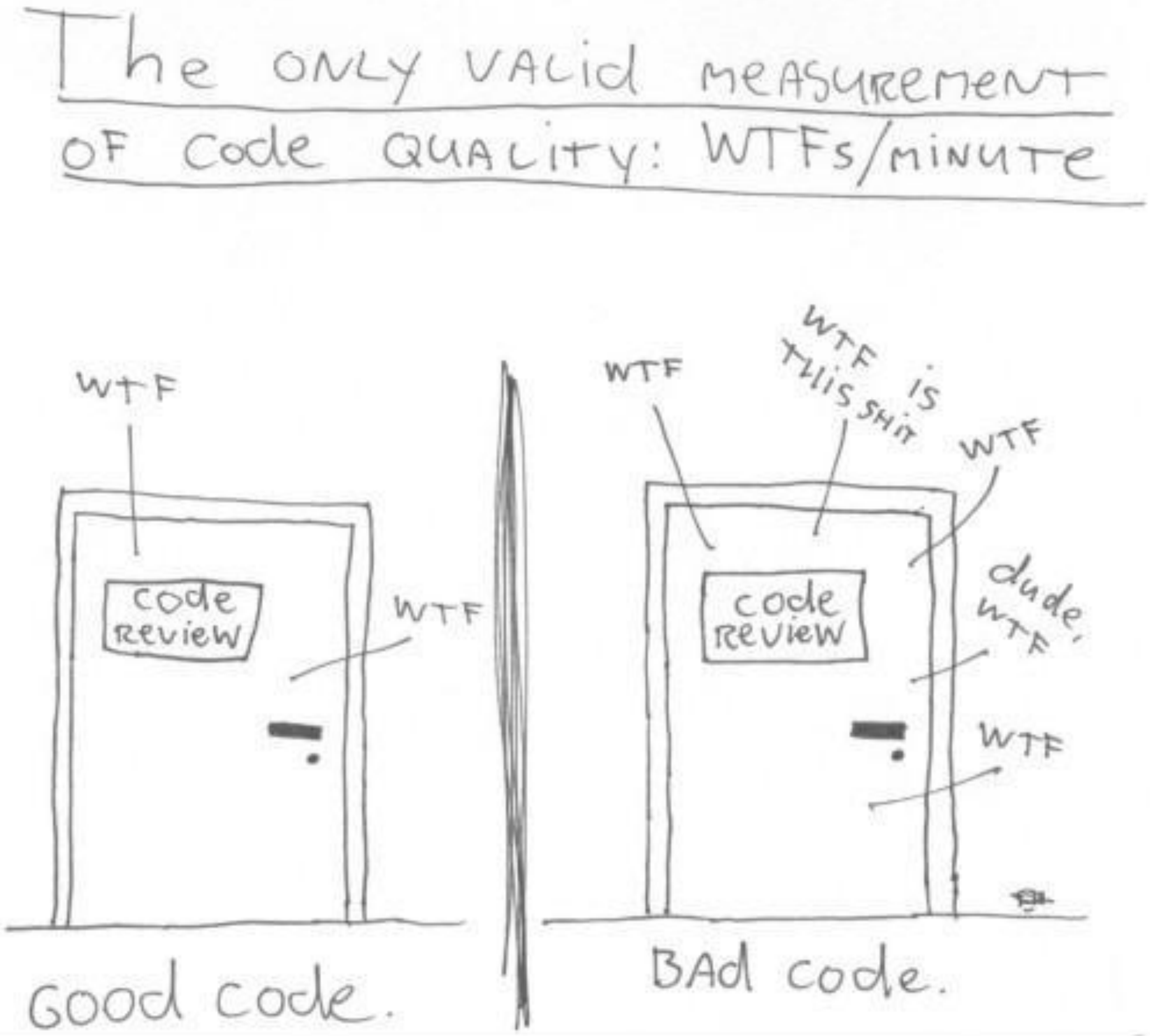
FACULTAD DE INGENIERIA
Universidad de Buenos Aires

Refactoring

FI.UBA

75.10 - Técnicas de Diseño


The only valid measurement of code quality: WTFs/minute
WTF
code review
WTF
Good code.
WTF
WTF is this shit
WTF
code review
dude, WTF
WTF
Bad code.

What is Refactoring?

A **technique** for **restructuring** an existing body of **code**, **altering** its **internal structure** **without changing** its **external behavior**

How to achieve it?

- Unit tests to guarantee the external behavior has not been changed

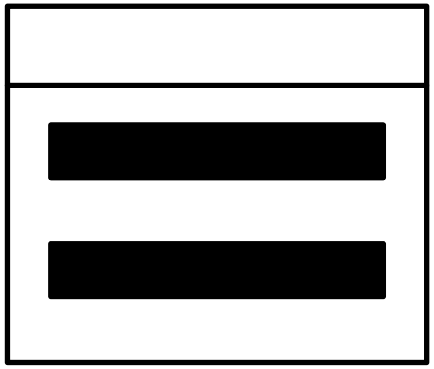
- Applying the proposed refactorings

Refactoring Flow

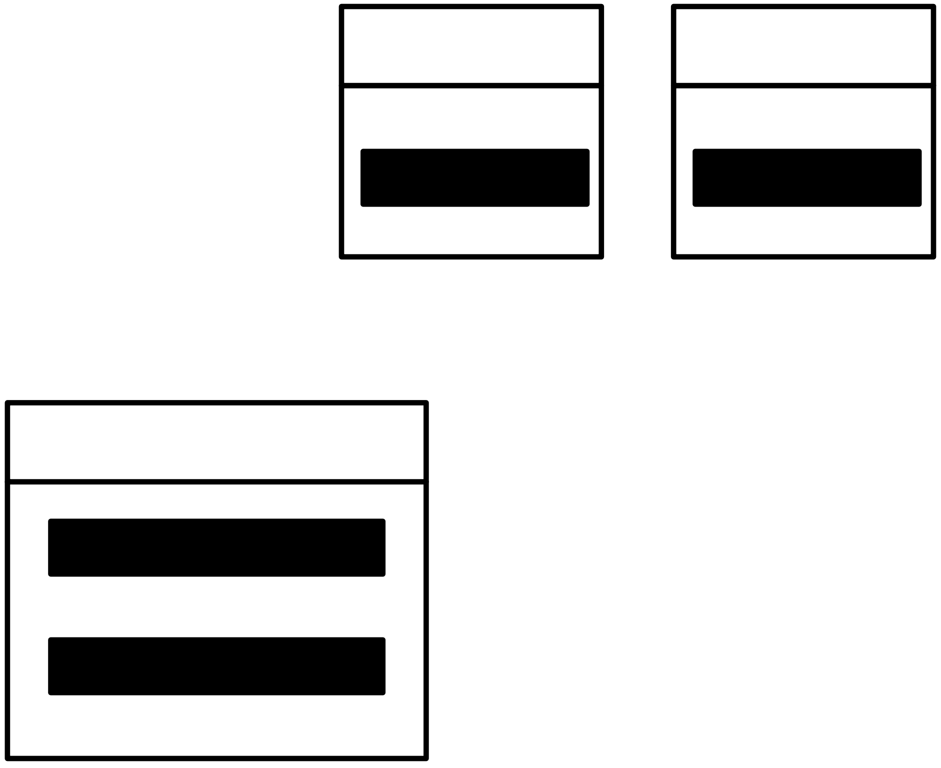
- Ensure all tests pass
- Find code that smells
- Find refactoring
- Apply refactoring

Code Smells

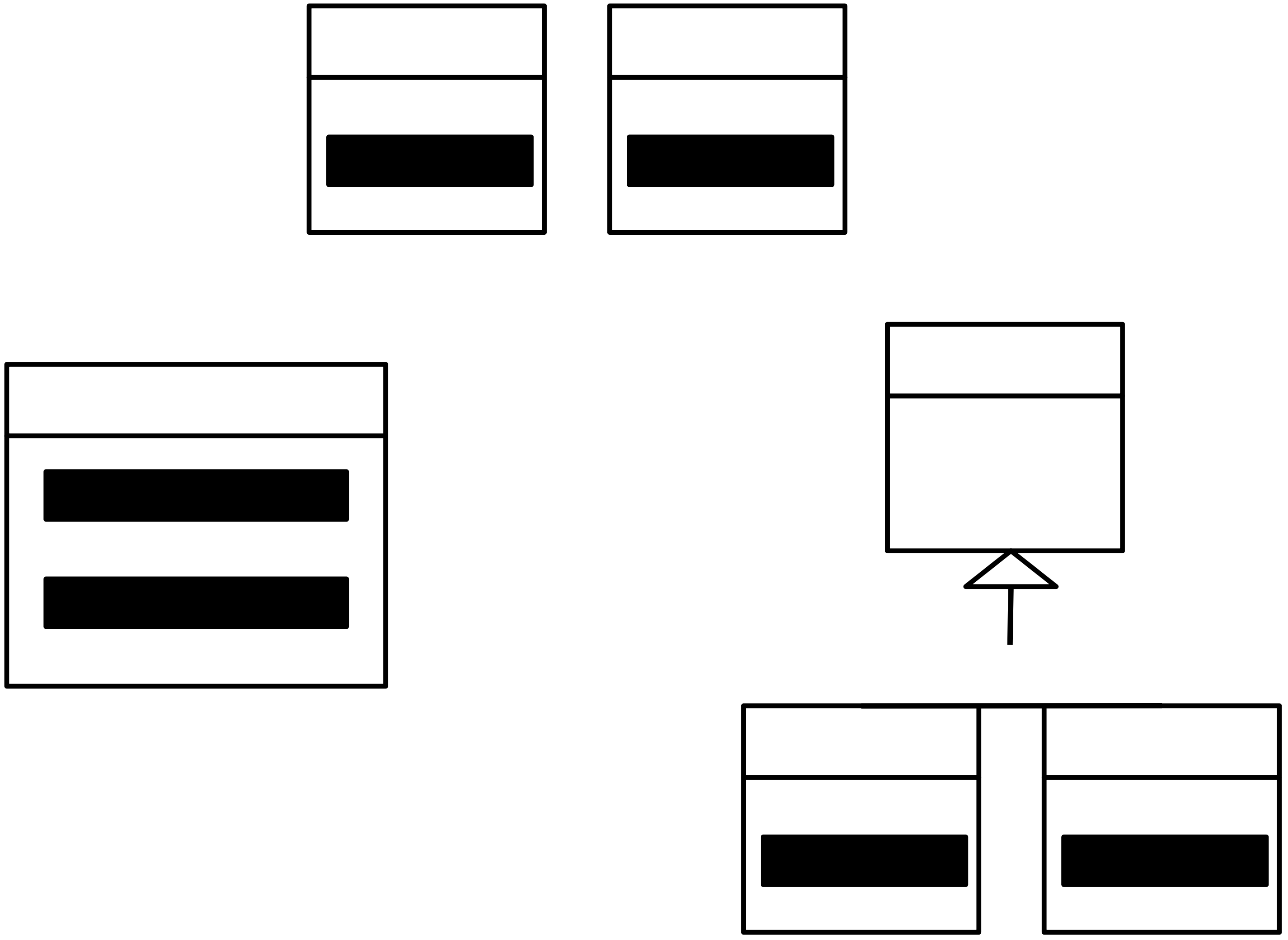
Duplicated Code



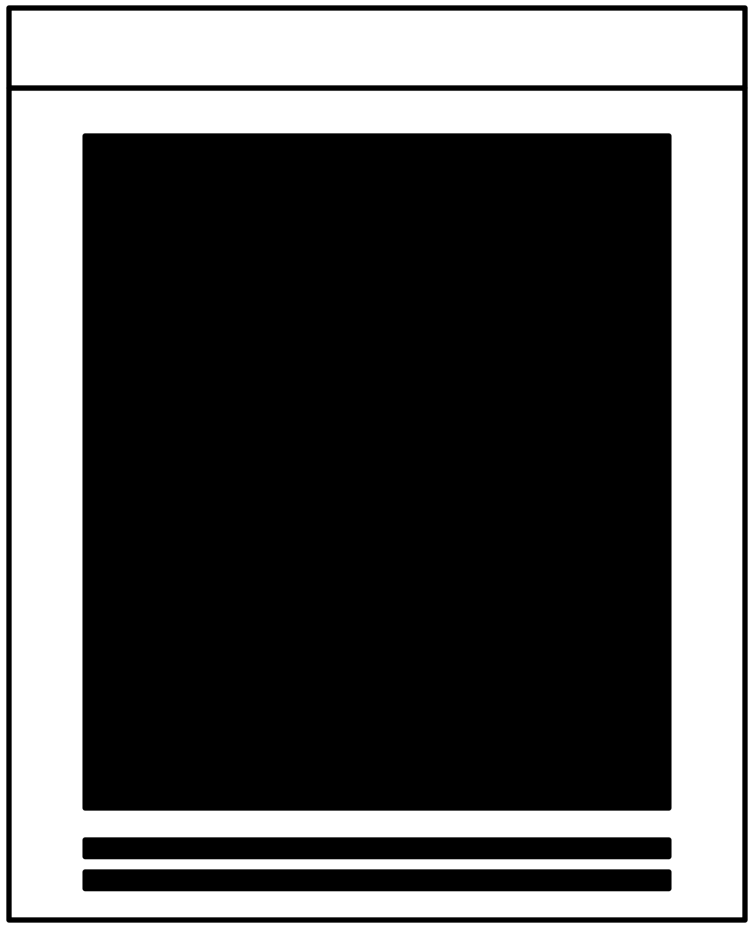
Duplicated Code



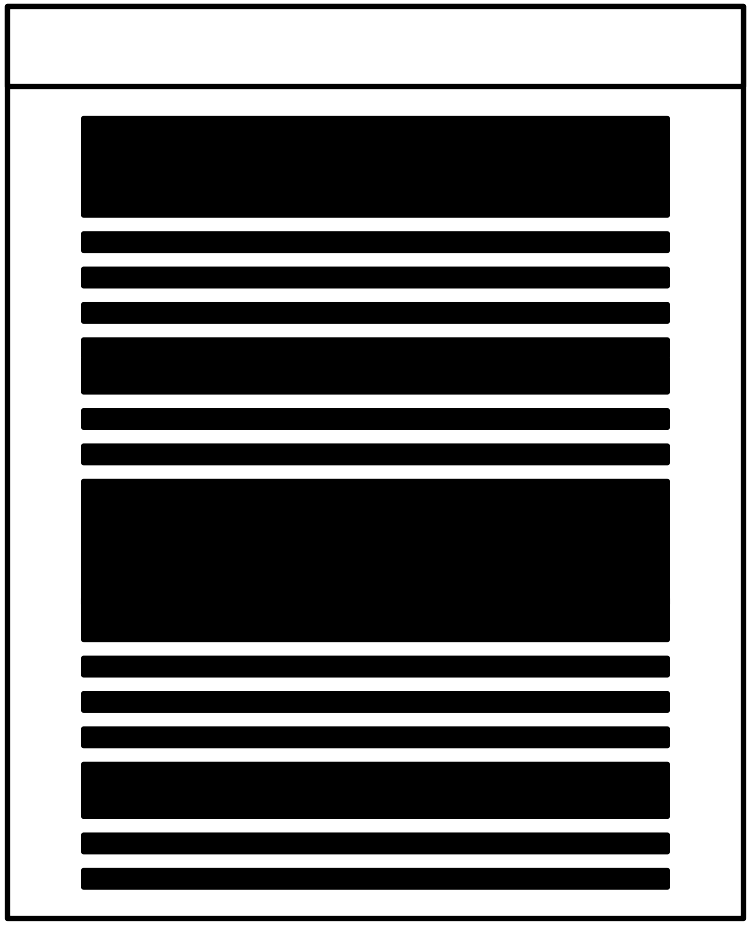
Duplicated Code



Long Method



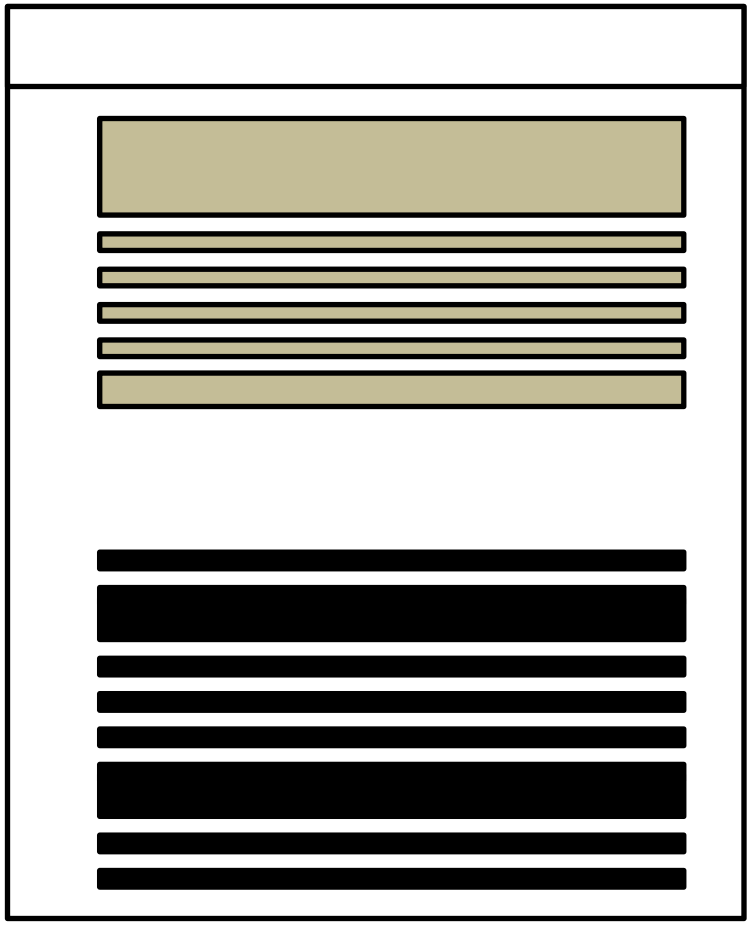
Large Class



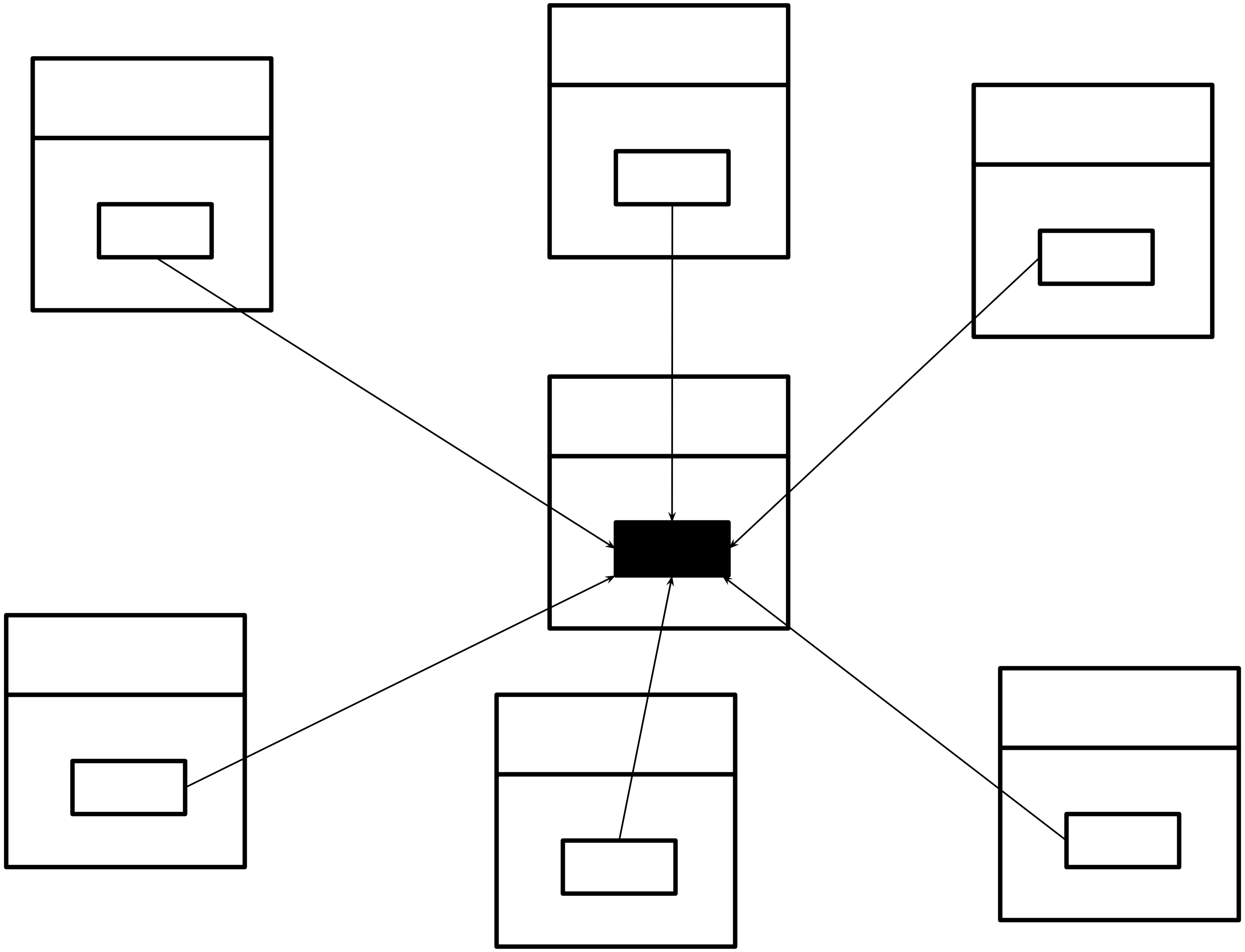
Long parameter list

xxxxxxx(■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■)

Divergent Change



Shotgun Surgery



feature envy

```
class CapitalCalculator {
 ...

 public double capital(Loan loan) {
  if (loan.getExpiry() == null && loan.getMaturity() != null)
   return loan.getCommitment()*loan.duration()*loan.riskFactor();

  if (loan.getExpiry() != null && loan.getMaturity() == null) {
    if (loan.getUnusedPercentage() != 1.0)
       return loan.getCommitment() * loan.getUnusedPercentage() *
                loan.duration() * loan.riskFactor();
    else
       return (loan.outstandingRiskAmount()*loan.duration()
                  * loan.riskFactor())
      + (loan.unusedRiskAmount() * loan.duration()
                  * loan.unusedRiskFactor());
   }

   return 0.0;
 }
 ...
}
```

```
class CapitalCalculator {
 ...

 public double capital(Loan loan) {
  if (loan.getExpiry() == null && loan.getMaturity() != null)
   return loan.getCommitment()*loan.duration()*loan.riskFactor();

  if (loan.getExpiry() != null && loan.getMaturity() == null) {
    if (loan.getUnusedPercentage() != 1.0)
       return loan.getCommitment() * loan.getUnusedPercentage() *
                 loan.duration() * loan.riskFactor();
    else
       return (loan.outstandingRiskAmount()*loan.duration()
                  * loan.riskFactor())
      + (loan.unusedRiskAmount() * loan.duration()
                  * loan.unusedRiskFactor());
   }

   return 0.0;
 }
 ...
}
```

Data Clumps

method1(■ ▲ ◆ ⬬)

method2(■ ▲ ◆ ⬬)

method3(■ ▲ ◆ ⬬)

method4(■ ▲ ◆ ⬬)

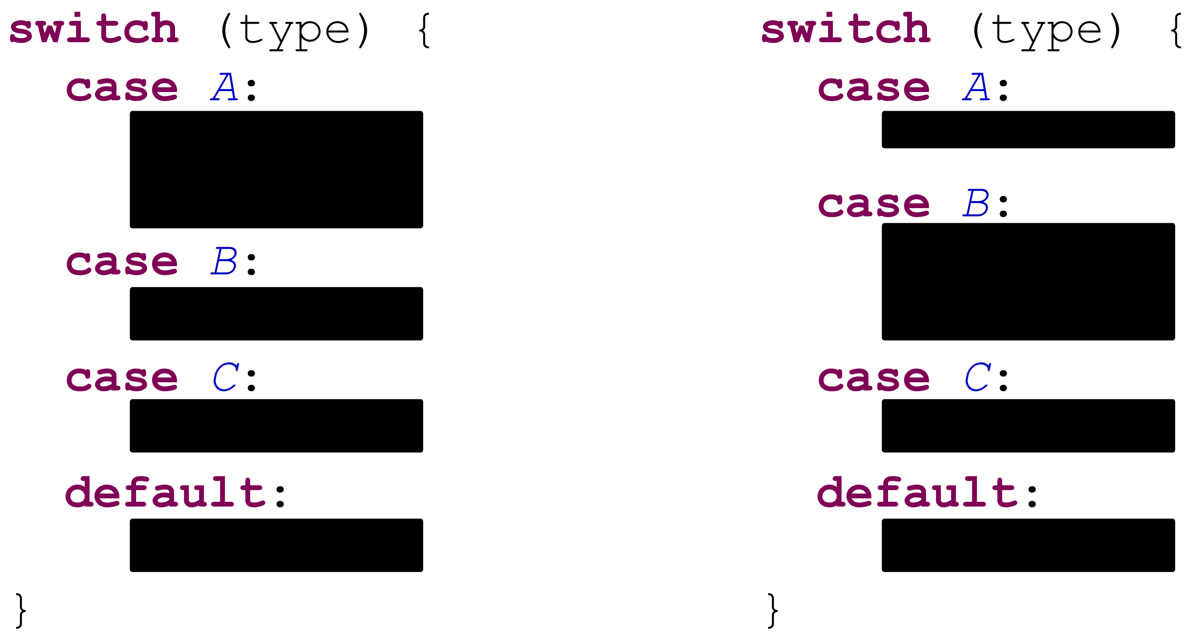
Primitive Obsession

```
double money;
String phone;
String zipCode;
String password;
```

Switch Statements

```
switch (type) {
  case A:
    [redacted]
  case B:
    [redacted]
  case C:
    [redacted]
  default:
    [redacted]
}
```

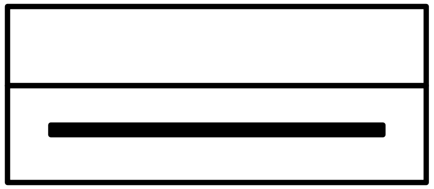
Switch Statements



```
switch (type) {
  case A:

  case B:

  case C:

  default:

}
```

```
switch (type) {
  case A:

  case B:

  case C:

  default:

}
```

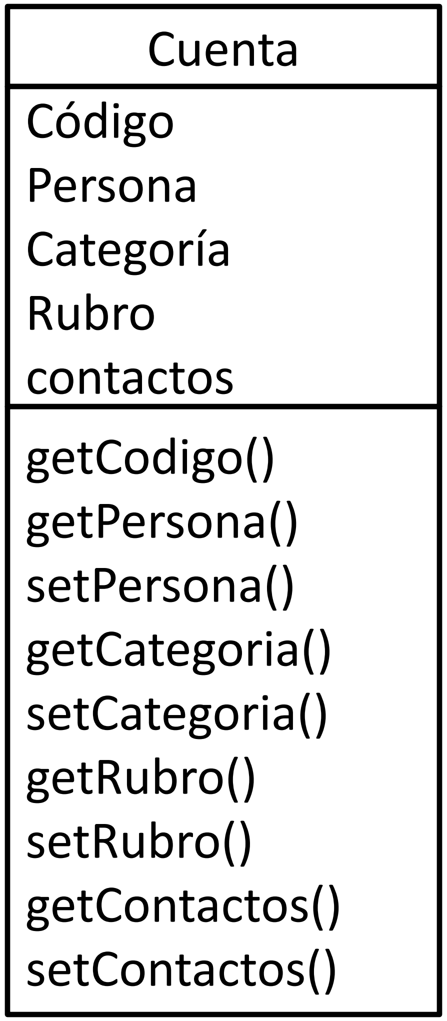
Lazy Class



Message Chains

```
■.■().■().■().■()
```

Data Class


Cuenta
Código
Persona
Categoría
Rubro
contactos
getCodigo()
getPersona()
setPersona()
getCategoria()
setCategoria()
getRubro()
setRubro()
getContactos()
setContactos()

Refactorings

1 38055 65154 7

```
if ((platform.toUpperCase().indexOf("MAC") > -1)
   && (platform.toUpperCase().indexOf("IE") > -1)
   && wasInitialized()
   && resize > 0) {

  someCode();

}

otherCode();
```

Introduce Explaining Method

```
if (isMacOs()
    && isIEBrowser()
    && wasInitialized()
    && wasResized()) {

      someCode();
}

otherCode();
```

```
if (isMacOs()
    && isIEBrowser()
    && wasInitialized()
    && wasResized()) {

      someCode();
}

otherCode();


if (isPlatformSupported()
    && wasInitialized()
    && wasResized()) {

    someCode();
}

otherCode();
```

```java
boolean wasResized() {
   return resize > 0;
}

boolean isIEBrowser() {
   return platform.toUpperCase().indexOf("IE") > -1;
}

boolean isMacOs() {
   return platform.toUpperCase().indexOf("MAC") > -1;
}

boolean isPlatformSupported() {
   return isMacOs() && isIEBrowser();
}
```

1 38055 65154 7

```java
double getDistanceTravelled (int time) {
    double result;

    double acc = primaryForce / mass;
    int primaryTime = Math.min(time, delay);
    result = 0.5 * acc * primaryTime * primaryTime;
    int secondaryTime = time - delay;

    if (secondaryTime > 0) {
        double primaryVel = acc * delay;
        acc = (primaryForce + secondaryForce) / mass;
        result += primaryVel * secondaryTime + 0.5 * acc *
secondaryTime * secondaryTime;
    }

    return result;
}
```

```
double getDistanceTravelled (int time) {
    double result;
    double acc = primaryForce / mass;
    int primaryTime = Math.min(time, delay);
    result = 0.5 * acc * primaryTime * primaryTime;
    int secondaryTime = time - delay;

    if (secondaryTime > 0) {
        double primaryVel = acc * delay;
        acc = (primaryForce + secondaryForce) / mass;
        result += primaryVel * secondaryTime + 0.5 * acc *
secondaryTime * secondaryTime;
    }

    return result;
}
```

Split Temporary Variable

```java
double getDistanceTravelled(int time) {
    double result;

    double primaryAcc = primaryForce / mass;
    int primaryTime = Math.min(time, delay);
    result = 0.5 * primaryAcc * primaryTime * primaryTime;
    int secondaryTime = time - delay;

    if (secondaryTime > 0) {
        double primaryVel = primaryAcc * delay;
        double secondaryAcc = (primaryForce + secondaryForce) / mass;
        result += primaryVel * secondaryTime + 0.5
         * secondaryAcc * secondaryTime * secondaryTime;
    }

    return result;
}
```

1 38055 65154 7

```java
class Page {
      private String[] lines;

      private double widthNumber;
      private String widthUnits;

      private double heightNumber;
      private String heightUnits;

      /**
       * return the page area in inches.
       */
      public double area() {
         double widthInches;
         double heightInches;
         widthInches = widthNumber *
              ((widthUnits.equals("mm")) ? 25.4 : 1.0);
         heightInches = heightNumber *
              ((heightUnits.equals("mm")) ? 25.4 : 1.0);
         return widthInches * heightInches;
      }

      ...
   }
```

Extract Class

```
class Page {
      private String[] lines;

      private Length width;
      private Length height;

      public Length area() {
         return width.multipliedBy(height);
      }
}
```

```java
class Length {

      private final double magnitude;
      private final Unit unit;

      public Length(Unit unit, double magnitude) {
         this.unit = unit;
         this.magnitude = magnitude;
      }

      private static Length newInInches(double magnitudeInInches) {
         return new Length(Unit.inches, magnitudeInInches);
      }

      public Length multipliedBy(Length aLength) {
         return Length.newInInches(this.magnitudeInInches()
            + aLength.magnitudeInInches());
      }

      private double magnitudeInInches() {
         return magnitude;
      }

      private double magnitudeInMM() {
         return magnitude * Unit.mmFactor();
      }
}
```

1 38055 65154 7

```
double chargeFor(Date date, int quantity) {
      double totalCharge = 0;
      if (date.after(WINTER_START) && date.before(WINTER_END)) {
         totalCharge = quantity * WINTER_RATE
             + WINTER_SERVICE_CHARGE;
      } else {
         totalCharge = quantity * NORMAL_RATE;
      }
      return totalCharge;
}
```

Decompose Conditional

```
double chargeFor(Date date, int quantity) {
      if (isAWinter(date)) {
         return winterCharge(quantity);
      }
      return normalCharge(quantity);
}
```

```
double chargeFor(Date date, int quantity) {
      return isAWinter(date) ?
      winterCharge(quantity) : normalCharge(quantity);
}
```

```java
private boolean isAWinter(Date date) {
    return date.after(WINTER_START) && date.before(WINTER_END);
}

private double normalCharge(int quantity) {
    return quantity * NORMAL_RATE;
}

private double winterCharge(int quantity) {
    return quantity * WINTER_RATE + WINTER_SERVICE_CHARGE;
}
```

```java
class YXZPaginator {
   ...

   public List<Element> nextFrom(int offset, int size) {
      if (((offset + size) > (cacheOffset + cacheSize))
        || (offset < cacheOffset)) {

         repopulateCache(offset, size);
      }

      return nextFromCache(offset, size);
    }

    ...
}
```

```java
public List<Element> nextFrom( int offset,  int size) {
  if (cacheNeedsRePopulation(offset, size)) {
     repopulateCache(offset, size);
  }

  return nextFromCache(offset, size);
}
```

```
public List<Element> nextFrom( int offset,  int size) {
  if (cacheNeedsRePopulation(offset, size)) {
     repopulateCache(offset, size);
  }

  return nextFromCache(offset, size);
}



boolean cacheNeedsRePopulation(int offset,  int size) {
  return requestIsUnderCacheWindow(offset, size)
            || requestIsOverCacheWindow(offset);
}
```

```java
public List<Element> nextFrom( int offset,  int size) {
  if (cacheNeedsRePopulation(offset, size)) {
     repopulateCache(offset, size);
  }

  return nextFromCache(offset, size);
}



boolean cacheNeedsRePopulation(int offset,  int size) {
  return requestIsUnderCacheWindow(offset, size)
            || requestIsOverCacheWindow(offset);
}

boolean requestIsOverCacheWindow(int offset) {
  return offset < cacheOffset;
}

boolean requestIsUnderCacheWindow(int offset,int size) {
  return (offset + size) > (cacheOffset + cacheSize);
}
```

1 38055 65154 7

```
class TicTacToeGame {

    boolean isGameOver() {
            if (allPositionsAreFilled()) {
                return true;
            }
            if (oneRowIsFilledByOnePlayer()) {
                return true;
            }
            if (oneColumnIsFilledByOnePlayer()) {
                return true;
            }
            if (oneDiagonalIsFilledByOnePlayer()) {
                return true;
            }
            return false;
        }

}
```

Consolidate conditional expression

```
boolean isGameOver() {
    if (allPositionsAreFilled()
            || oneRowIsFilledByOnePlayer()
            || oneColumnIsFilledByOnePlayer()
            || oneDiagonalIsFilledByOnePlayer()) {
        return true;
    }
    return false;
}
```

```
boolean isGameOver() {
    if (allPositionsAreFilled()
            || oneRowIsFilledByOnePlayer()
            || oneColumnIsFilledByOnePlayer()
            || oneDiagonalIsFilledByOnePlayer()) {
        return true;
    }
    return false;
}
```

```
boolean isGameOver() {
    return allPositionsAreFilled()
            || oneRowIsFilledByOnePlayer()
            || oneColumnIsFilledByOnePlayer()
            || oneDiagonalIsFilledByOnePlayer();
}
```

```
public double getRate() {
        if (onVacation()) {
            if (lengthOfService() > 10) {
                return 1;
            }
        }
        return 0.5;
    }
```

```
public double getRate() {
    if (onVacation() && lengthOfService() > 10) {
        return 1;
    }
    return 0.5;
}

public double getRate() {
    return (onVacation() && lengthOfService() > 10) ? 1 : 0.5;
}
```

1 38055 65154 7

```java
public double finalPrice(double price) {
      double total = 0;
      if (isSpecialDeal()) {
         total = price * 0.95;
         changed();
      } else {
         total = price;
         changed();
      }
      return total;
}
```

```
public double finalPrice(double price) {
      double total = 0;
      if (isSpecialDeal()) {
         total = price * 0.95;
         changed();
      } else {
         total = price;
         changed();
      }
      return total;
}
```

Consolidate duplicate conditional fragments

```java
public double finalPrice(double price) {
      double total = 0;
      if (isSpecialDeal()) {
         total = price * 0.95;
      } else {
         total = price;
      }
      changed();
      return total;
}
```

1 38055 65154 7

```java
public boolean exist(String nameToFind) {
      boolean found = false;
      for (String name : names) {
         if (name.equals(nameToFind)) {
            found = true;
         }
      }
      return found;
}
```

Remove control flag

```java
public boolean exist(String nameToFind) {
      for (String name : names) {
         if (name.equals(nameToFind)) {
            return true;
         }
      }
      return false;
}
```

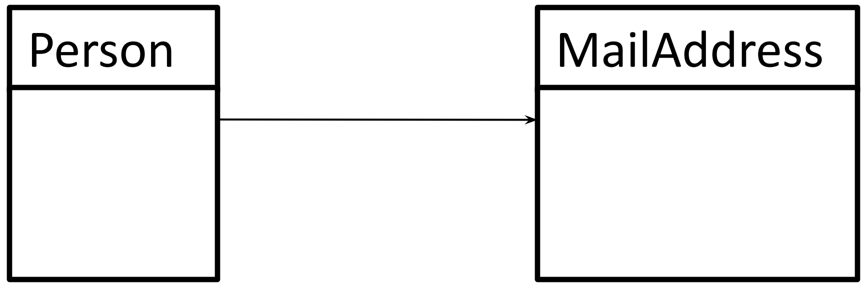
1 38055 65154 7

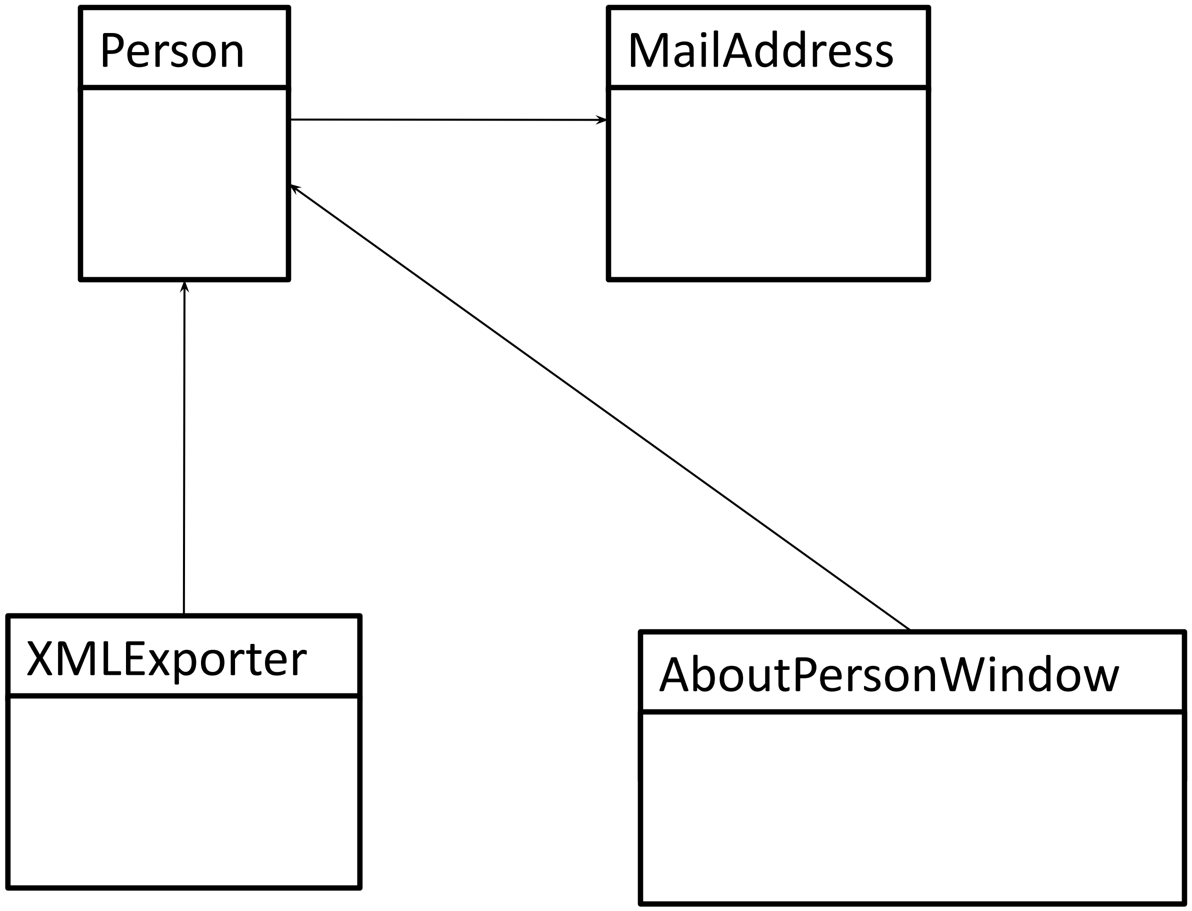
```java
double getPayAmount() {
    double result;
    if (isDead) {
        result = deadAmount();
    } else {
        if (isSeparated) {
            result = separatedAmount();
        } else {
            if (isRetired) {
                result = retiredAmount();
            } else {
                result = normalAmount();
            }
        }
    }
    return result;
}
```

Replace nested conditionals with guard clauses

```
double getPayAmount() {
      if (isDead) {
         return deadAmount();
      }
      if (isSeparated) {
         return separatedAmount();
      }
      if (isRetired) {
         return retiredAmount();
      }
      return normalAmount();
}
```

1 38055 65154 7


Person
MailAddress


Person
MailAddress
XMLExporter
AboutPersonWindow

```java
class Person {
      private String name;
      private MailAddress mailAddress;

      public MailAddress getMailAddress() {
         return mailAddress;
      }
      public void setMail(MailAddress mailAddress) {
         this.mailAddress = mailAddress;
      }
      // ..
   }
```

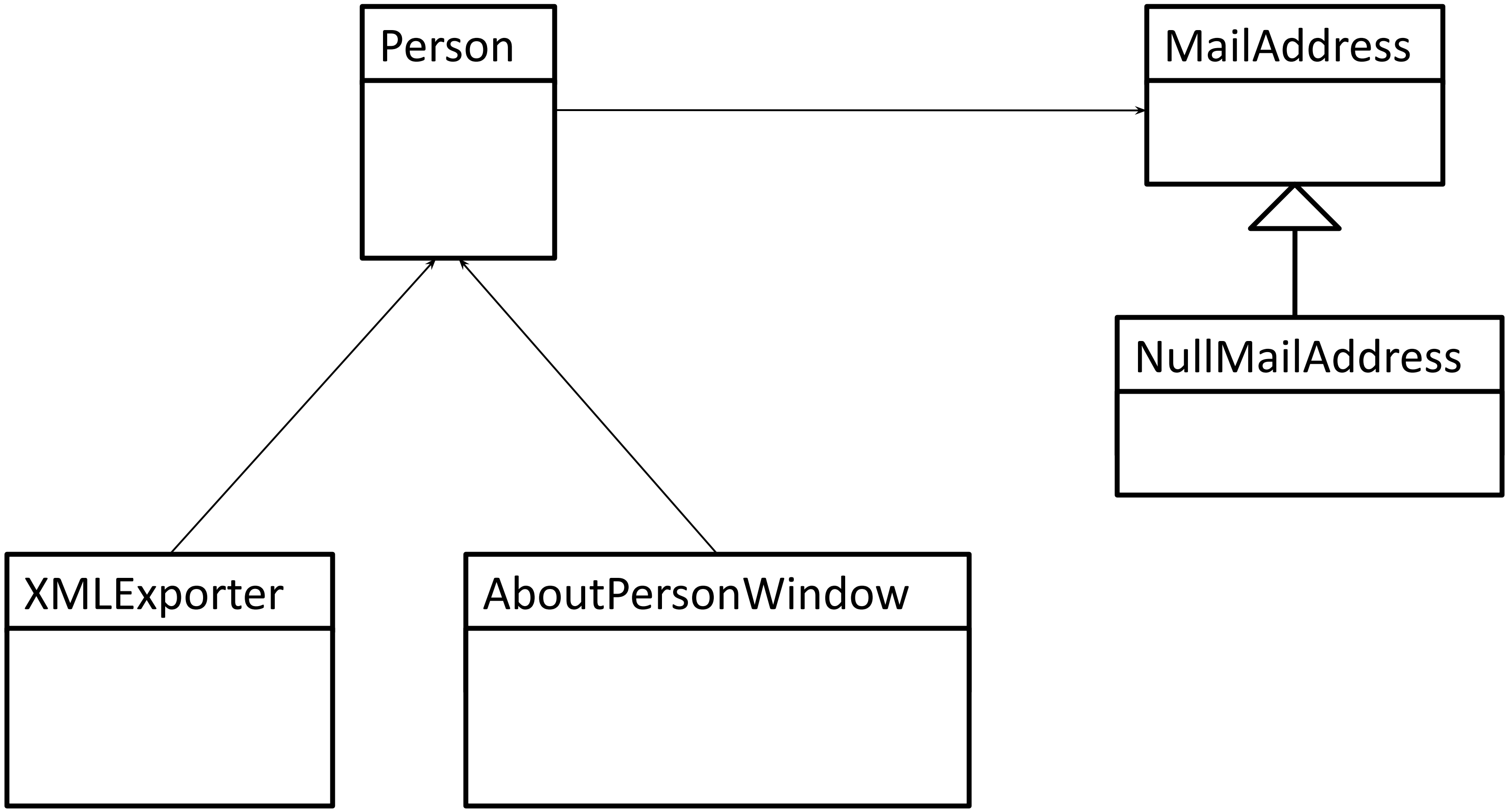
```java
class AboutPersonWindow {
      private JTextField mailAddressTextField;
      private JTextField nameTextField;

      private Person person;

      public void render() {
          // ...
          nameTextField.setText(person.name);
          if (person.getMailAddress() != null) {
             mailAddressTextField.setText(
           person.getMailAddress().toString());
          } else {
         mailAddressTextField.setText("-");
      }
          // ...
       }
}
```

```java
class AboutPersonWindow {
      private JTextField mailAddressTextField;
      private JTextField nameTextField;

      private Person person;

      public void render() {
         // ...
         nameTextField.setText(person.name);
         if (person.getMailAddress() != null) {
            mailAddressTextField.setText(
          person.getMailAddress().toString());
         } else {
        mailAddressTextField.setText("-");
     }
         // ...
      }
}
```

```java
class XMLExporter {
      private List<Person> persons;

      public void export() {
            // ...
            for (Person person : persons) {
                  print("<person>");
                  // ..
                  print("<mailAddress>");
                  if (person.getMailAddress() != null) {
                        print(person.getMailAddress().toString());
                  } else {
                        print("-");
                  }
                  print("</mailAddress>");
                  // ..
                  print("</person>");
            }
            // ...
      }
}
```

```
class XMLExporter {
      private List<Person> persons;

      public void export() {
         // ...
         for (Person person : persons) {
            print("<person>");
            // ..
            print("<mailAddress>");
            if (person.getMailAddress() != null) {
               print(person.getMailAddress().toString());
            } else {
               print("-");
            }
            print("</mailAddress>");
            // ..
            print("</person>");
         }
         // ...
      }
}
```

Introduce NullObject


Person
MailAddress
NullMailAddress
XMLExporter
AboutPersonWindow

```
class Person {
      // ..
      private MailAddress mailAddress = new NullMailAddress();
      // ..
}

class NullMailAddress extends MailAddress {

      public String toString() {
         return "-";
      }

}
```

```java
class AboutPersonWindow {

      private JTextField mailAddressTextField;
      private JTextField nameTextField;

      private Person person;

      public void render() {
         // ...
         nameTextField.setText(person.name);
         mailAddressTextField.setText(
                person.getMailAddress().toString());
         // ...
      }
}
```

```java
class XMLExporter {
      private List<Person> persons;

      public void export() {
         // ...
         for (Person person : persons) {
            print("<person>");
            // ..
            print("<mailAddress>");
            print(person.getMailAddress().toString());
            print("</mailAddress>");
            // ..
            print("</person>");
         }
         // ...
      }
```

1 38055 65154 7

```
calculateWeeklyPay(true);

calculateWeeklyPay(false);
```

```java
calculateWeeklyPay(true);

calculateWeeklyPay(false);
```

```java
public int calculateWeeklyPay(final boolean overtime) {
   int straightTime = Math.min(400, getHoursWorked());
   int straightPay = straightTime * getHoursRate();
   int overTime = Math.max(0, getHoursWorked() - straightTime);
   double overtimeRate = overtime ? 1.5 : 1.0 * getHoursRate();
   int overtimePay = (int) Math.round(overTime * overtimeRate);
   return straightPay + overtimePay;
}
```

```
public int straightPay() {
    ...
    ...
}

public int overtimePay() {
    ...
    ...
}
```

1 38055 65154 7

```java
double getPrice(int quantity, double itemPrice) {
   double basePrice = quantity * itemPrice;
   int discountLevel = getDiscountLevel();
   return discountedPrice(basePrice, discountLevel);
}


double discountedPrice(double basePrice, int discountLevel) {
   if (discountLevel == 2) {
      return basePrice * 0.1;
   }
   return basePrice * 0.05;
}
```

Replace parameter with method

```java
double getPrice(int quantity, double itemPrice) {
    double basePrice = quantity * itemPrice;
    return discountedPrice(basePrice);
}

double discountedPrice(double basePrice) {
    if (getDiscountLevel() == 2) {
        return basePrice * 0.1;
    }
    return basePrice * 0.05;
}
```

1 38055 65154 7

```java
interface ClaimsRepository {

      List<Claim> claimsReceivedIn(Date start, Date end);

      List<Claim> claimsApprovedIn(Date start, Date end);

      List<Claim> claimsRejectedIn(Date start, Date end);

}
```

Introduce parameter object

```
interface ClaimsRepository {

      List<Claim> claimsReceivedIn(Range<Date> range);

      List<Claim> claimsApprovedIn(Range<Date> range);

      List<Claim> claimsRejectedIn(Range<Date> range);

}
```

```java
class RepositorioDeClientes {

      public void agregar(long id, String doc, String cuit,
       String nombre, String apellido, String telefono,
             String mail, String direccion, String localidad,
       String piso, String provincia) {

       // Agrega un nuevo cliente a la DB
      }

      public void modificar(long id, String doc, String cuit,
       String nombre, String apellido, String telefono,
             String mail, String direccion, String localidad,
       String piso, String provincia) {

       // Agrega un nuevo cliente a la DB
      }

}
```

```
class RepositorioDeClientes {

      public void agregar(Cliente cliente) {
         // Agrega un nuevo cliente a la DB
      }

      public void modificar(Cliente cliente) {
         // modifica un cliente de la DB
      }

}
```

1 38055 65154 7

```java
int withdraw(double amount) {
    if (amount > balance) {
        return -1;
    }

    balance -= amount;
    return 0;
}


void usoEnCodigoCliente () {
    if (withdraw(200) < 0) {
        handleError();
    }
    moreCode();
}
```

Replace error code with exception

```java
void withdraw(double amount) {
   if (amount > balance) {
      throw new BalanceException(balance, amount);
   }
   balance -= amount;
}


public void usoEnCodigoCliente() {
   try {
      withdraw(200);
   } catch (BalanceException e) {
      handleError();
   }
   moreCode();
}
```

```java
public ErrorCode xxxxxxxxx(String inputFileName, Status status) {
  ErrorCode errorCode = ErrorCode.NONE;
  File file = openFile(inputFileName, status);
  if (status == Status.ERROR) {
    errorCode = ErrorCode.FileOpenError;
  } else {
    FileData data = readFile(file, status);
    if (status == Status.SUCCESS) {
       FileData summary = summarizeFileData(data, status);
       if (status == Status.ERROR) {
         errorCode = ErrorCode.DataSummaryError;
       } else {
      printSummary(summary);
         saveSummary(summary, status);
         if (status == Status.ERROR) {
        errorCode = ErrorCode.SummarySaveError;
      } else {
        updateAllAccounts();
        eraseUndoFile();
      }
       }
     } else {
        errorCode = ErrorCode.FileReadError;
     }
   }
   return errorCode;
}
```

```
public ErrorCode xxxx(inputFileName, Status status) {
   ErrorCode errorCode = ErrorCode.NONE;
   File file = openFile(inputFileName, status);
   if (status == Status.SUCCESS) {
      FileData data = readFile(file, status);
      if (status == Status.SUCCESS) {
        FileData summary = summarizeFileData(data, status);
        if (status == Status.SUCCESS) {
           printSummary(summary);
           saveSummary(summary, status);
           if (status == Status.SUCCESS) {
              updateAllAccounts();
              eraseUndoFile();
           } else {
           errorCode = ErrorCode.SummarySaveError;
           }
        } else {
         errorCode = ErrorCode.DataSummaryError;
        }
      } else {
         errorCode = ErrorCode.FileReadError;
      }
   } else {
      errorCode = ErrorCode.FileOpenError;
   }
   return errorCode;
}
```

```java
public ErrorCode xxxx(String inputFileName, Status status) {
   File file = openFile(inputFileName, status);
   if (status == Status.SUCCESS) {
      return ErrorCode.NONE;
   }
   FileData data = readFile(file, status);
   if (status == Status.ERROR) {
      return ErrorCode.FileReadError;
   }
   FileData summary = summarizeFileData(data, status);
   if (status == Status.ERROR) {
      return ErrorCode.DataSummaryError;
   }
   printSummary(summary);
   saveSummary(summary, status);
   if (status == Status.ERROR) {
      return ErrorCode.SummarySaveError;
   }
   updateAllAccounts();
   eraseUndoFile();
   return ErrorCode.NONE;
}
```

```java
void xxxx(String inputFileName) {
    File file = openFile(inputFileName);
    FileData data = readFile(file);
    FileData summary = summarizeFileData(data);
    printSummary(summary);
    saveSummary(summary);
    updateAllAccounts();
    eraseUndoFile();
}
```

1 38055 65154 7

```java
class Loan {

      private  double minAmount = 100;
      private  double maxAmount = 5000;

      private final Client client;
      private final double amountToLoan;

      public Loan(Client aClient, double amount) {
         client = aClient;
         checkAmountToLoan(amount);
         amountToLoan = amount;
      }
      private void checkAmountToLoan(double amount) {
         if (amount < minAmount || maxAmount < amount) {
            throw new RuntimeException("the loan can not be ...");
         }
      }
      // ...
}
```

```java
// Montana rusa
class RollerCoaster {

   private  int minAge = 13;
   private  int maxAge = 40;

   public void admit(Person person) {
      if (person.age < minAge || maxAge < person.age) {
         throw new RuntimeException("no esta en el rango...");
      }
   }
   // ...
}
```

Replace primitive with object

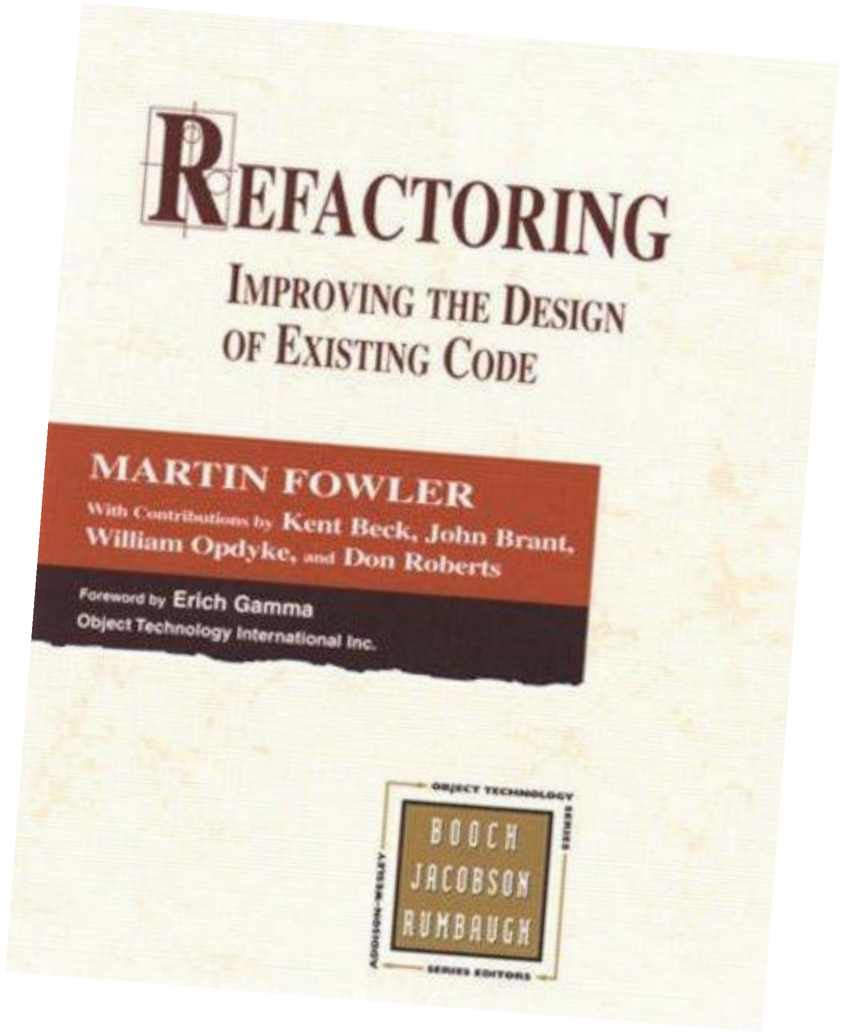
```java
class Loan {
      private Range<Double> rangeOfAllowedAmount
              = new Range<Double>(100.0, 5000.0);

      private final Client client;
      private final double amountToLoan;

      public Loan(Client aClient, double amount) {
         client = aClient;
         checkAmountToLoan(amount);
         amountToLoan = amount;
      }

      private void checkAmountToLoan(double amount) {
         if (rangeOfAllowedAmount.notIncludes(amount)) {
            throw new RuntimeException("the loan can not be ..");
         }
      }
      // ...
   }
```

```java
 class RollerCoaster {
   private  Range<Integer> rangeOfAllowedAge
                                     = new Range<Integer>(13, 40);
   public void admit(Person person) {
      if (rangeOfAllowedAge.notIncludes(person.age)) {
         throw new RuntimeException("no esta en el rango de ..");
      }
   }
   // ...
}
```

```java
public class Range<T extends Comparable<? super T>> {

    private final T start;
    private final T end;

    public Range(T start, T end) {
        this.start = start;
        this.end = end;
    }

    public boolean notIncludes(T value) {
        return !includes(value);
    }
    public boolean includes(T value) {
        return startIsLowerOrEqualThan(value)
                 && endIsGreaterOrEqualThan(value);
    }
    private boolean startIsLowerOrEqualThan(T value) {
        return start.compareTo(value) <= 0;
    }
    private boolean endIsGreaterOrEqualThan(T value) {
        return end.compareTo(value) >= 0;
    }
}
```

```
class Loan {

   private Range<Money> rangeOfAllowedAmount
      = new Range<Money>(Money.dollars(100), Money.dollars(5000));

   private final Client client;
   private final Money amountToLoan;

   public Loan(Client aClient, Money amount) {
      client = aClient;
      checkAmountToLoan(amount);
      amountToLoan = amount;
   }

   private void checkAmountToLoan(Money amount) {
      if (rangeOfAllowedAmount.notIncludes(amount)) {
         throw new RuntimeException("the loan can not be ... etc")
      }
   }
   // ...
}


class Money implements Comparable<Money> { ... }
```

Bibliografía



REFACTORING

IMPROVING THE DESIGN OF EXISTING CODE

MARTIN FOWLER

With Contributions by Kent Beck, John Brant, William Opdyke, and Don Roberts

Foreword by Erich Gamma
Object Technology International Inc.

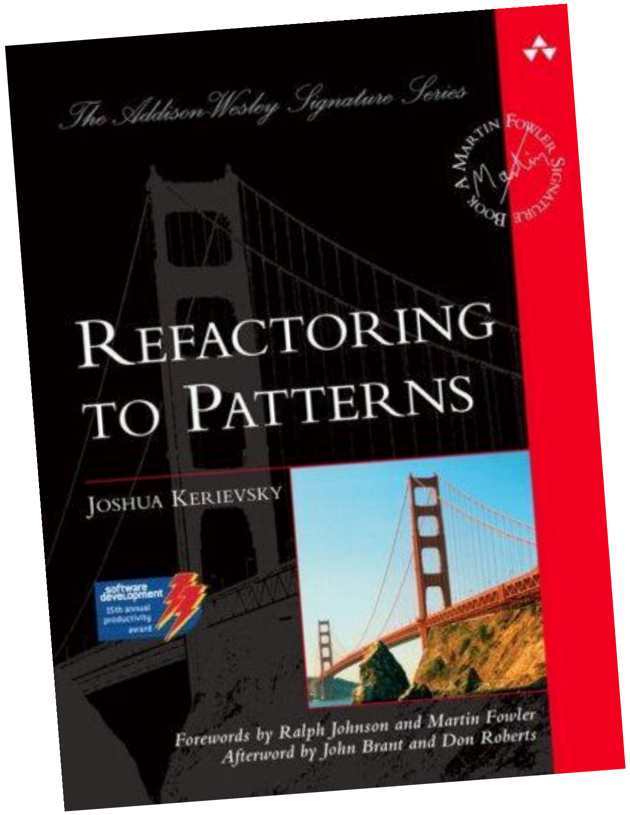
OBJECT TECHNOLOGY SERIES

BOOCH
JACOBSON
RUMBAUGH

ADDISON-WESLEY

SERIES EDITORS

Lectura Adicional


Refactoring to Patterns

Joshua Kerievsky

Forewords by Ralph Johnson and Martin Fowler

Afterword by John Brant and Don Roberts

The Addison-Wesley Signature Series
A Martin Fowler Signature Book
software development 15th annual productivity award

?