## "WRITE SHORT UNITS OF CODE"

1. **gertaerakSortu()** (autora: Jaione Gonzalez)

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;
}
```

2) 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) {
        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) {
            gsLaguntzaile(description, eventDate, spo);
        }
    }else {
        return false;
    }
    db.getTransaction().commit();
    return b;
}

public void gsLaguntzaile(String description, Date eventDate, Sport spo) {
    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);
}
```

Hemos empezado con un solo método de 22 líneas, y el objetivo es reducirlo a, como mínimo, 15. En este caso he creado un método auxiliar, porque aunque ahora tengamos dos métodos en vez de uno solo, ambos son más cortos y fáciles de entender. Con el nuevo método de 8 líneas de código, el gertaerakSortu() se queda en 15 líneas. Una vez llegue a la condición if(b), si el valor de esa variable es *true* saldrá del método para ejecutar el auxiliar.

2. **gertaerakKopiatu()** (autor: Abderraouf Khedidji)

1) 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;
            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);
                    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;
    }
```

2) Código refactorizado:

Hemos dividido el código en 3 métodos diferentes para poder visualizar de una forma más clara lo que hace. Empezó siendo un método de 27 líneas a 3 métodos de 10 líneas cada uno.

```
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 = gertaerakKopiatu2(gertaera, date);
    }
    db.getTransaction().commit();
    return b;
}
public boolean gertaerakKopiatu2(Event gertaera, Date date) {
    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);

        gertaerakKopiatu3(gertaera, gertKopiatu);
    return true;
}

public void gertaerakKopiatu3(Event gertaera, Event 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);
        }
    }
}
```

3. **emaitzakIpini()** (autor: Nestor Guimerans)

1) 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());
        }
    }
}
```

2) Código refactorizado:

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

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();


    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();
    IrabazitakoApustuakMarkatu(q);
}
```

En un principio tenemos un método de 23 líneas, del cual se ha extraído la parte donde actualizaba el resultado de los eventos. Así hemos separado el método original en dos métodos más cortos.

## "WRITE SIMPLE UNITS OF CODE"

1. **gertaeraEzabatu()** (autor: Abderraouf Khedidji)

1) Código inicial:

1.1 El if de la línea 1002 no tiene sentido pudiendo hacer un return en la condición que lo

```
992    public boolean gertaeraEzabatu(Event ev) {
993        Event event  = db.find(Event.class, ev);
994        boolean resultB = true;
995        List<Question> listQ = event.getQuestions();
996
                +1
997        1 for(Question q : listQ) {
                +2 (incl 1 for nesting)
998            2 if(q.getResult() == null) {
999                resultB = false;
1000           }
1001       }
                +1
1002       3 if(resultB == false) {
1003           return false;
               +1
1004       }else  4 if(new Date().compareTo(event.getEventDa
```

1.2 No tiene sentido que compruebe si apustuak no esté limpio ya que lo comprueba en la condición anterior

```
1013                    6 if(ap1.getApustuak().isEmpty()  7 && !ap1.getEgoera().equals("galduta")) {
1014                        this.apustuaEzabatu(ap1.getUser(), ap1);
                        +1 | +1
1015                    }else  8 if(!ap1.getApustuak().isEmpty()  9 && ap1.irabazitaMarkatu()){
1016                        this.ApustuaIrabazi(ap1);
1017                    }
```

2) Código refactorizado:

2.1 En caso de que se cumpla la condición del for returnea false directamente sin necesidad de usar una variable booleana

```
992    public boolean gertaeraEzabatu(Event ev) {
993        Event event  = db.find(Event.class, ev);
994        List<Question> listQ = event.getQuestions();
995
                +1
996        1 for(Question q : listQ) {
                +2 (incl 1 for nesting)
997            2 if(q.getResult() == null) {
998                return false;
999            }
1000       }
               +1
1001       3 if(new Date().compareTo(event.getEventDate())<0) {
1002           TypedQuery<Quote> Qquery = db.createQuery("SELECT q FROM
1003           Qquery.setParameter(1, event.getEventNumber());
1004           List<Quote> listQUO = Qquery.getResultList();
```

2.2 Quitamos la condición sobrante ya comprobada anteriormente

```
1013                    6 if(ap1.getApustuak().isEmpty()  7 && !ap1.getEgoera().equals("galduta")) {
1014                        this.apustuaEzabatu(ap1.getUser(), ap1);
                        +1 | +1
1015                    }else  8 if(ap1.irabazitaMarkatu()){
1016                        this.ApustuaIrabazi(ap1);
1017                    }
1018                    db.getTransaction().begin();
1019                    Sport spo =quo.getQuestion().getEvent().getSport();
1020                    spo.setApustuKantitatea(spo.getApustuKantitatea()-1);
1021                    db.getTransaction().commit();
```

2. **apustuaEgin()** (autores: Nestor Guimerans y Jaione Gonzalez)

Este método lo hemos hecho entre dos integrantes del grupo ya que aparte de no haber encontrado más métodos que refactorizar, era el más complejo de los dos que había.

1) Código inicial:

```
public boolean ApustuaEgin(ApustuaEginParameter parameterObject) {
    Integer apustuBikoitzaGalarazi = parameterObject.apustuBikoitzaGalarazi;
    Registered user = (Registered) db.find(Registered.class, parameterObject.u.getUsername());
    Boolean b;
      +1
     1 if(user.getDirukop()>=parameterObject.balioa) {
        db.getTransaction().begin();
        ApustuAnitza apustuAnitza = new ApustuAnitza(user, parameterObject.balioa);
        db.persist(apustuAnitza);
            +2 (incl 1 for nesting)
         2 for(Quote quo: parameterObject.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();
        +2 (incl 1 for nesting)
        if(apu 3 stuBikoitzaGalarazi==-1) {
            apustuBikoitzaGalarazi=apustuAnitza.getApustuAnitzaNumber();
        }
        apustuAnitza.setApustuKopia(apustuBikoitzaGalarazi);
        user.updateDiruKontua(-parameterObject.balioa);
        Transaction t = new Transaction(user, parameterObject.balioa, new Date(), "ApustuaEgin");
        user.addApustuAnitza(apustuAnitza);
        +2 (incl 1 for nesting)
        for(Ap 4 ustua 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();
        +2 (incl 1 for nesting)
        for(la 5 rraitzailea reg:user.getJarraitzaileLista()) {
            Jarraitzailea erab=db.find(Jarraitzailea.class, reg.getJarraitzaileaNumber());
            b=true;
            +3 (incl 2 for nesting)
            for(Ap 6 ustuAnitza apu: erab.getNork().getApustuAnitzak()) {
                +4 (incl 3 for nesting)
                if(apu 7 .getApustuKopia().equals(apustuAnitza.getApustuKopia())) {
                    b=false;
                }//
            }
            +3 (incl 2 for nesting)
            if(b) 8 {
                +4 (incl 3 for nesting)
                if(era 9 b.getNork().getDiruLimitea()<parameterObject.balioa) {
                    this.ApustuaEgin(new ApustuaEginParameter(erab.getNork(), parameterObject.quote, erab.getNork().getDiruLimitea(), apustuBikoitzaGalarazi));
                +1
                }else{
                    this.ApustuaEgin(new ApustuaEginParameter(erab.getNork(), parameterObject.quote, parameterObject.balioa, apustuBikoitzaGalarazi));
                }
            }
        }
        return true;
    +1
    }else{
        return false;
    }
}
```

2) Código refactorizado:

```
public boolean ApustuaEgin(ApustuaEginParameter parameterObject) {
    Integer apustuBikoitzaGalarazi = parameterObject.apustuBikoitzaGalarazi;
    Registered user = (Registered) db.find(Registered.class, parameterObject.u.getUsername());
    if(user.getDirukop()>=parameterObject.balioa) {
        db.getTransaction().begin();
        ApustuAnitza apustuAnitza = new ApustuAnitza(user, parameterObject.balioa);
        db.persist(apustuAnitza);
        for(Quote quo: parameterObject.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(-parameterObject.balioa);
        Transaction t = new Transaction(user, parameterObject.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());
            for(ApustuAnitza apu: erab.getNork().getApustuAnitzak()) {

                if(apu.getApustuKopia().equals(apustuAnitza.getApustuKopia())) {
                    return true;
                }
            }

            ApustuaEginParameter apustua = new ApustuaEginParameter(erab.getNork(), parameterObject.quote, parameterObject.balioa, apustuBikoitzaGalarazi);
            if(erab.getNork().getDiruLimitea()<parameterObject.balioa) {

                apustua.balioa = erab.getNork().getDiruLimitea();
            }
            this.ApustuaEgin(apustua);
        }
        return true;
    }
    return false;
}
```

Hemos logrado reducir los caminos de este método de 25 a 19, pese a no ser lo ideal (lo cual sería 15), es lo máximo que hemos conseguido optimizar el código. Esto lo hemos hecho eliminar el mayor número de *else* posible, ya que en algunos casos eran inútiles (en el caso de la variable booleana, que sólo determinaba si el programa directamente hacía return *true,* por lo que directamente hemos añadido el return true en el if para ahorrar el else).

## "DUPLICATE CODE"

1. **String "apustua egin"** (autora: Jaione Gonzalez)

1) Código inicial:

```
Transaction t1 = new Transaction(reg1, apA1.getBalioa(), new Date(), "ApustuaEgin");
Transaction t3 = new Transaction(reg2, apA4.getBalioa(), new Date(), "ApustuaEgin");
Transaction t4 = new Transaction(reg3, apA5.getBalioa(), new Date(), "ApustuaEgin");
Transaction t5 = new Transaction(reg4, apA3.getBalioa(), new Date(), "ApustuaEgin");
Transaction t6 = new Transaction(reg4, apA6.getBalioa(), new Date(), "ApustuaEgin");
Transaction t7 = new Transaction(reg1, apA7.getBalioa(), new Date(), "ApustuaEgin");
Transaction t8 = new Transaction(reg1, apA8.getBalioa(), new Date(), "ApustuaEgin");
Transaction t9 = new Transaction(reg2, apA9.getBalioa(), new Date(), "ApustuaEgin");
Transaction t10 = new Transaction(reg2, apA10.getBalioa(), new Date(), "ApustuaEgin");
Transaction t11 = new Transaction(reg3, apA11.getBalioa(), new Date(), "ApustuaEgin");
Transaction t12 = new Transaction(reg3, apA12.getBalioa(), new Date(), "ApustuaEgin");
```

2) Código refactorizado:

```
String a = "ApustuaEgin";

Transaction t1 = new Transaction(reg1, apA1.getBalioa(), new Date(), a);
Transaction t3 = new Transaction(reg2, apA4.getBalioa(), new Date(), a);
Transaction t4 = new Transaction(reg3, apA5.getBalioa(), new Date(), a);
Transaction t5 = new Transaction(reg4, apA3.getBalioa(), new Date(), a);
Transaction t6 = new Transaction(reg4, apA6.getBalioa(), new Date(), a);
Transaction t7 = new Transaction(reg1, apA7.getBalioa(), new Date(), a);
Transaction t8 = new Transaction(reg1, apA8.getBalioa(), new Date(), a);
Transaction t9 = new Transaction(reg2, apA9.getBalioa(), new Date(), a);
Transaction t10 = new Transaction(reg2, apA10.getBalioa(), new Date(), a);
Transaction t11 = new Transaction(reg3, apA11.getBalioa(), new Date(), a);
Transaction t12 = new Transaction(reg3, apA12.getBalioa(), new Date(), a);
```

En este caso tan sólo hemos sustituido el parámetro "Apustua Egin" por *a* una variable que hemos definido justo antes, de esta forma, utilizamos la variable varias veces en lugar de escribir el mismo string una y otra vez.

2. **String "¿Quién ganará el partido?"** (autor: Abderraouf Khedidji)

1) Código inicial:

```
if (Locale.getDefault().equals(new Locale("es"))) {
    Duplication
    q1=ev1.addQuestion(1,"Â¿QuiÃ©n ganarÃ¡ el partido?",1);
    q2=ev1.addQuestion("Â¿QuiÃ©n meterÃ¡ el primer gol?",2);
    Duplication
    q3=ev11.addQuestion(2,"Â¿QuiÃ©n ganarÃ¡ el partido?",1);
    q4=ev11.addQuestion("Â¿CuÃ¡ntos goles se marcarÃ¡n?",2);
    Duplication
    q5=ev17.addQuestion(3,"Â¿QuiÃ©n ganarÃ¡ el partido?",1);
    q6=ev17.addQuestion("Â¿HabrÃ¡ goles en la primera parte?",2);

}
```

Esto se repite con el addQuestion en euskera y en inglés

2) Código refactorizado:

Lo corregimos en todas las preguntas usando una única variable para cada pregunta que queramos añadir para no generar esa repetición de código

```
        q4=ev11.addQuestion("Â¿cuÃ¡ntos goles se marcarÃ¡n?",2);
        q5=ev17.addQuestion(pregunta1,1);
        q6=ev17.addQuestion("Â¿HabrÃ¡ goles en la primera parte?",2);

    }

    else if (Locale.getDefault().equals(new Locale("en"))) {
        String pregunta2 = "Who will win the match?";
        q1=ev1.addQuestion(pregunta2,1);
        q2=ev1.addQuestion("Who will score first?",2);
        q3=ev11.addQuestion(pregunta2,1);
        q4=ev11.addQuestion("How many goals will be scored in the match?",2);
        q5=ev17.addQuestion(pregunta2,1);
        q6=ev17.addQuestion("Will there be goals in the first half?",2);

    }
    else {
        String pregunta3 = "Zeinek irabaziko du partidua?";
        q1=ev1.addQuestion(pregunta3,1);
        q2=ev1.addQuestion("Zeinek sartuko du lehenengo gola?",2);
        q3=ev11.addQuestion(pregunta3,1);
        q4=ev11.addQuestion("Zenbat gol sartuko dira?",2);
```

3. **String "DiruaSartu"** (autor: Nestor Guimerans)

1) Código inicial:

```
595
                Duplication
596             this.DiruaSartu(reg1, 50.0, new Date(), 1 "DiruaSartu");
                Duplication
597             this.DiruaSartu(reg2, 50.0, new Date(), 2 "DiruaSartu");
                Duplication
598             this.DiruaSartu(reg3, 50.0, new Date(), 3 "DiruaSartu");
                Duplication
599             this.DiruaSartu(reg4, 50.0, new Date(), 4 "DiruaSartu");
600
```

2) Código refactorizado:

```
String s = "DiruaSartu";
Duplication
this.DiruaSartu(reg1, 50.0, new Date(), s);
Duplication
this.DiruaSartu(reg2, 50.0, new Date(), s);
Duplication
this.DiruaSartu(reg3, 50.0, new Date(), s);
Duplication
this.DiruaSartu(reg4, 50.0, new Date(), s);
```

En este fragmento de código hemos convertido el texto duplicado en un "String" para meter un mismo parámetro en vez de el mismo texto repetido 4 veces.

## "KEEP UNIT INTERFACES SMALL"

1. **jarraitu()** (autora: Jaione Gonzalez)

1) Código inicial:

```
public boolean jarraitu(Registered jabea, Registered jarraitua, Double limit) {
    Boolean b=false;
    Registered jarraitu = (Registered) db.find(Registered.class, jarraitua.getUsername());
    Registered harpideduna = (Registered) db.find(Registered.class, jabea.getUsername());
    if(!harpideduna.getJarraitutakoLista().contains(jarraitu)) {
        db.getTransaction().begin();
        Jarraitzailea jar = new Jarraitzailea(harpideduna, jarraitu);
        harpideduna.addJarraitutako(jar);
        jarraitu.addJarraitzailea(jar);
        b=true;
        db.persist(jar);
        harpideduna.setDiruLimitea(limit);
        db.getTransaction().commit();
    }
    return b;
}
```

2) Código refactorizado:

```
public boolean jarraitu(JarraituParameter parameterObject) {
    Boolean b=false;
    Registered jarraitu = (Registered) db.find(Registered.class, parameterObject.jarraitua.getUsername());
    Registered harpideduna = (Registered) db.find(Registered.class, parameterObject.jabea.getUsername());
    if(!harpideduna.getJarraitutakoLista().contains(jarraitu)) {
        db.getTransaction().begin();
        Jarraitzailea jar = new Jarraitzailea(harpideduna, jarraitu);
        harpideduna.addJarraitutako(jar);
        jarraitu.addJarraitzailea(jar);
        b=true;
        db.persist(jar);
        harpideduna.setDiruLimitea(parameterObject.limit);
        db.getTransaction().commit();
    }
    return b;
}
```

```
public class JarraituParameter {
    public Registered jabea;
    public Registered jarraitua;
    public Double limit;

    public JarraituParameter(Registered jabea, Registered jarraitua, Double limit) {
        this.jabea = jabea;
        this.jarraitua = jarraitua;
        this.limit = limit;
    }
}
```

En este caso he optado por crear una clase nueva, cuyos parámetros sean los que previamente se le metían al método, y de esta forma solo se le proporciona como parámetro un solo objeto. (He utilizado un método con tres parámetros porque no he encontrado ninguno que tuviera más, pero el procedimiento sería el mismo).

2. **ApustuaEgin()** (autor: Abderraouf Khedidji)

1) Código inicial:

En el caso del siguiente método nos encontramos con que tiene demasiados parámetros que hacen que el método sea difícil de manejar.

```
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();
```

2) Código refactorizado:

Una solución sería meter por parámetro un objeto que tenga por atributos los parámetros que tenía de antes.

```
public boolean ApustuaEgin(ApustuaEginParameter parameterObject) {
    Integer apustuBikoitzaGalarazi = parameterObject.apustuBikoitzaGalarazi;
    Registered user = (Registered) db.find(Registered.class, parameterObject.u.getUsername());
    Boolean b;
    if(user.getDirukop()>=parameterObject.balioa) {
        db.getTransaction().begin();
        ApustuAnitza apustuAnitza = new ApustuAnitza(user, parameterObject.balioa);
        db.persist(apustuAnitza);
        for(Quote quo: parameterObject.quote) {
            Quote kuote = db.find(Quote.class, quo.getQuoteNumber());
            Apustua ap = new Apustua(apustuAnitza, kuote);
            db.persist(ap);
            apustuAnitza.addApustua(ap);
```

```
package dataAccess;

import java.util.Vector;

public class ApustuaEginParameter {
    public Registered u;
    public Vector<Quote> quote;
    public Double balioa;
    public Integer apustuBikoitzaGalarazi;

    public ApustuaEginParameter(Registered u, Vector<Quote> quote, Double balioa, Integer apustuBikoitzaGalarazi) {
        this.u = u;
        this.quote = quote;
        this.balioa = balioa;
        this.apustuBikoitzaGalarazi = apustuBikoitzaGalarazi;
    }
```

3. **DiruaSartu()**(autor: Nestor Guimerans)

1) Código inicial:

```
public void DiruaSartu(Registered u, Double dirua, Date data, String mota) {
    Registered user = (Registered) db.find(Registered.class, u.getUsername());
    db.getTransaction().begin();
    Transaction t = new Transaction(user, dirua, data, mota);
    System.out.println(t.getMota());
    user.addTransaction(t);
    user.updateDiruKontua(dirua);
    db.persist(t);
    db.getTransaction().commit();
}
```

2) Código refactorizado:

```java
package dataAccess;

import java.util.Date;

public class DiruaSartuParameter {
    public Registered u;
    public Double dirua;
    public Date data;
    public String mota;

    public DiruaSartuParameter(Registered u, Double dirua, Date data, String mota) {
        this.u = u;
        this.dirua = dirua;
        this.data = data;
        this.mota = mota;
    }
}
```

```java
public void DiruaSartu(DiruaSartuParameter parameterObject) {
    Registered user = (Registered) db.find(Registered.class, parameterObject.u.getUsername());
    db.getTransaction().begin();
    Transaction t = new Transaction(user, parameterObject.dirua, parameterObject.data, parameterObject.mota);
    System.out.println(t.getMota());
    user.addTransaction(t);
    user.updateDiruKontua(parameterObject.dirua);
    db.persist(t);
    db.getTransaction().commit();
}
```

En vez de meter 4 parámetros diferentes, creamos un objeto que contenga estos parámetros y lo metemos como parámetro único.