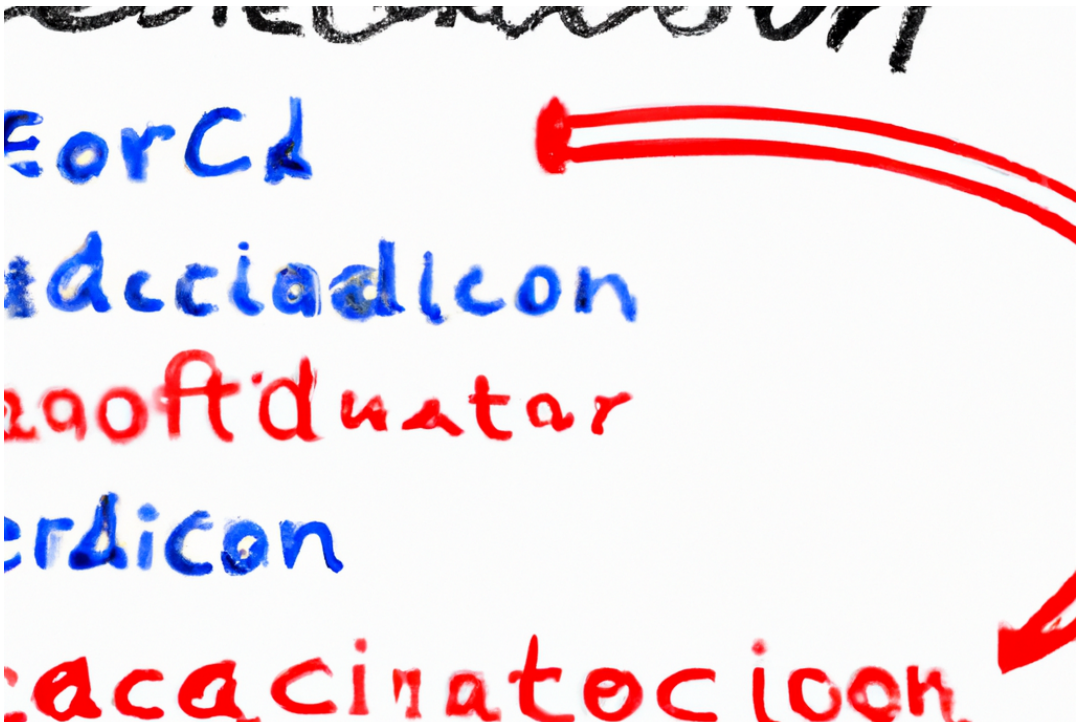

Parte 2 - Refactorización

Proyecto BETS - Ingeniería del Software II - UPV/EHU

16 de Octubre de 2023

—

Nicolás Aguado

Xiomara Cáceres

Eider Fernández

Índice

Refactorización #1

Código Inicial

```
public boolean gertaerakSortu(String description,Date eventDate, String sport) {
    boolean b = true;
    db.getTransaction().begin();
    Sport spo =db.find(Sport.class, sport);
    if(spo!=null) {
        TypedQuery<Event> Equery = db.createQuery("SELECT e FROM Event e WHERE e.getEventDate() =?1 ",Event.class);
        Equery.setParameter(1, eventDate);
        for(Event ev: Equery.getResultList()) {
            if(ev.getDescription().equals(description)) {
                b = false;
            }
        }
        if(b) {
            String[] taldeak = description.split("-");
            Team lokala = new Team(taldeak[0]);
            Team kanpokoa = new Team(taldeak[1]);
            Event e = new Event(description, eventDate, lokala, kanpokoa);
            e.setSport(spo);
            spo.addEvent(e);
            db.persist(e);
        }
    }else {
        return false;
    }
    db.getTransaction().commit();
    return b;
}
```

Código Refactorizado

```
public boolean gertaerakSortu(String description,Date eventDate, String sport) {
    boolean b = true;
    db.getTransaction().begin();
    Sport spo =db.find(Sport.class, sport);
    if(spo!=null) {
        b = isAnyFreeSpotForEvent(description, eventDate);
        if(b) {
            String[] taldeak = description.split("-");
            Team lokala = new Team(taldeak[0]);
            Team kanpokoa = new Team(taldeak[1]);
            Event e = new Event(description, eventDate, lokala, kanpokoa);
            e.setSport(spo);
            spo.addEvent(e);
            db.persist(e);
        }
    }else {
        return false;
    }
    db.getTransaction().commit();
    return b;
}

private boolean isAnyFreeSpotForEvent(String description, Date eventDate) {
    boolean b = true;
    TypedQuery<Event> Equery = db.createQuery("SELECT e FROM Event e WHERE e.getEventDate() =?1 ",Event.class);
    Equery.setParameter(1, eventDate);
    for(Event ev: Equery.getResultList()) {
        if(ev.getDescription().equals(description)) {
            b = false;
        }
    }
    return b;
}
```

Descripción del Error y Acciones

El método era muy largo y complejo de entender porque tenía muchas bifurcaciones y condicionales ("Write simple units of code"). He extraído una llamada a la base de datos a un método aparte. El método que he extraído comprueba si hay hueco libre (si no existe el evento) en el día indicado.

LINK AL COMMIT DE GITHUB

Miembro del Grupo

Cambios realizados por Nicolás Aguado

Refactorización #2

Código Inicial

```
public void EmaitzakIpini(Quote quote) throws EventNotFinished{

    Quote q = db.find(Quote.class, quote);
    String result = q.getForecast();

    if(new Date().compareTo(q.getQuestion().getEvent().getEventDate())<0)
        throw new EventNotFinished();

    Vector<Apustua> listApustuak = q.getApustuak();
    db.getTransaction().begin();
    Question que = q.getQuestion();
    Question question = db.find(Question.class, que);
    question.setResult(result);
    for(Quote quo: question.getQuotes()) {
        for(Apustua apu: quo.getApustuak()) {

            Boolean b=apu.galdutaMarkatu(quo);
            if(b) {
                apu.getApustuAnitza().setEgoera("galduta");
            }else {
                apu.setEgoera("irabazita");
            }
        }
    }
    db.getTransaction().commit();
    for(Apustua a : listApustuak) {
        db.getTransaction().begin();
        Boolean bool=a.getApustuAnitza().irabazitaMarkatu();
        db.getTransaction().commit();
        if(bool) {
            this.ApustuaIrabazi(a.getApustuAnitza());
        }
    }
}
```

Código Refactorizado

```
public void EmaitzakIpini(Quote quote) throws EventNotFinished{

    Quote q = db.find(Quote.class, quote);
    String result = q.getForecast();

    if(new Date().compareTo(q.getQuestion().getEvent().getEventDate())<0)
        throw new EventNotFinished();

    Vector<Apustua> listApustuak = q.getApustuak();
    db.getTransaction().begin();
    Question que = q.getQuestion();
    Question question = db.find(Question.class, que);
    question.setResult(result);
    markatuKuotak(question);
    db.getTransaction().commit();
    markatuIrabaziak(listApustuak);
}

private void markatuKuotak(Question question) {
    for(Quote quo: question.getQuotes()) {
        for(Apustua apu: quo.getApustuak()) {

            Boolean b=apu.galdutaMarkatu(quo);
            if(b) {
                apu.getApustuAnitza().setEgoera("galduta");
            }else {
                apu.setEgoera("irabazita");
            }
        }
    }
}

private void markatuIrabaziak(Vector<Apustua> listApustuak) {
    for(Apustua a : listApustuak) {
        db.getTransaction().begin();
        Boolean bool=a.getApustuAnitza().irabazitaMarkatu();
        db.getTransaction().commit();
        if(bool) {
            this.ApustuaIrabazi(a.getApustuAnitza());
        }
    }
}
```

Descripción del Error y Acciones

El código del método es muy largo, y hay partes que se podrían extraer como métodos. Esto favorecería la reutilización de código y la mejor comprensión de la lógica en si.

LINK AL COMMIT EN GITHUB

Miembro del Grupo

Cambios realizados por Nicolás Aguado

Refactorización #3

Código Inicial

```
        }
        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(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

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

    }
```

Descripción del Error y Acciones

En el método se están repitiendo algunos cachos del código, cosa que perjudica la eficiencia y es una mala práctica (“Duplicate code”)

LINK AL COMMIT DE GITHUB

Miembro del Grupo

Cambios realizados por Nicolás Aguado

Refactorización #4

Código Inicial

```
if(query.getResultList().isEmpty()) {
    b=true;
    String[] taldeak = gertaera.getDescription().split("-");
    Team lokala = new Team(taldeak[0]);
    Team kanpokoa = new Team(taldeak[1]);
    Event gertKopiatu = new Event(gertaera.getDescription(), date, lokala, kanpokoa);
    gertKopiatu.setSport(gertaera.getSport());
    gertaera.getSport().addEvent(gertKopiatu);
    db.persist(gertKopiatu);
        for(Question q : gertaera.getQuestions()) {
            Question que= new Question(q.getQuestion(), q.getBetMinimum(), gertKopiatu);
            gertKopiatu.listaraGehitu(que);
            Question galdera = db.find(Question.class, q.getQuestionNumber());
            db.persist(que);
```

Código Refactorizado

```
query.setParameter(2, date);
if(query.getResultList().isEmpty()) {
    b=true;
    Event gertKopiatu = eventuaBerriaSortu(date, gertaera);
    gertKopiatu.setSport(gertaera.getSport());
    gertaera.getSport().addEvent(gertKopiatu);
    db.persist(gertKopiatu);
    for(Question q : gertaera.getQuestions()) {
```

```
private Event eventuaBerriaSortu(Date date, Event gertaera) {
    String[] taldeak = gertaera.getDescription().split("-");
    Team lokala = new Team(taldeak[0]);
    Team kanpokoa = new Team(taldeak[1]);
    return new Event(gertaera.getDescription(), date, lokala, kanpokoa);
}
```

Descripción del Error y Acciones

Este error se basa en el objetivo único de los métodos (“Keep unit interfaces small”). Las tareas que no formen parte del objetivo final del método se pueden modularizar y así se favorece la reutilización de código y se mejora la lectura posterior.

LINK AL COMMIT DE GITHUB

Miembro del Grupo

Cambios realizados por Nicolás Aguado

Refactorización #5

Código Inicial

```
public boolean gertaerakKopiatu(Event e, Date date) {
	Boolean b=false;
	Event gertaera = db.find(Event.class, e.getEventNumber());
	db.getTransaction().begin();


	TypedQuery<Event> query = db.createQuery("SELECT ev FROM Event ev
WHERE ev.getDescription()=?1 and ev.getEventDate()=?2",Event.class);
	query.setParameter(1,gertaera.getDescription());
	query.setParameter(2, date);
	if(query.getResultList().isEmpty()) {
		b=true;
		Event gertKopiatu = eventuBerriaSortu(date, gertaera);
		gertKopiatu.setSport(gertaera.getSport());
		gertaera.getSport().addEvent(gertKopiatu);
		db.persist(gertKopiatu);
			for(Question q : gertaera.getQuestions()) {
				Question que= new Question(q.getQuestion(),
q.getBetMinimum(), gertKopiatu);
				gertKopiatu.listaraGehitu(que);
				Question galdera = db.find(Question.class,
q.getQuestionNumber());
				db.persist(que);
				for(Quote k: galdera.getQuotes()) {
					Quote kuo= new Quote(k.getQuote(),
```

```
k.getForecast(), que);
                                que.listaraGehitu(kuo);
                                db.persist(kuo);
                            }
                        }
        }
        db.getTransaction().commit();
        return b;
}
```

Código Refactorizado

```
Event gertaera = db.find(Event.class, e.getEventNumber());
if(gertaera == null) return false;
db.getTransaction().begin();

query.setParameter(2, date);
if(!query.getResultList().isEmpty()) return false;
Event gertKopiatu = eventuBerriaSortu(date, gertaera);
gertKopiatu.setSport(gertaera.getSport());

db.persist(gertKopiatu);
for(Question q : gertaera.getQuestions()) {
    Question que= new Question(q.getQuestion(), q.getBetMinimum(), gertKopiatu);
    gertKopiatu.listaraGehitu(que);
    Question galdera = db.find(Question.class, q.getQuestionNumber());
    db.persist(que);
    for(Quote k: galdera.getQuotes()) {
        Quote kuo= new Quote(k.getQuote(), k.getForecast(), que);
        que.listaraGehitu(kuo);
        db.persist(kuo);
    }
}
db.getTransaction().commit();
return true;
```

Descripción del Error y Acciones

Es un método con más de 15 líneas (“Write short units of code”) por lo que hemos querido simplificarlo lo máximo posible. En este método no hay mucho más que podamos eliminar más que la variable b y simplificar esos casos y hacer *return true/false*.

LINK AL COMMIT DE GITHUB

Miembro del Grupo

Cambios realizados por Xiomara G. Cáceres.

Refactorización #6

Código Inicial

```
public boolean ApustuaEgin(Registered u, Vector<Quote> quote, Double
balioa, Integer apustuBikoitzaGalarazi) {
	Registered user = (Registered) 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());
				Registered nork = erab.getNork();
				b=true;
				for(ApustuAnitza apu: nork.getApustuAnitzak()) {

if(apu.getApustuKopia()==apustuAnitza.getApustuKopia()) {
					b=false;
				}
				}
				if(b) {
					if(nork.getDiruLimitea()<balioa) {
						this.ApustuaEgin(nork, quote,
nork.getDiruLimitea(), apustuBikoitzaGalarazi);
					}else{
						this.ApustuaEgin(nork, quote, balioa,
apustuBikoitzaGalarazi);
					}
				}
			}
			return true;
		}else{
			return false;
		}
}
```

Código Refactorizado

```
Registered user = (Registered) db.find(Registered.class, u.getUsername());
if (user.getDirukop()<balioa) return false;

db.getTransaction().begin();
ApustuAnitza apustuAnitza = new ApustuAnitza(user, balioa);
db.persist(apustuAnitza);
```

```
        Registered nork = erab.getNork();
        Boolean b=true;
        for(ApustuAnitza apu: nork.getApustuAnitzak()) {
            if(apu.getApustuKopia()==apustuAnitza.getApustuKopia()) {
                b=false;
            }
        }
        if(b) {
            if(nork.getDiruLimitea()<balioa) {
                this.ApustuaEgin(nork, quote, nork.getDiruLimitea(), apustuBikoitzaGalarazi);
            }else{
                this.ApustuaEgin(nork, quote, balioa, apustuBikoitzaGalarazi);
            }
        }
    }
    return true;
```

Descripción del Error y Acciones

ApustuaEgin(...) es un método muy extenso, de más de 15 líneas "Write short units of code") por lo que sería oportuno minimizar el código al máximo. Lo único destacable que hemos eliminado ha sido la variable booleana *b* y hemos sustituido esas condiciones con *return true/false* que es más directo.

LINK AL COMMIT DE GITHUB

Miembro del Grupo

Cambios realizados por Xiomara G. Cáceres.

Refactorización #7

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 == false) {
        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 apl = db.find(ApustuAnitza.class, apustuAnitza.getApustuAnitzaNumber());
                db.getTransaction().begin();
                apl.removeApustua(quo.getApustuak().get(i));
                db.getTransaction().commit();
                if(apl.getApustuak().isEmpty() && !apl.getEgoera().equals("galduta")) {
                    this.apustuaEzabatu(apl.getUser(), apl);
                }else if(!apl.getApustuak().isEmpty() && apl.irabazitaMarkatu()){
                    this.ApustuaIrabazi(apl);
                }
                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);
    List<Question> listQ = event.getQuestions();
    for(Question q : listQ) {
        if(q.getResult() == null) {
            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));
            for(int i=0; i<quo.getApustuak().size(); i++) {
                ApustuAnitza apustuAnitza = quo.getApustuak().get(i).getApustuAnitza();
                ApustuAnitza apl = db.find(ApustuAnitza.class, apustuAnitza.getApustuAnitzaNumber());
                db.getTransaction().begin();
                apl.removeApustua(quo.getApustuak().get(i));
                db.getTransaction().commit();
                if(apl.getApustuak().isEmpty() && !apl.getEgoera().equals("galduta")) {
                    this.apustuaEzabatu(apl.getUser(), apl);
                }else if(!apl.getApustuak().isEmpty() && apl.irabazitaMarkatu()){
                    this.ApustuaIrabazi(apl);
                }
                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;
```

Descripción del Error y Acciones

El método gertaeraEzabatu() tenía una variable booleana (resultB) que se actualizaba a false en caso de que una de las preguntas del evento no tuviera respuesta, pues en ese caso, la función deberá terminar inmediatamente devolviendo false. Esto se hacía mediante una condición if dentro del bucle for que recorría la lista de preguntas del evento, sin embargo, en vez de hacer 'return false;' directamente dentro del bucle, fuera del mismo había otra condicional para que en el caso de que resultB fuera falso, se acabase la función mediante el uso de 'return false;'. Por lo tanto, el cambio que se ha hecho ha sido eliminar esa variable 'resultB', dentro del bucle for devolver falso directamente en el momento que se detecte que una pregunta no tiene respuesta y, se ha eliminado la condicional extra que había después del primer bucle. De esta forma, hemos conseguido reducir en uno la complejidad ciclomática del método gertaeraEzabatu().

LINK AL COMMIT DE GITHUB

Miembro del Grupo

Cambios realizados por Eider Fernández.