Refactorización (Bet22Lite)

Mateo Aguirre Duque - AZ846680

1. Write short units of code

Código inicial:

```
public boolean ApustuaEgin(Registered u, Vector<Quote> quote, Double balioa, Integer
apustuBikoitzaGalarazi) {
		Registered user = db.find(Registered.class, u.getUsername());
		Boolean b;
		if (user.getDirukop() >= balioa) {
			db.getTransaction().begin();
			ApustuAnitza apustuAnitza = new ApustuAnitza(user, balioa);
			db.persist(apustuAnitza);
			for (Quote quo : quote) {
				Quote kuote = db.find(Quote.class, quo.getQuoteNumber());
				Apustua ap = new Apustua(apustuAnitza, kuote);
				db.persist(ap);
				apustuAnitza.addApustua(ap);
				kuote.addApustua(ap);
			}
			db.getTransaction().commit();
			db.getTransaction().begin();
			if (apustuBikoitzaGalarazi == -1) {
				apustuBikoitzaGalarazi =
apustuAnitza.getApustuAnitzaNumber();
			}
			apustuAnitza.setApustuKopia(apustuBikoitzaGalarazi);
			user.updateDiruKontua(-balioa);
			Transaction t = new Transaction(user, balioa, new Date(),
"ApustuaEgin");
			user.addApustuAnitza(apustuAnitza);
			for (Apustua a : apustuAnitza.getApustuak()) {
				Apustua apu = db.find(Apustua.class, a.getApostuaNumber());
				Quote q = db.find(Quote.class,
apu.getKuota().getQuoteNumber());
				Sport spo = q.getQuestion().getEvent().getSport();
				spo.setApustuKantitatea(spo.getApustuKantitatea() + 1);

			}
```

```
                        user.addTransaction(t);
                        db.persist(t);
                        db.getTransaction().commit();
                        for (Jarraitzailea reg : user.getJarraitzaileLista()) {
                                Jarraitzailea erab = db.find(Jarraitzailea.class,
reg.getJarraitzaileaNumber());
                                b = true;
                                for (ApustuAnitza apu : erab.getNork().getApustuAnitzak()) {
                                        if (Objects.equals(apu.getApustuKopia(),
apustuAnitza.getApustuKopia())) {
                                                b = false;
                                        }
                                }
                                if (b) {
                                        if (erab.getNork().getDiruLimitea() < balioa) {
                                                this.ApustuaEgin(erab.getNork(), quote,
erab.getNork().getDiruLimitea(),
                                                                apustuBikoitzaGalarazi);
                                        } else {
                                                this.ApustuaEgin(erab.getNork(), quote, balioa,
apustuBikoitzaGalarazi);
                                        }
                                }
                        }
                        return true;
                } else {
                        return false;
                }

        }
```

Código refactorizado:

```
public boolean ApustuaEgin(Registered u, Vector<Quote> quote, Double balioa, Integer
apustuBikoitzaGalarazi) {
                Registered user = db.find(Registered.class, u.getUsername());
                if (user.getDirukop() >= balioa) {
                        ApustuAnitza apustuAnitza = this.apustuAnitzaSortu(quote, balioa,
user);

                        db.getTransaction().begin();
                        if (apustuBikoitzaGalarazi == -1) {
                                apustuBikoitzaGalarazi =
apustuAnitza.getApustuAnitzaNumber();
                        }
                        apustuAnitza.setApustuKopia(apustuBikoitzaGalarazi);
                        user.updateDiruKontua(-balioa);
```

```
                        Transaction t = new Transaction(user, balioa, new Date(),
"ApustuaEgin");
                        user.addApustuAnitza(apustuAnitza);

                        this.handituApustuarenZenbatekoaBat(apustuAnitza);

                        user.addTransaction(t);
                        db.persist(t);
                        db.getTransaction().commit();
                        this.jarraituApustuak(quote, balioa, apustuBikoitzaGalarazi, user,
apustuAnitza);
                        return true;
                } else {
                        return false;
                }

        }

        private void jarraituApustuak(Vector<Quote> quote, Double balioa, Integer
apustuBikoitzaGalarazi, Registered user,
                        ApustuAnitza apustuAnitza) {
                Boolean b;
                for (Jarraitzailea reg : user.getJarraitzaileLista()) {
                        Jarraitzailea erab = db.find(Jarraitzailea.class,
reg.getJarraitzaileaNumber());
                        b = true;
                        for (ApustuAnitza apu : erab.getNork().getApustuAnitzak()) {
                                if (Objects.equals(apu.getApustuKopia(),
apustuAnitza.getApustuKopia())) {
                                        b = false;
                                }
                        }
                        if (Boolean.TRUE.equals(b)) {
                                if (erab.getNork().getDiruLimitea() < balioa) {
                                        this.ApustuaEgin(erab.getNork(), quote,
erab.getNork().getDiruLimitea(), apustuBikoitzaGalarazi);
                                } else {
                                        this.ApustuaEgin(erab.getNork(), quote, balioa,
apustuBikoitzaGalarazi);
                                }
                        }
                }
        }

        private ApustuAnitza apustuAnitzaSortu(Vector<Quote> quote, Double balioa,
Registered user) {
                db.getTransaction().begin();
                ApustuAnitza apustuAnitza = new ApustuAnitza(user, balioa);
```

```
            db.persist(apustuAnitza);
            for (Quote quo : quote) {
                  Quote kuote = db.find(Quote.class, quo.getQuoteNumber());
                  Apustua ap = new Apustua(apustuAnitza, kuote);
                  db.persist(ap);
                  apustuAnitza.addApustua(ap);
                  kuote.addApustua(ap);
            }
            db.getTransaction().commit();
            return apustuAnitza;
      }

      private void handituApustuarenZenbatekoaBat(ApustuAnitza apustuAnitza) {
            for (Apustua a : apustuAnitza.getApustuak()) {
                  Apustua apu = db.find(Apustua.class, a.getApostuaNumber());
                  Quote q = db.find(Quote.class, apu.getKuota().getQuoteNumber());
                  Sport spo = q.getQuestion().getEvent().getSport();
                  spo.setApustuKantitatea(spo.getApustuKantitatea() + 1);
            }
      }
```

Descripción del error concreto detectado y descripción de la refactorización realizada:

El método "ApustuaEgin" supera las 15 líneas e interactúa con varios componentes y objetos. Esto va en contra del principio del software mantenible. La solución a este problema fue extraer fragmentos del método y convertirlos en métodos independientes con su propia identidad, para hacer que el código sea más sostenible a largo plazo y más legible a corto plazo.

Miembro que ha realizado la refactorización: Mateo Aguirre Duque

2. Write simple units of code

Código inicial:

```
public boolean gertaeraEzabatu(Event ev) {
            Event event = db.find(Event.class, ev);
            boolean resultB = true;
            List<Question> listQ = event.getQuestions();

            for (Question q : listQ) {
                  if (q.getResult() == null) {
                        resultB = false;
                  }
            }
            if (!resultB) {
```

```
                        return false;
                } else if (new Date().compareTo(event.getEventDate()) < 0) {
                        TypedQuery<Quote> Qquery = db.createQuery(
                                        "SELECT q FROM Quote q WHERE
q.getQuestion().getEvent().getEventNumber() =?1", Quote.class);
                        Qquery.setParameter(1, event.getEventNumber());
                        List<Quote> listQUO = Qquery.getResultList();
                        for (int j = 0; j < listQUO.size(); j++) {
                                Quote quo = db.find(Quote.class, listQUO.get(j));
                                for (int i = 0; i < quo.getApustuak().size(); i++) {
                                        ApustuAnitza apustuAnitza =
quo.getApustuak().get(i).getApustuAnitza();
                                        ApustuAnitza ap1 = db.find(ApustuAnitza.class,
apustuAnitza.getApustuAnitzaNumber());
                                        db.getTransaction().begin();
                                        ap1.removeApustua(quo.getApustuak().get(i));
                                        db.getTransaction().commit();
                                        if (ap1.getApustuak().isEmpty() &&
!ap1.getEgoera().equals("galduta")) {
                                                this.apustuaEzabatu(ap1.getUser(), ap1);
                                        } else if (!ap1.getApustuak().isEmpty() &&
ap1.irabazitaMarkatu()) {
                                                this.ApustuaIrabazi(ap1);
                                        }
                                        db.getTransaction().begin();
                                        Sport spo = quo.getQuestion().getEvent().getSport();
                                        spo.setApustuKantitatea(spo.getApustuKantitatea() -
1);
                                        db.getTransaction().commit();
                                }
                        }

                }
                db.getTransaction().begin();
                db.remove(event);
                db.getTransaction().commit();
                return true;
        }
```

Código refactorizado:

```
        public boolean gertaeraEzabatu(Event ev) {
                Event event = db.find(Event.class, ev);
                boolean resultB = true;
                List<Question> listQ = event.getQuestions();

                resultB = checkResult(resultB, listQ);
                if (!resultB) {
```

```
                return false;
            }

            if (new Date().compareTo(event.getEventDate()) < 0) {
                TypedQuery<Quote> Qquery = db.createQuery(
                        "SELECT q FROM Quote q WHERE
q.getQuestion().getEvent().getEventNumber() =?1", Quote.class);
                Qquery.setParameter(1, event.getEventNumber());
                List<Quote> listQUO = Qquery.getResultList();
                for (int j = 0; j < listQUO.size(); j++) {
                    Quote quo = db.find(Quote.class, listQUO.get(j));
                    removeBetsFromEvent(quo);
                }
            }
            db.getTransaction().begin();
            db.remove(event);
            db.getTransaction().commit();
            return true;
        }

        private void removeBetsFromEvent(Quote quo) {
            for (int i = 0; i < quo.getApustuak().size(); i++) {
                ApustuAnitza apustuAnitza =
quo.getApustuak().get(i).getApustuAnitza();
                ApustuAnitza ap1 = db.find(ApustuAnitza.class,
apustuAnitza.getApustuAnitzaNumber());
                db.getTransaction().begin();
                ap1.removeApustua(quo.getApustuak().get(i));
                db.getTransaction().commit();
                if (ap1.getApustuak().isEmpty() &&
!ap1.getEgoera().equals("galduta")) {
                    this.apustuaEzabatu(ap1.getUser(), ap1);
                } else if (!ap1.getApustuak().isEmpty() && ap1.irabazitaMarkatu()) {
                    this.ApustuaIrabazi(ap1);
                }
                db.getTransaction().begin();
                Sport spo = quo.getQuestion().getEvent().getSport();
                spo.setApustuKantitatea(spo.getApustuKantitatea() - 1);
                db.getTransaction().commit();
            }
        }

        private boolean checkResult(boolean resultB, List<Question> listQ) {
            for (Question q : listQ) {
                if (q.getResult() == null) {
                    resultB = false;
                }
            }
```

```
        return resultB;
    }
```

Descripción del error concreto detectado y descripción de la refactorización realizada:

La refactorización aplicada al método "gertaeraEzabatu" fue separar 2 componentes del método en métodos independientes, de forma que todos los métodos queden con un máximo de 4 bifurcaciones. Las bifurcaciones, en el caso del lenguaje Java, se generan por los siguientes componentes: if, case, ?, &&, ||, while, for y catch. El método original contaba con 8 bifurcaciones, lo cual va en contra de los principios de software mantenible.

Miembro que ha realizado la refactorización: Mateo Aguirre Duque

3. Duplicate code

Código inicial:

```
public class ObjectdbManagerServer extends JDialog {

    private static final long serialVersionUID = 1L;
    private final JPanel contentPanel = new JPanel();
    JTextArea textArea;
    ConfigXML c;

    // For windows
    private String objectDbpath = "src\\main\\resources\\objectdb.jar";

    // For mac

    public static void main(String[] args) {
        try {

            ObjectdbManagerServer dialog = new ObjectdbManagerServer();
            dialog.setDefaultCloseOperation(JDialog.DISPOSE_ON_CLOSE);
            dialog.setVisible(true);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public ObjectdbManagerServer() {

        setTitle("objectDBManagerServer: running the database server");
        setBounds(100, 100, 486, 180);
        getContentPane().setLayout(new BorderLayout());
        contentPanel.setBorder(new EmptyBorder(5, 5, 5, 5));
```

```
		getContentPane().add(contentPanel, BorderLayout.CENTER);
		contentPanel.setLayout(new BorderLayout(0, 0));

		textArea = new JTextArea();
		contentPanel.add(textArea);

		JPanel buttonPane = new JPanel();
		buttonPane.setLayout(new FlowLayout(FlowLayout.RIGHT));
		getContentPane().add(buttonPane, BorderLayout.SOUTH);

		JButton okButton = new JButton("OK");
		okButton.addActionListener(new ActionListener() {
			public void actionPerformed(ActionEvent e) {
				textArea.append("\n\n\nClosing the database... ");
				try {
					System.out.println("Server close");
					try {

						Runtime.getRuntime().exec("java -cp " +
objectDbpath + " com.objectdb.Server -port "
+ c.getDatabasePort() + " stop");

					} catch (Exception ioe) {
						System.out.println(ioe);
					}

					System.exit(1);

				} catch (Exception e1) {
				}
				System.exit(1);
			}
		});
		okButton.setActionCommand("OK");
		buttonPane.add(okButton);
		getRootPane().setDefaultButton(okButton);

		JButton cancelButton = new JButton("Cancel");
		cancelButton.setActionCommand("Cancel");
		buttonPane.add(cancelButton);

		ConfigXML c = ConfigXML.getInstance();

		if (c.isDatabaseLocal()) {
			textArea.append("\nERROR, the database is configured as local");
		} else {
			try {
				System.out.println("Lauching objectdb server");
```

```
                                try {
                                        Runtime.getRuntime().exec("java -cp " + objectDbpath
+ " com.objectdb.Server -port "
                                                        + c.getDatabasePort() + " start");
                                } catch (Exception ioe) {
                                        System.out.println(ioe);
                                }

                                textArea.append("\nAccess granted to: " + c.getUser());

                                textArea.append("\nPress button to exit this database server...
");

                        } catch (Exception e) {
                                textArea.append("Something has happened in
ObjectDbManagerServer: " + e.toString());

                        }

                }
        }

}
```

Código refactorizado:

```
public class ObjectdbManagerServer extends JDialog {

        private static final long serialVersionUID = 1L;
        private final JPanel contentPanel = new JPanel();
        JTextArea textArea;
        ConfigXML c;

        // For windows
        private String objectDbpath = "src\\main\\resources\\objectdb.jar";

        // For mac

        public static void main(String[] args) {
                try {

                        ObjectdbManagerServer dialog = new ObjectdbManagerServer();
                        dialog.setDefaultCloseOperation(JDialog.DISPOSE_ON_CLOSE);
                        dialog.setVisible(true);
                } catch (Exception e) {
                        e.printStackTrace();
                }
```

```
}

public ObjectdbManagerServer() {

	setTitle("objectDBManagerServer: running the database server");
	setBounds(100, 100, 486, 180);
	getContentPane().setLayout(new BorderLayout());
	contentPanel.setBorder(new EmptyBorder(5, 5, 5, 5));
	getContentPane().add(contentPanel, BorderLayout.CENTER);
	contentPanel.setLayout(new BorderLayout(0, 0));

	textArea = new JTextArea();
	contentPanel.add(textArea);

	JPanel buttonPane = new JPanel();
	buttonPane.setLayout(new FlowLayout(FlowLayout.RIGHT));
	getContentPane().add(buttonPane, BorderLayout.SOUTH);

	JButton okButton = new JButton("OK");
	okButton.addActionListener(new ActionListener() {
		public void actionPerformed(ActionEvent e) {
			textArea.append("\n\n\nClosing the database... ");
			try {
				System.out.println("Server close");
				serverAction(c, "stop");
				System.exit(1);
			} catch (Exception e1) {
			}
			System.exit(1);
		}
	});
	okButton.setActionCommand("OK");
	buttonPane.add(okButton);
	getRootPane().setDefaultButton(okButton);

	JButton cancelButton = new JButton("Cancel");
	cancelButton.setActionCommand("Cancel");
	buttonPane.add(cancelButton);

	ConfigXML c = ConfigXML.getInstance();

	if (c.isDatabaseLocal()) {
		textArea.append("\nERROR, the database is configured as local");
	} else {
		try {
			System.out.println("Lauching objectdb server");
			serverAction(c, "start");
			textArea.append("\nAccess granted to: " + c.getUser());
```

```
                                textArea.append("\nPress button to exit this database server...
");
                        } catch (Exception e) {
                                textArea.append("Something has happened in
ObjectDbManagerServer: " + e.toString());

                        }

                }
        }

        private void serverAction(ConfigXML c, String action) {
                try {
                        Runtime.getRuntime().exec("java -cp " + objectDbpath + "
com.objectdb.Server -port "
                                        + c.getDatabasePort() + " " + action);
                } catch (Exception ioe) {
                        System.out.println(ioe);
                }
        }

}
```

Descripción del error concreto detectado y descripción de la refactorización realizada:

Se realizó una refactorización en el método "ObjectdbManagerServer" de la clase "ObjectdbManagerServer". El objetivo fue separar un componente de ejecución de comandos de la base de datos para evitar la duplicación de código. La idea consistió en separar este componente y pasar la acción "start" o "stop" como parámetro, dependiendo de la lógica correspondiente. Esta modificación logra prevenir la repetición del código.

Miembro que ha realizado la refactorización: Mateo Aguirre Duque

4. Keep unit interfaces small

Código inicial:

Código refactorizado:

Descripción del error concreto detectado y descripción de la refactorización realizada:

Se constató que en el proyecto de prueba proporcionado por el docente no existía ningún método con más de 4 parámetros de entrada, lo que imposibilitó llevar a cabo la refactorización prevista. A pesar de esto, se realizó un análisis exhaustivo de en qué consistiría dicha refactorización. En términos generales, este proceso implica la creación de

clases u objetos que agrupen los parámetros de entrada en una entidad más descriptiva y manejable. El propósito principal de esta acción es reducir la cantidad de parámetros en los métodos y convertir las unidades en entidades más pequeñas y menos complejas.

Miembro que ha realizado la refactorización: Mateo Aguirre Duque