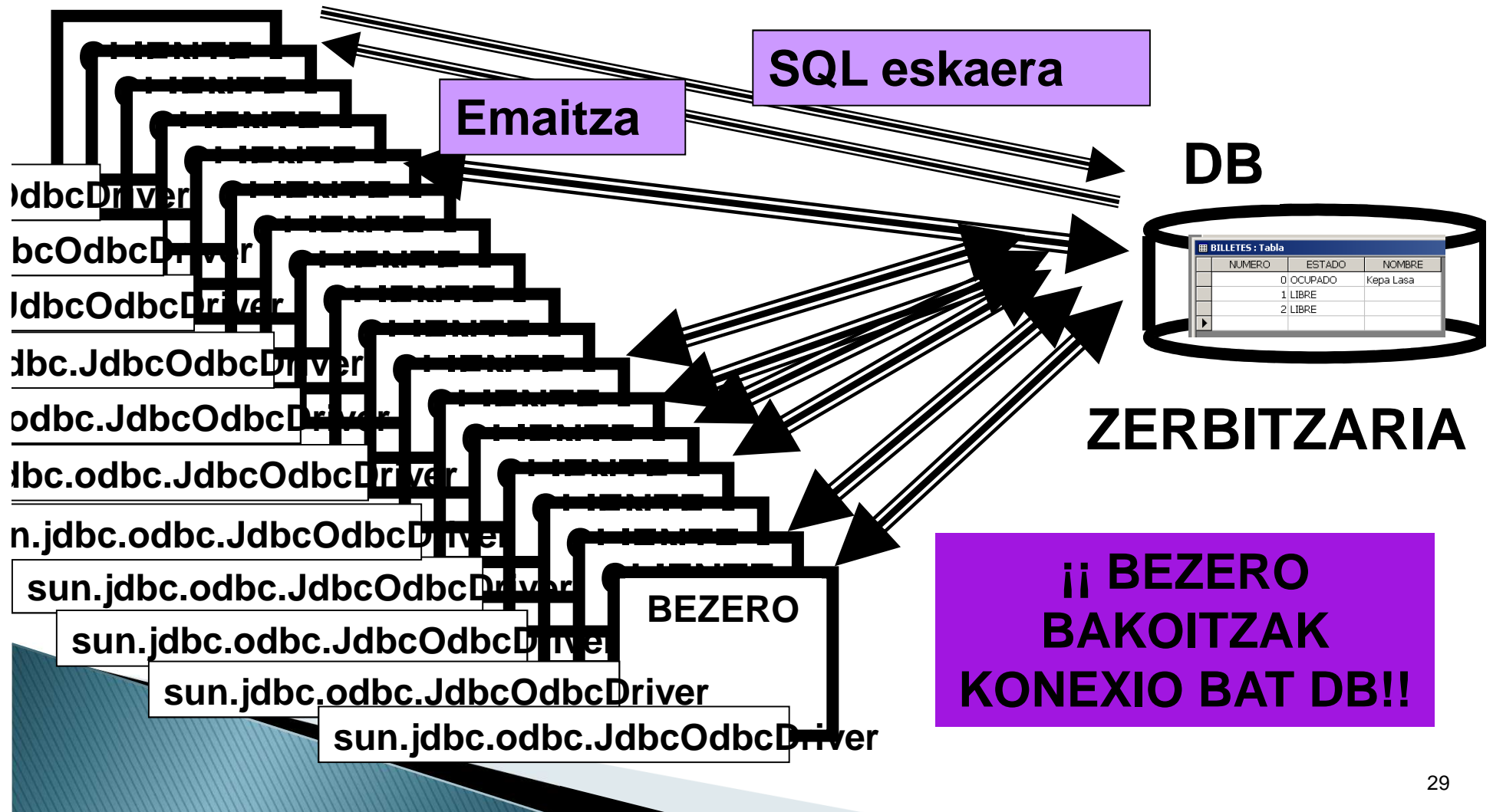
Datua

K



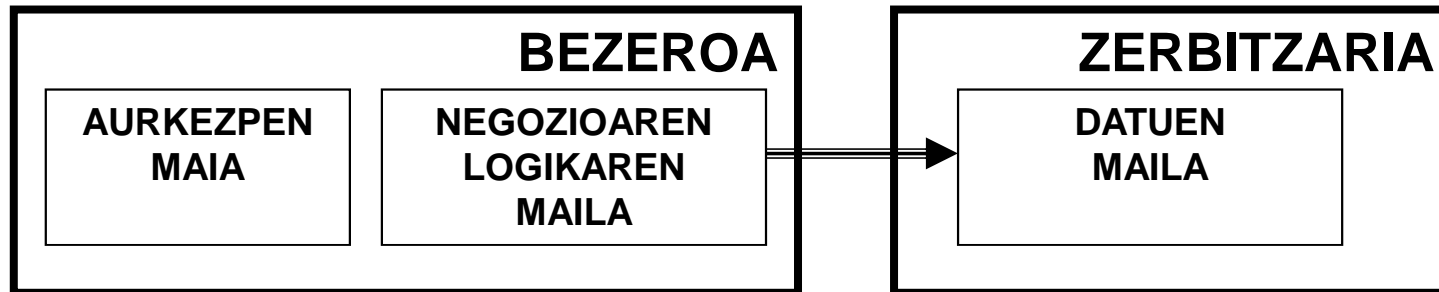
NUMERO	ESTADO	NOMBRE
0	Ocupado	Kepa Lasa
1	LIBRE	
2	LIBRE	

ZERBITZARIA



Bi mailatako arkitektura fisikoa:

bezero gizona / zerbitzari mehea

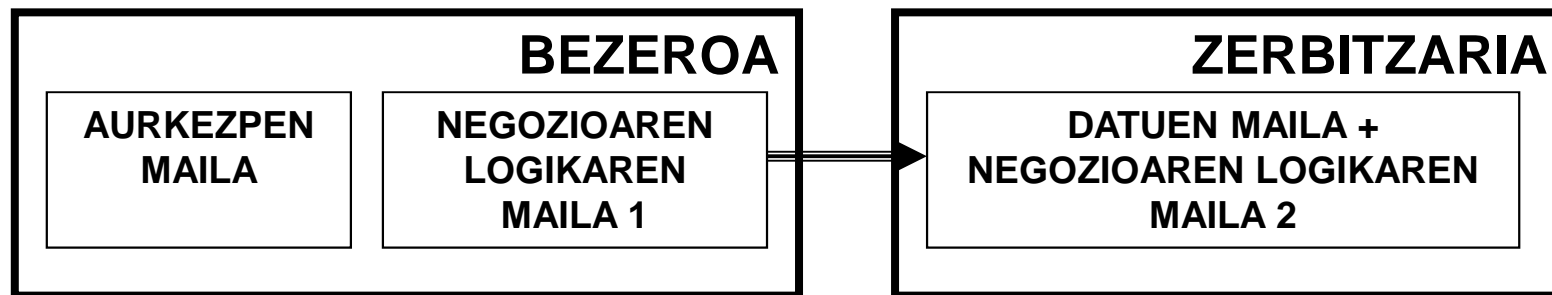


- ▶ Aplikazioaren hedaketa garestia da: driverrak instalatu eta bezero guztiak konfiguratu behar dira.
- ▶ DBKSa aldatzeak bezero guztietan berrinstalatzea suposatzen du.
- ▶ DBaren eskema aldatzeak bezero guztiei eragin diezaieke.
- ▶ Negozioaren logika aldatzeak ekartzen du birkonpilatzea eta bezero guztietan hedatzea.
- ▶ DBarekin konexioa garestia da. Bezero bakoitzak konexio bat du.
- ▶ Sarea gainkargatu daiteke, SQL sententzi bakoitzak sarea erabiltzen du.

Bi mailatako arkitektura fisikoa:

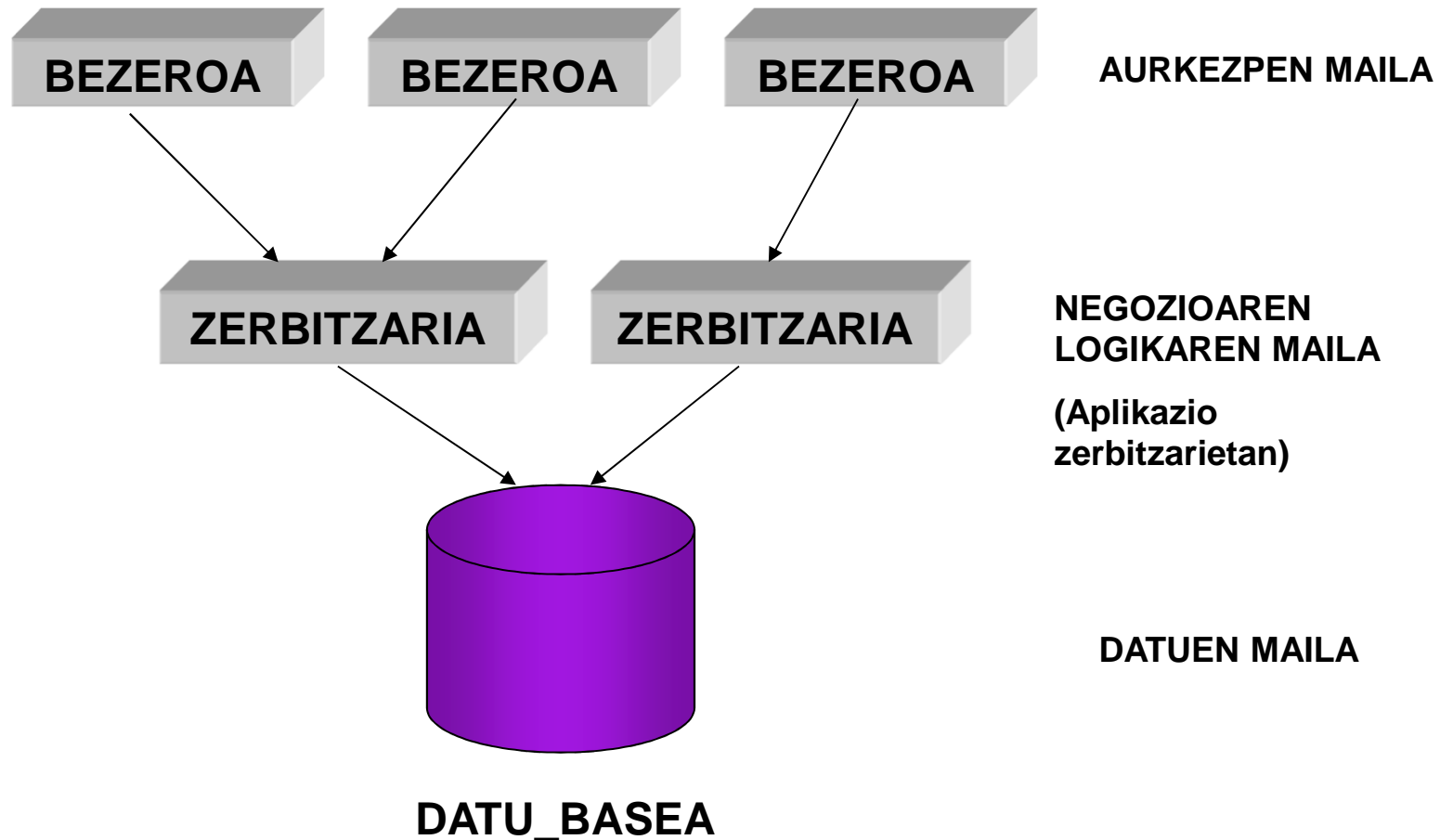
bezero mehea / zerbitzari gizona

- ▶ Negozioaren logikaren zati bat datuen mailarekin konbinatzen da.



- DBan biltegitratutako prozedurak (stored procedures) erabiltzen dira. Biltegitratutako prozedura batek SQL sententzia multzo bat egikaritzeko balio du.
- Bezeroa eta Zerbitzariaren arteko komunikazioa: SQLz eta biltegitratutako prozedurekin.
- APIak behar dira (adibidez JDBC edota ODBC).
- Bezero guztietan DBaren DRIVER-ak instalatu behar dira.

3 mailatako arkitektura fisikoa



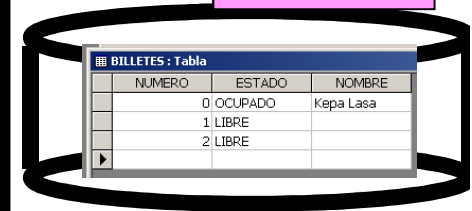
BEZEROA

```
public class BileteEskatu extends JFrame {
    BileteKud bileteKud;
    void jButton1_actionPerformed(ActionEvent e) {
        int res = bileteKud.getBilete(jTextField1.getText()).getNum();
        if (res<0) jTextArea1.append(" Errorea bilete esleitzerakoan ");
        else jTextArea1.append(" Esleipena. \nErref: "+ema+"\n ");
    }
}
```

Aurkezpena

DB

Datuak



NUMERO	ESTADO	NOMBRE
0	OCUPADO	Kepa Lasa
1	LIBRE	
2	LIBRE	

DATU
ZERBITZARIA

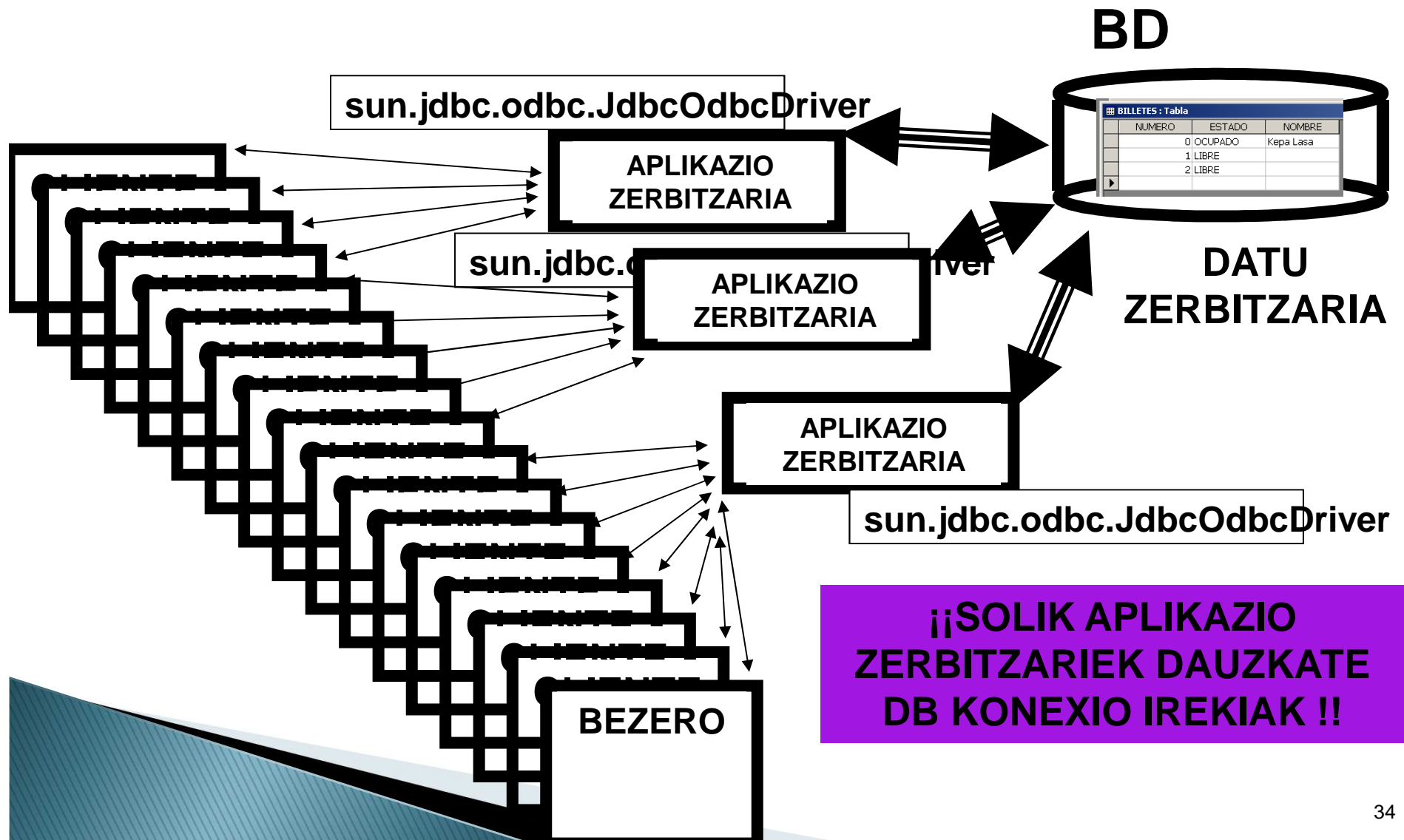
APLIKAZIO ZERBITZARIA

```
public class BileteKudDB
    implements BileteKud
{ public BileteKudDB () {
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");

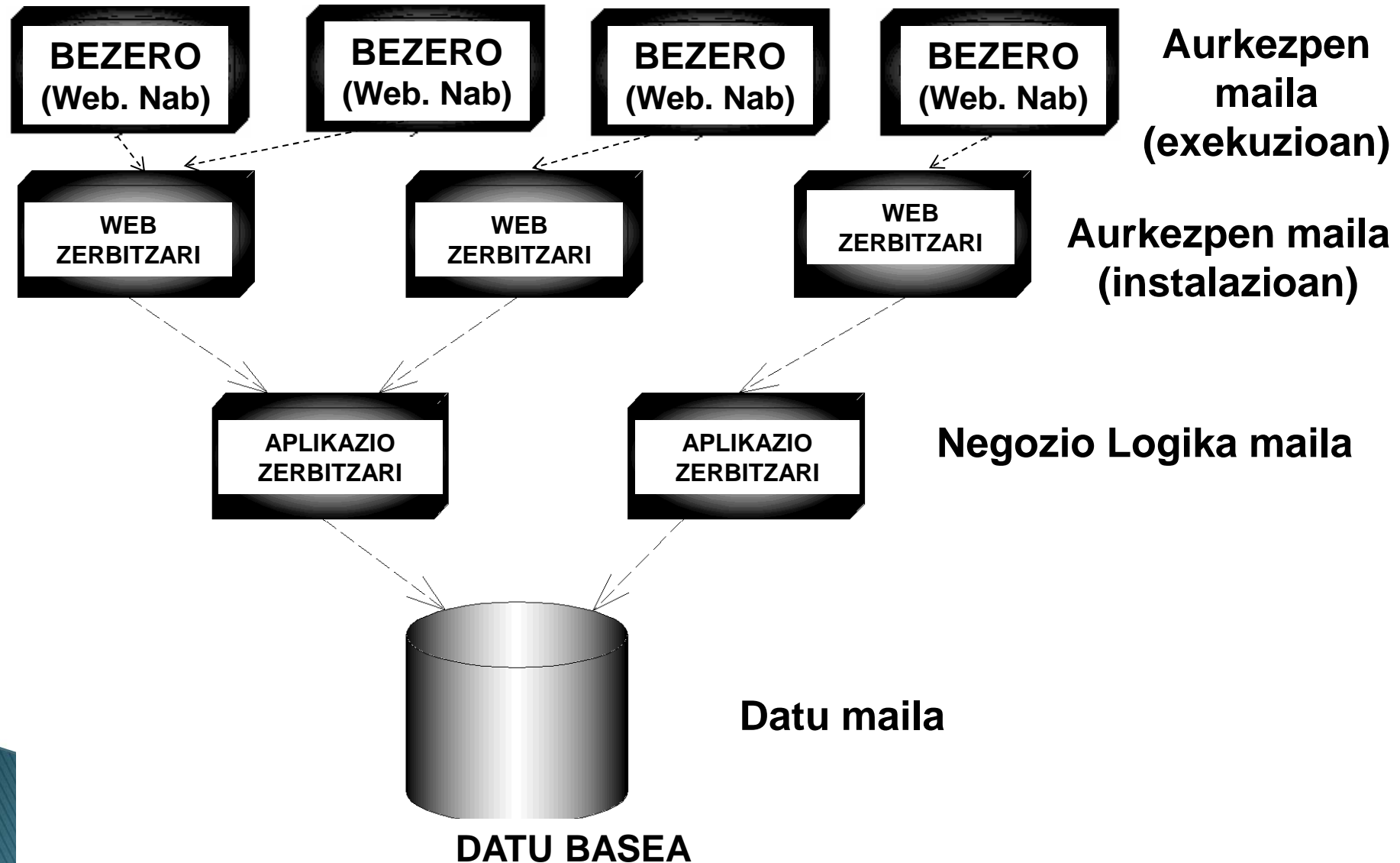
    konexioa=DriverManager.getConnection("jdbc:odbc:Billetes");
    public Bilete getBillete(String izena)
    {ResultSet rs = sent.executeQuery("SELECT
    NUMERO...");
    int res=sent.executeUpdate("UPDATE BILLETES ...");
}
```

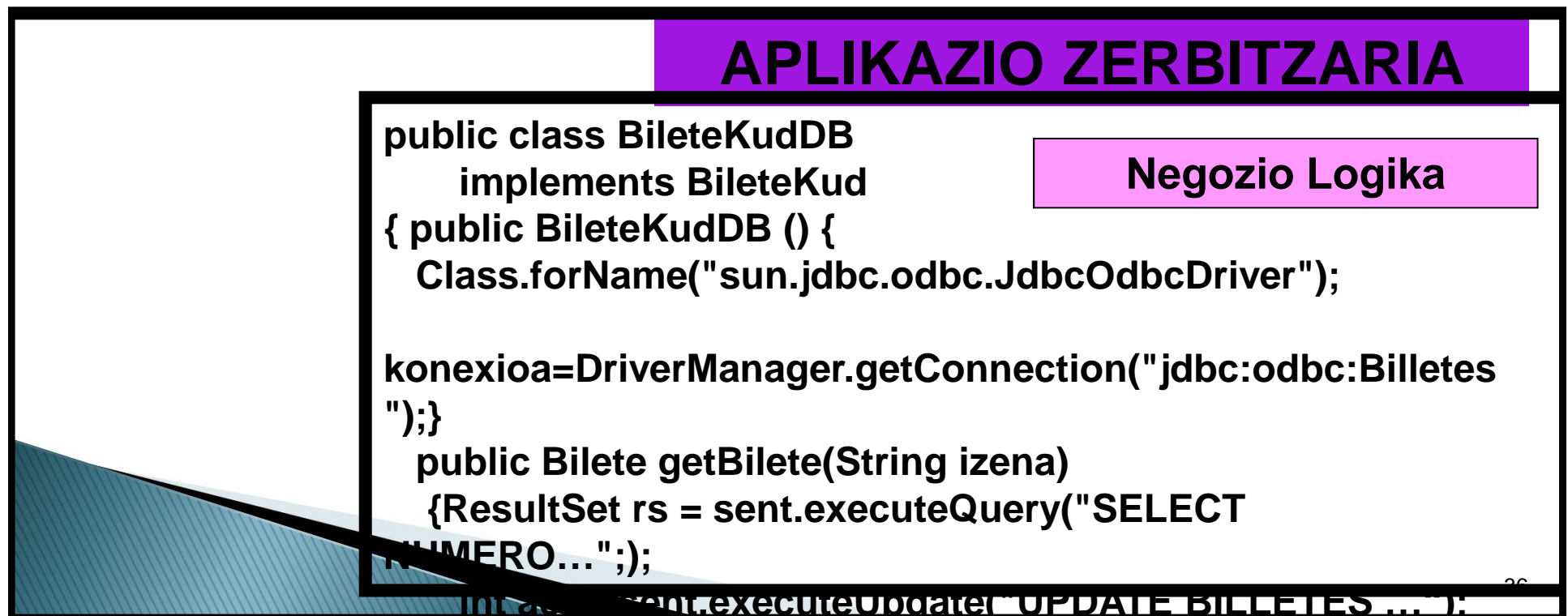
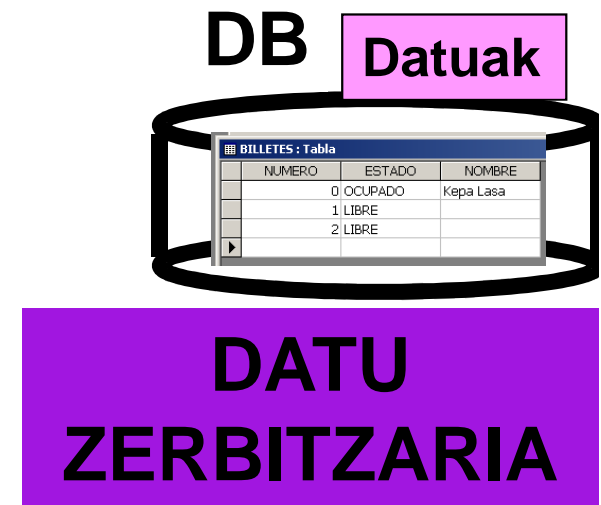
Negozio Logika

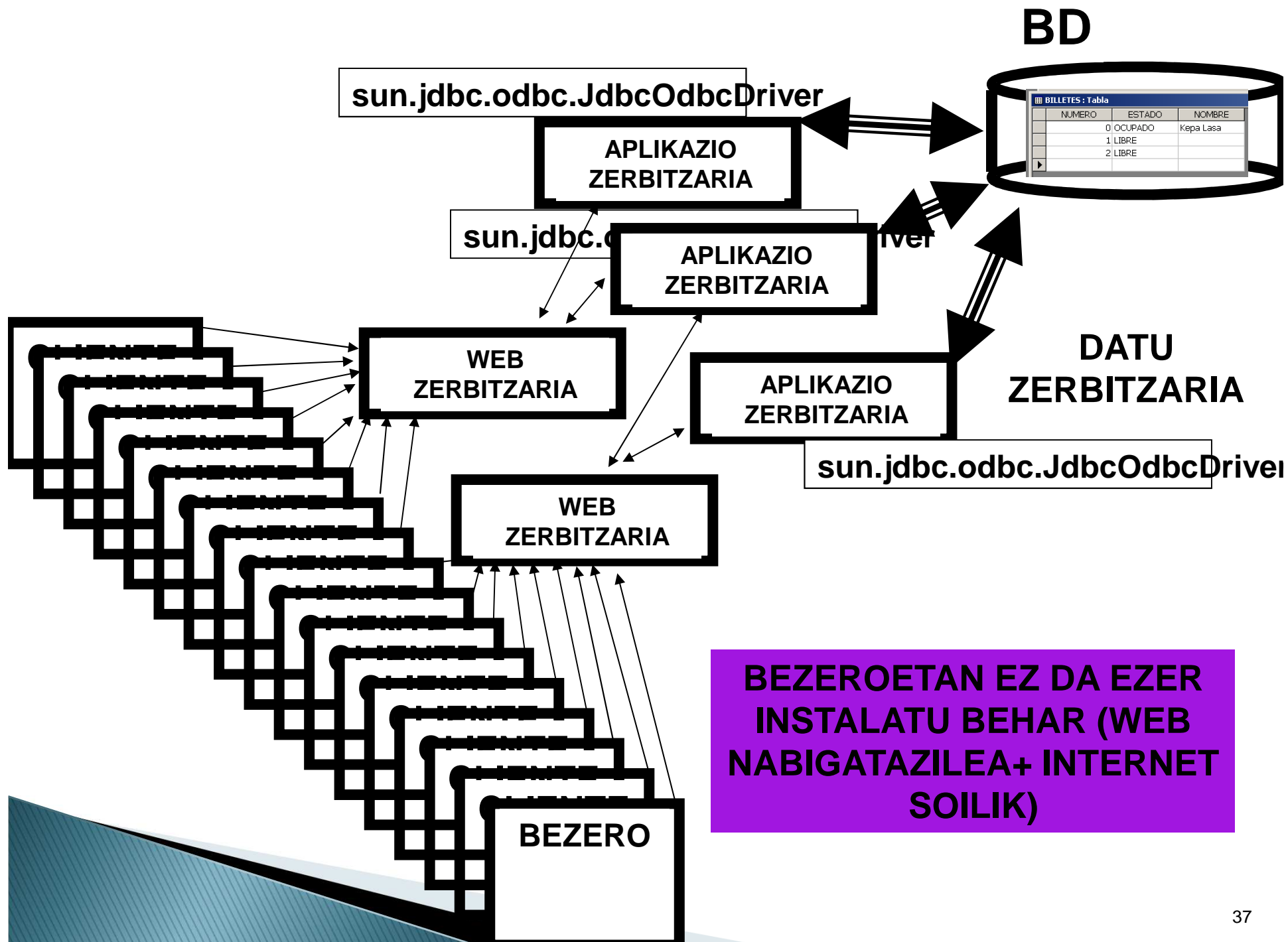
3 mailatako arkitektura fisikoa



Web aplikazioek maila gehiago eskeintzen dizkigute







3 mailatako arkitektura fisikoa

- DBaren driverrak soilik negozioaren logika dagoen nodoetan (nodo zerbitzarietan) instalatu behar da.
- DBKSa edo DBaren eskema aldatzeak ez du suposatzen bezero guztietan berrinstalatzea. Soilik negozioaren logikakoak.
- Negozioaren logika aldatzeak ez du eragiten berkonpilatzea eta bezero guztietan hedatzea.
- DBarekin konexioa ez da hain garestia. Bezeroek ez dituzte DBarekin konexioak egiten, soilik negozioaren logika duten zerbitzariak.

Orokorrean hobetzen da
eraginkortasunean, hedagarritasunean eta mantentzetan

3 mailatako arkitektura fisikoa

- ▶ Badago aplikazioak eraikitzeko teknologia, osagai eta objektu banatuen filosofia jarraituz (server-side components):
 - Enterprise JavaBeans (EJB):
 - Java 2 Enterprise Edition (J2EE) plataformarako osagaien arkitektura
 - Sun Microsystems-ek definitua
 - Aurkezpen maila gehiago zatitu daiteke Java Applet-ak, Servlet-ak edota JSP-ak erabiliz
 - CORBA:
 - Object Request Broker (ORB) bidez objektu banatuen artean komunikaziorako arkitektura
 - OMGk definitutako estandarra
 - DCOM/COM+ eta .NET plataforma
 - Microsoft-ek garatutako teknologia baliokidea