# *Assignment 2 -Refactoring-*

Group 27a: Ata Ağırkaya, Alex Bolfă, Lauren de Hoop, Ioan Hedea, Paris Loizides, Rok Štular

## Introduction

For this assignment we have used the MetricsTree plugin to calculate our initial metrics and monitor our refactoring results. Our initial strategy was to look through the different class metrics on a project level and assess which of those have the most extreme values. Looking more closely at what classes caused the extreme values we could then focus on those that were present in more than one of the metric values and see if refactoring can be applied. Looking further at the classes with high/extreme values, we would then examine them closely to find the presence of complex methods. Additionally, as we have written our code quite recently, some methods that might need refactoring came to mind. Eventually we have come decided to refactor the following classes and methods:

<u>Classes:</u>

- Contract - the contract database entity class in the contract microservice
- ContractDto - the DTO class part of the commons package of the contract microservice
- Request - the request entity in the request microservice
- UserModify - the UserModify DTO in the user commons package
- SalaryScale - the SalaryScale database entity in the contract microservice

<u>Methods:</u>

- RequestService: approveRequest - approves open requests
- ContractService: modiftyDraftContract - makes modifications to an existing contract
- User: updateUser - updates the user information
- BaseEntity: equals - equals method for all entities
- JwtRequestFilter: doFilterInternal - verifies and authenticates JWT token

**Refactoring Metrics and Thresholds**

See charts in Appendix A for class metrics, on the project level.

***For the Class refactoring our focus was on the following metrics:***

**Number of Attributes -** Contract: 15, ContractDto: 25, Request: 11, UserModify: 8, SalaryScale: 5.

- We intend to get all values below 9, and decrease the already lower values by a realistic margin, which is class dependent.

**Weighted Methods per Class -** Contract: 82, ContractDto: 81, Request: 64, UserModify: 50, SalaryScale: 34.

- For the higher values, we intend to get them below 35, with the intention to halve most . As for the Salary Scale, we will reduce it to below 20.

**Number of Methods -** Contract: 40, ContractDto: 37, Request: 30, UserModify: 23, SalaryScale: 17.

- The number of methods are also related to the number of attributes as the getter and setter methods of these attributes contribute to the number. All values should be below 25, for the UserModify the intent threshold should be below 15, and for the salary scale below 15 as well.

***As for the Method refactoring:***

**McCabe Cyclomatic Complexity:** userUpdate: 9 , approveRequest: 7 , doInternalFilter: 7 , modifyDraftContract: 10, equals: 5

- A common value for the cyclomatic complexity is below 3, thus our goal is to reduce the MCC to 3 or lower.

**Lines of Code:** updateUser: 60, approveRequest: 37 , doInternalFilter: 44, modifyDraftContract: 39, equals: 11

- The updateUser method has a very high LOC, the goal for this method will be to get the LOC to below 30. For the other methods, below 15 will be our target threshold.

**Number of Parameters:** updateUser: 1, approveRequest: 1, doInternalFilter: 3, modifyDraftContract: 2, equals: 1

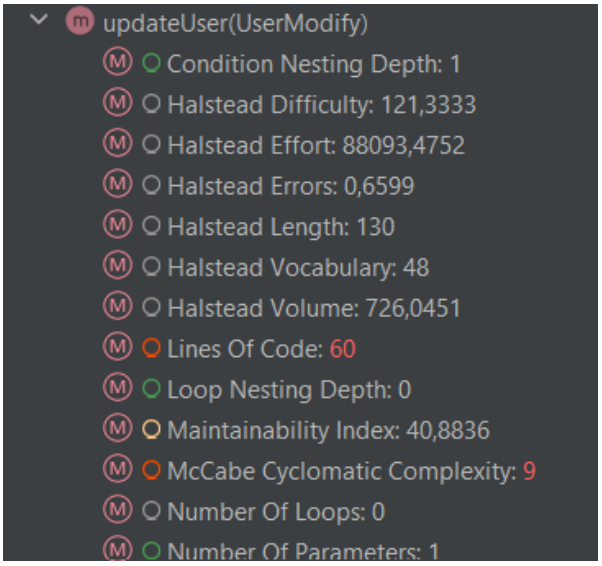
- The number of parameters is only relevant for the doInternalFilter method, which should be below 3. The other methods already have an acceptable amount of parameters.

# Overview of changes:

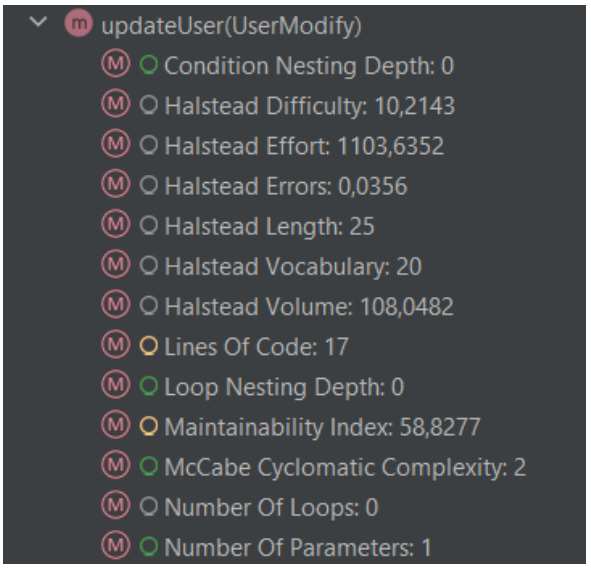
## userUpdate method in UserService class:

Refactoring of the updateUser method leads to a decent reduction of the LOC metric, as well as the CC metric of the method. The refactoring includes the extraction of code responsible for creating the updated User entity that will be saved into the database replacing the previous one into a new one called createUpdatedUser. The createUpdatedUser metrics can also be seen in the following screenshots. In this way we split the logic of the updateUser method in two different methods thus making it more readable, maintainable and reduce the Lines of Code.

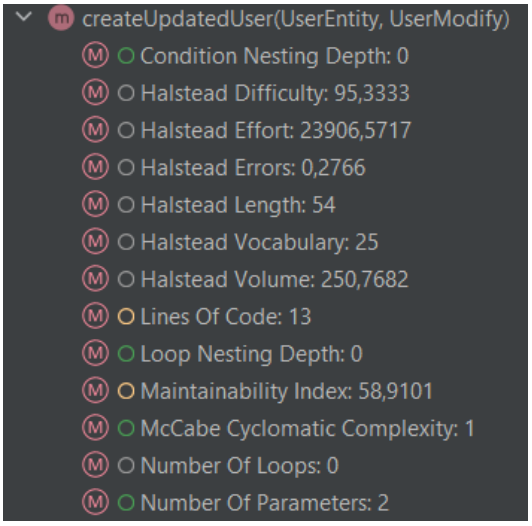
**Before:**




**After:**




**Extra method introduced:**



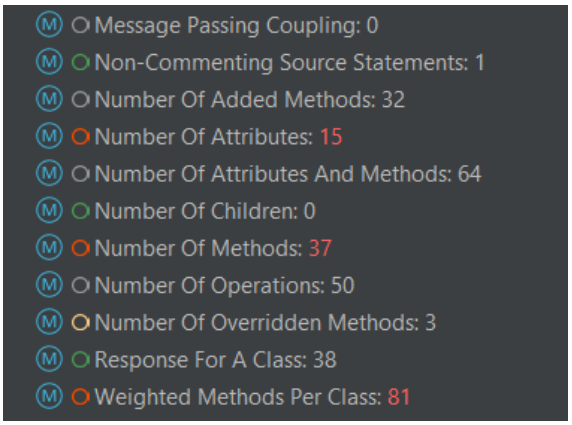
## ContractDto:

For the refactoring of this class we decided to improve the metrics by reducing the number of attributes as well as remove some unnecessary methods. Therefore, the solution was to extract the following attributes and move them into their own class:

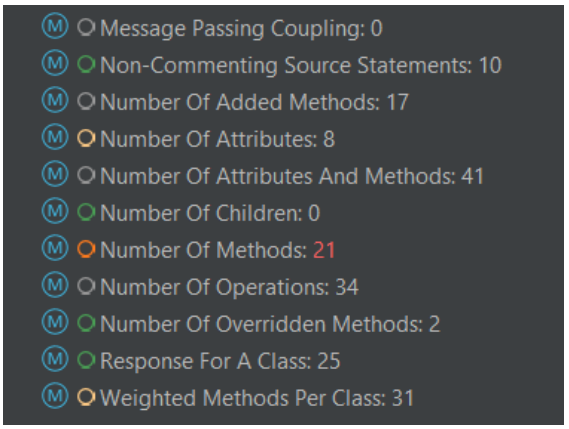
- Extracted into ***ContractTerms*** embedded class.
    - *hoursPerWeek*
    - *vacationDays*
    - *startDate*
    - *endDate*
    - *terminationDate*
    - *salaryScalePoint*
    - *lastSalaryIncreaseDate*
- Extracted into ***ContractInfo*** embedded class:
    - *type*
    - *status*
- Extracted into ***ContractParties*** embedded class:
    - *employeeId*
    - *employerId*

Lastly to reduce the number of methods present in classes as well as the weighted methods per class we removed all lombok annotations that we didn't really need. This avoided introducing unnecessary methods, and we implemented only the methods that we really needed in code like getters and setters.

**Before:**




**After:**



# Contract class:

As with the *ContractDto*, several fields were extracted from the *Contract* class and moved into their own respective subclasses. These fields were:

- Extracted into ***ContractTerms*** embedded class.
    - *hoursPerWeek*
    - *vacationDays*
    - *startDate*
    - *endDate*
    - *terminationDate*
    - *salaryScalePoint*
    - *lastSalaryIncreaseDate*
- Extracted into ***ContractInfo*** embedded class:
    - *type*
    - *status*
- Extracted into ***ContractParties*** embedded class:
    - *employeeId*
    - *employerId*

As a result, the several metrics, mainly the ones relating to the number of attributes present dropped drastically. Additionally, by coupling related fields into nested classes, we increased the overall readability of the code, as the class is not as long as it has been before, and requires less tedious navigation in order to determine which field is where and what its purpose is.

Here are the screenshots of the metrics screen before and after the refactoring of the *Contract* class.

**Before refactoring:**

Class: Contract

| Metric | Metrics Set | Description | Value | Regular Range |
|---|---|---|---|---|
| CHVL | Halstead Metric Set | Halstead Volume | 552.3562 | |
| CHD | Halstead Metric Set | Halstead Difficulty | 31.2 | |
| CHL | Halstead Metric Set | Halstead Length | 100 | |
| CHEF | Halstead Metric Set | Halstead Effort | 17233.5133 | |
| CHVC | Halstead Metric Set | Halstead Vocabulary | 46 | |
| CHER | Halstead Metric Set | Halstead Errors | 0.2224 | |
| WMC | Chidamber-Kemerer Metrics Set | Weighted Methods Per Class | 37 | [0..12) |
| DIT | Chidamber-Kemerer Metrics Set | Depth Of Inheritance Tree | 2 | [0..3) |
| RFC | Chidamber-Kemerer Metrics Set | Response For A Class | 52 | [0..45) |
| LCOM | Chidamber-Kemerer Metrics Set | Lack Of Cohesion Of Methods | 16 | |
| NOC | Chidamber-Kemerer Metrics Set | Number Of Children | 0 | [0..2) |
| NOA | Lorenz-Kidd Metrics Set | Number Of Attributes | 15 | [0..4) |
| NOO | Lorenz-Kidd Metrics Set | Number Of Operations | 58 | |
| NOOM | Lorenz-Kidd Metrics Set | Number Of Overridden Methods | 1 | [0..3) |
| NOAM | Lorenz-Kidd Metrics Set | Number Of Added Methods | 31 | |
| SIZE2 | Li-Henry Metrics Set | Number Of Attributes And Methods | 73 | |
| NOM | Li-Henry Metrics Set | Number Of Methods | 37 | [0..7) |
| MPC | Li-Henry Metrics Set | Message Passing Coupling | 16 | |
| DAC | Li-Henry Metrics Set | Data Abstraction Coupling | 8 | |
| NCSS | Chr. Clemens Lee Metrics Set | Non-Commenting Source Statements | 4 | [0..1000) |
| CMI | Maintainability Index | Maintainability Index | 36.593 | [0.0..19.0] |

**After refactoring:**

Class: Contract

| Metric | Metrics ... | Description | Value | Regular ... |
|---|---|---|---|---|
| CHVL | Halstead... | Halstead Volume | 312.7524 | |
| CHD | Halstead... | Halstead Difficulty | 13.2 | |
| CHL | Halstead... | Halstead Length | 62 | |
| CHEF | Halstead... | Halstead Effort | 4128.3321 | |
| CHVC | Halstead... | Halstead Vocabulary | 33 | |
| CHER | Halstead... | Halstead Errors | 0.0858 | |
| WMC | Chidamb... | Weighted Methods Per Class | 19 | [0..12) |
| DIT | Chidamb... | Depth Of Inheritance Tree | 2 | [0..3) |
| RFC | Chidamb... | Response For A Class | 31 | [0..45) |
| LCOM | Chidamb... | Lack Of Cohesion Of Methods | 8 | |
| NOC | Chidamb... | Number Of Children | 0 | [0..2) |
| NOA | Lorenz-K... | Number Of Attributes | 7 | [0..4) |
| NOO | Lorenz-K... | Number Of Operations | 40 | |
| NOOM | Lorenz-K... | Number Of Overridden Methods | 0 | [0..3) |
| NOAM | Lorenz-K... | Number Of Added Methods | 15 | |
| SIZE2 | Li-Henry ... | Number Of Attributes And Methods | 47 | |
| NOM | Li-Henry ... | Number Of Methods | 19 | [0..7) |
| MPC | Li-Henry ... | Message Passing Coupling | 10 | |
| DAC | Li-Henry ... | Data Abstraction Coupling | 6 | |

## approveRequest() inside of RequestService:

We observed that this method was violating the Open-Closed Principle by having nested if-else statements to choose what should be done when a request is approved, depending on the request type. Since the types of request might grow in the future this would mean a modification of the structure. To avoid this whilst also allowing for expansion we refactored using Replace Conditional With Polymorphism by creating a different class with its own logic for each request type. All classes inherit from a bigger general request type class which holds all common methods and attributes.

This refactoring improved various metrics such as cutting LOC (Lines of Code) by 24 and getting the McCabe Cyclomatic Complexity from 7 to 1.

**Metrics:**

*Before*

approveRequest(Request)
- Condition Nesting Depth: 2
- Halstead Difficulty: 32,8235
- Halstead Effort: 16315,3181
- Halstead Errors: 0,2144
- Halstead Length: 89
- Halstead Vocabulary: 48
- Halstead Volume: 497,0617
- Lines Of Code: 37
- Loop Nesting Depth: 0
- Maintainability Index: 46,6497
- McCabe Cyclomatic Complexity: 7
- Number Of Loops: 0
- Number Of Parameters: 1

*After*

approveRequest(GeneralRequest)
- Condition Nesting Depth: 0
- Halstead Difficulty: 8,0
- Halstead Effort: 458,8752
- Halstead Errors: 0,0198
- Halstead Length: 16
- Halstead Vocabulary: 12
- Halstead Volume: 57,3594
- Lines Of Code: 13
- Loop Nesting Depth: 0
- Maintainability Index: 63,4058
- McCabe Cyclomatic Complexity: 1
- Number Of Loops: 0
- Number Of Parameters: 1

# UserModify Class before:

The method that solved the weighted methods per class was to delete the lombok @Data annotation and just implement the getters and setters manually. This is because we efficiently implement just what is needed instead of just generating all the methods.

**Before:**

- Data Abstraction Coupling: 2
- Depth Of Inheritance Tree: 1
- Halstead Difficulty: 5.5909
- Halstead Effort: 3647.9027
- Halstead Errors: 0.079
- Halstead Length: 121
- Halstead Vocabulary: 42
- Halstead Volume: 652.4704
- Lack Of Cohesion Of Methods: 10
- Maintainability Index: 38.1368
- Message Passing Coupling: 0
- Non-Commenting Source Statements: 1
- Number Of Added Methods: 18
- Number Of Attributes: 8
- Number Of Attributes And Methods: 43
- Number Of Children: 0
- Number Of Methods: 23
- Number Of Operations: 36
- Number Of Overridden Methods: 3
- Response For A Class: 24
- Weighted Methods Per Class: 50

**After:**

- Data Abstraction Coupling: 2
- Depth Of Inheritance Tree: 1
- Halstead Difficulty: 21.1452
- Halstead Effort: 41691.7739
- Halstead Errors: 0.4008
- Halstead Length: 311
- Halstead Vocabulary: 81
- Halstead Volume: 1971.6934
- Lack Of Cohesion Of Methods: 9
- Maintainability Index: 34.606
- Message Passing Coupling: 9
- Non-Commenting Source Statements: 26
- Number Of Added Methods: 17
- Number Of Attributes: 8
- Number Of Attributes And Methods: 41
- Number Of Children: 0
- Number Of Methods: 21
- Number Of Operations: 34
- Number Of Overridden Methods: 2
- Response For A Class: 25
- Weighted Methods Per Class: 31

# DoInternalFileter method in the JwtRequestFilter class:

The lines of code were high so I delegated the additional functionality of the authorization header and parsing the directions in a helper class. I split the number of parameters in the same way by splitting the functionality into multiple classes(the original one plus the helper.). The cyclotomic complexity was of order 7 (the upper strict bound should have been 3), which was reduced with the help of the helper method by modifying the parameters in place. Also, we have set flags for reducing this CC by just checking the flag without going through nested if statements.

**Before:**

doFilterInternal(HttpServletRequest, HttpServle
- Condition Nesting Depth: 3
- Halstead Difficulty: 17.5238
- Halstead Effort: 5835.8724
- Halstead Errors: 0.108
- Halstead Length: 61
- Halstead Vocabulary: 44
- Halstead Volume: 333.0253
- Lines Of Code: 44
- Loop Nesting Depth: 0
- Maintainability Index: 46.2259
- McCabe Cyclomatic Complexity: 7
- Number Of Loops: 0
- Number Of Parameters: 3

**After:**

doFilterInternal(HttpServletRequest, HttpServletResponse, FilterChain)
- Condition Nesting Depth: 1
- Halstead Difficulty: 14.4
- Halstead Effort: 2436.7079
- Halstead Errors: 0.0604
- Halstead Length: 36
- Halstead Vocabulary: 26
- Halstead Volume: 169.2158
- Lines Of Code: 27
- Loop Nesting Depth: 0
- Maintainability Index: 53.0288
- McCabe Cyclomatic Complexity: 3
- Number Of Loops: 0
- Number Of Parameters: 3

doFilterInternalHelper(HttpServletRequest, HttpServletResponse, FilterChain)
- Condition Nesting Depth: 1

# Separation of RequestBuilder class:

When refactoring the code, I considered various code metrics such as readability, maintainability, and scalability. I found that having a class within a class (nested class) did not meet my desired standards in terms of these metrics. Specifically, I separated the class RequestBuilder from inside the Request class. This decision not only improved the code readability by making it easier to understand the relationship between classes and their responsibilities, but also decreased the number of Weighted Methods per Class. Additionally, it made the code more maintainable by allowing for easier modification and debugging of the individual classes. Furthermore, this separation may also improve scalability by allowing the classes to grow and evolve independently of each other. Overall, by refactoring the nested class RequestBuilder into a standalone class, I was able to improve the overall quality and structure of the codebase.

Before:

Class: Request

| Metric | Metrics Set | Description | Value |
|---|---|---|---|
| CHVL | Halstead M... | Halstead Volume | 1295,6843 |
| CHD | Halstead M... | Halstead Difficulty | 19,7826 |
| CHL | Halstead M... | Halstead Length | 210 |
| CHEF | Halstead M... | Halstead Effort | 25632,0145 |
| CHVC | Halstead M... | Halstead Vocabulary | 72 |
| CHER | Halstead M... | Halstead Errors | 0,2898 |
| WMC | Chidamber... | Weighted Methods Per Class | 29 |
| DIT | Chidamber... | Depth Of Inheritance Tree | 2 |
| RFC | Chidamber... | Response For A Class | 37 |
| LCOM | Chidamber... | Lack Of Cohesion Of Methods | 12 |
| NOC | Chidamber... | Number Of Children | 0 |
| NOA | Lorenz-Kid... | Number Of Attributes | 11 |
| NOO | Lorenz-Kid... | Number Of Operations | 48 |
| NOOM | Lorenz-Kid... | Number Of Overridden Methods | 1 |
| NOAM | Lorenz-Kid... | Number Of Added Methods | 22 |
| SIZE2 | Li-Henry M... | Number Of Attributes And Methods | 59 |
| NOM | Li-Henry M... | Number Of Methods | 27 |
| MPC | Li-Henry M... | Message Passing Coupling | 0 |
| DAC | Li-Henry M... | Data Abstraction Coupling | 5 |
| NCSS | Chr. Cleme... | Non-Commenting Source Statements | 21 |
| CMI | Maintainab... | Maintainability Index | 37,7807 |

After:

Class: Request

| Metric | Metrics Set | Description | Value |
|---|---|---|---|
| CHVL | Halstead M... | Halstead Volume | 851,7234 |
| CHD | Halstead M... | Halstead Difficulty | 42,4286 |
| CHL | Halstead M... | Halstead Length | 148 |
| CHEF | Halstead M... | Halstead Effort | 36137,405 |
| CHVC | Halstead M... | Halstead Vocabulary | 54 |
| CHER | Halstead M... | Halstead Errors | 0,3643 |
| WMC | Chidamber... | Weighted Methods Per Class | 53 |
| DIT | Chidamber... | Depth Of Inheritance Tree | 2 |
| RFC | Chidamber... | Response For A Class | 39 |
| LCOM | Chidamber... | Lack Of Cohesion Of Methods | 10 |
| NOC | Chidamber... | Number Of Children | 0 |
| NOA | Lorenz-Kid... | Number Of Attributes | 11 |
| NOO | Lorenz-Kid... | Number Of Operations | 40 |
| NOOM | Lorenz-Kid... | Number Of Overridden Methods | 3 |
| NOAM | Lorenz-Kid... | Number Of Added Methods | 12 |
| SIZE2 | Li-Henry M... | Number Of Attributes And Methods | 51 |
| NOM | Li-Henry M... | Number Of Methods | 19 |
| MPC | Li-Henry M... | Message Passing Coupling | 20 |
| DAC | Li-Henry M... | Data Abstraction Coupling | 5 |
| NCSS | Chr. Cleme... | Non-Commenting Source Statements | 30 |
| CMI | Maintainab... | Maintainability Index | 32,55 |

# modifyDraftContract in ContractService class:

The initial code has high values for McCabe Cyclomatic Complexity and the lines of code metrics. Examining the method will tell you that it is too large, there are too many if statements with a lack of overall cohesion in the method. It is also too reliant on methods from the Contract entity class, directly calling the setter methods within the class.

This method takes as input several variables that are allowed to be changed if the contract is still in the draft stage. There is one method for all modifications, thus the method relies on individual variable validation to ensure only the non null values are modified. First of all to reduce the high coupling in the class, the method is moved to the contract entity class. The modify method in the ContractService class will now serve as a delegate to the implementation in the Contract class. Secondly the code is too small and the Cyclomatic Complexity is too large. To delegate this complexity instead of validating the variables in the modify method, this functionality is moved to the set method for each individual variable. The modify method now only has to call the setter methods. The complexity is further reduced by the class refactoring of the contract class, as fields have been extracted into different classes, and thus the modify method is implemented in all these classes separately.

As can be seen in the relevant metric tables, the CC and LOC have been reduced significantly, as well as the Halstead Metric Set.

Method: modifyDraftContract(Contract, ContractModificationDto)

| Metric | Metrics Set | Description | Value | Regular Ra... |
|---|---|---|---|---|
| CND | | Condition Nesting Depth | 1 | [0..2) |
| LND | | Loop Nesting Depth | 0 | [0..2) |
| CC | | McCabe Cyclomatic Complexity | 10 | [0..3) |
| NOL | | Number Of Loops | 0 | |
| LOC | | Lines Of Code | 39 | [0..11) |
| NOPM | | Number Of Parameters | 2 | [0..3) |
| HVL | Halstead M... | Halstead Volume | 448.9511 | |
| HD | Halstead M... | Halstead Difficulty | 62.5 | |
| HL | Halstead M... | Halstead Length | 89 | |
| HEF | Halstead M... | Halstead Effort | 28059.4423 | |
| HVC | Halstead M... | Halstead Vocabulary | 33 | |
| HER | Halstead M... | Halstead Errors | 0.3078 | |
| MMI | Maintainabi... | Maintainability Index | 46.4186 | [0.0..19.0] |

*Before:*

Method: modifyDraftContract(Contract, ContractModificationDto)

| Metric | Metrics Set | Description | Value | Regular Range |
|---|---|---|---|---|
| CND | | Condition Nesting Depth | 1 | [0..2) |
| LND | | Loop Nesting Depth | 0 | [0..2) |
| CC | | McCabe Cyclomatic Complexity | 3 | [0..3) |
| NOL | | Number Of Loops | 0 | |
| LOC | | Lines Of Code | 22 | [0..11) |
| NOPM | | Number Of Parameters | 2 | [0..3) |
| HVL | Halstead Metric Set | Halstead Volume | 138.2424 | |
| HD | Halstead Metric Set | Halstead Difficulty | 13.125 | |
| HL | Halstead Metric Set | Halstead Length | 31 | |
| HEF | Halstead Metric Set | Halstead Effort | 1814.4312 | |
| HVC | Halstead Metric Set | Halstead Vocabulary | 22 | |
| HER | Halstead Metric Set | Halstead Errors | 0.0496 | |
| MMI | Maintainability Index | Maintainability Index | 55.5852 | [0.0..19.0] |

*After:*

Method: modifyDraft(ContractModificationDto, JobPosition)

| Metric | Metrics Set | Description | Value | Regular Range |
|---|---|---|---|---|
| CND | | Condition Nesting Depth | 1 | [0..2) |
| LND | | Loop Nesting Depth | 0 | [0..2) |
| CC | | McCabe Cyclomatic Complexity | 2 | [0..3) |
| NOL | | Number Of Loops | 0 | |
| LOC | | Lines Of Code | 18 | [0..11) |
| NOPM | | Number Of Parameters | 2 | [0..3) |
| HVL | Halstead Metric Set | Halstead Volume | 89.2065 | |
| HD | Halstead Metric Set | Halstead Difficulty | 9.8 | |
| HL | Halstead Metric Set | Halstead Length | 21 | |
| HEF | Halstead Metric Set | Halstead Effort | 874.2235 | |
| HVC | Halstead Metric Set | Halstead Vocabulary | 19 | |
| HER | Halstead Metric Set | Halstead Errors | 0.0305 | |
| MMI | Maintainability Index | Maintainability Index | 58.8746 | [0.0..19.0] |

Method: setSalaryScalePoint(BigDecimal)

| Metric | Metrics Set | Description | Value | Regular Range |
|---|---|---|---|---|
| CND | | Condition Nesting Depth | 1 | [0..2) |
| LND | | Loop Nesting Depth | 0 | [0..2) |
| CC | | McCabe Cyclomatic Complexity | 4 | [0..3) |
| NOL | | Number Of Loops | 0 | |
| LOC | | Lines Of Code | 11 | [0..11) |
| NOPM | | Number Of Parameters | 1 | [0..3) |
| HVL | Halstead Metric Set | Halstead Volume | 72.3397 | |
| HD | Halstead Metric Set | Halstead Difficulty | 6.6667 | |
| HL | Halstead Metric Set | Halstead Length | 19 | |
| HEF | Halstead Metric Set | Halstead Effort | 482.265 | |
| HVC | Halstead Metric Set | Halstead Vocabulary | 14 | |
| HER | Halstead Metric Set | Halstead Errors | 0.0205 | |
| MMI | Maintainability Index | Maintainability Index | 64.0916 | [0.0..19.0] |

*Additional methods:*

Method: modify(ContractModificationDto)

| Metric | Metrics Set | Description | Value | Regular Range |
|---|---|---|---|---|
| CND | | Condition Nesting Depth | 1 | [0..2) |
| LND | | Loop Nesting Depth | 0 | [0..2) |
| CC | | McCabe Cyclomatic Complexity | 3 | [0..3) |
| NOL | | Number Of Loops | 0 | |
| LOC | | Lines Of Code | 16 | [0..11) |
| NOPM | | Number Of Parameters | 1 | [0..3) |
| HVL | Halstead Metric Set | Halstead Volume | 108.0 | |
| HD | Halstead Metric Set | Halstead Difficulty | 31.5 | |
| HL | Halstead Metric Set | Halstead Length | 27 | |
| HEF | Halstead Metric Set | Halstead Effort | 3402.0 | |
| HVC | Halstead Metric Set | Halstead Vocabulary | 16 | |
| HER | Halstead Metric Set | Halstead Errors | 0.0754 | |
| MMI | Maintainability Index | Maintainability Index | 59.3475 | [0.0..19.0] |

# Refactoring of the equals method of the BaseEntity class:

The equals method of the BaseEntity class is the method used by all JPA entities to determine equality. The method was convoluted, so the refactoring focused on improving the readability of the method and combining some of the unnecessarily separated if conditions. The result is an equals method which is approximately 40% shorter than the original method.

Method: equals(Object)

| Metric | Metrics Set | Description | Value | Regular ... |
|---|---|---|---|---|
| CND | | Condition Nesting Depth | 1 | [0..2) |
| LND | | Loop Nesting Depth | 0 | [0..2) |
| CC | | McCabe Cyclomatic Complexity | 5 | [0..3) |
| NOL | | Number Of Loops | 0 | |
| LOC | | Lines Of Code | 11 | [0..11) |
| NOPM | | Number Of Parameters | 1 | [0..3) |
| HVL | Halstead Metric Set | Halstead Volume | 112.0 | |
| HD | Halstead Metric Set | Halstead Difficulty | 7.7143 | |
| HL | Halstead Metric Set | Halstead Length | 28 | |
| HEF | Halstead Metric Set | Halstead Effort | 864.0 | |
| HVC | Halstead Metric Set | Halstead Vocabulary | 16 | |
| HER | Halstead Metric Set | Halstead Errors | 0.0302 | |
| MMI | Maintainability Index | Maintainability Index | 62.718 | [0.0..19.0] |

Method: equals(Object)

| Metric | Metrics Set | Description | Value | Regular Range |
|---|---|---|---|---|
| CND | | Condition Nesting Depth | 1 | [0..2) |
| LND | | Loop Nesting Depth | 0 | [0..2) |
| CC | | McCabe Cyclomatic Complexity | 5 | [0..3) |
| NOL | | Number Of Loops | 0 | |
| LOC | | Lines Of Code | 7 | [0..11) |
| NOPM | | Number Of Parameters | 1 | [0..3) |
| HVL | Halstead Metric Set | Halstead Volume | 97.6723 | |
| HD | Halstead Metric Set | Halstead Difficulty | 10.0 | |
| HL | Halstead Metric Set | Halstead Length | 25 | |
| HEF | Halstead Metric Set | Halstead Effort | 976.7226 | |
| HVC | Halstead Metric Set | Halstead Vocabulary | 15 | |
| HER | Halstead Metric Set | Halstead Errors | 0.0328 | |
| MMI | Maintainability Index | Maintainability Index | 67.4372 | [0.0..19.0] |

# SalaryScale:

For this class the refactoring metrics looked at are the WMC, NOA and NOM. The NOM especially since this is over the casual threshold of 14.

To reduce the code smell, class extraction is implemented. This will reduce the number of methods in the class and also the number of attributes. Two related fields have been moved to a separate class, and replaced by an object of the new class. This reduced the attributes by only one, but as there were already not many attributes, this would be considered a significant reduction in attributes. As the program scales, the new class, Pay, may start to contain more attributes that would have previously been placed in the SalaryScale class. Additionally, the equals method of the SalaryScale class is very complex, but only has to be reliant on the ID comparison. The complexity of the equals method in the SalaryScale class has been reduced significantly by implementing a call to the super equals method in the base entity that relies on ID comparison of two objects.

The refactoring of this method has moved the NOM from an extreme measure to high, and halved the WMC.

***Before:***

Class: SalaryScale

| Metric | Metrics Set | Description | Value | Regular Ra... |
|---|---|---|---|---|
| CHVL | Halstead M... | Halstead Volume | 208.8932 | |
| CHD | Halstead M... | Halstead Difficulty | 9.6429 | |
| CHL | Halstead M... | Halstead Length | 43 | |
| CHEF | Halstead M... | Halstead Effort | 2014.3271 | |
| CHVC | Halstead M... | Halstead Vocabulary | 29 | |
| CHER | Halstead M... | Halstead Errors | 0.0532 | |
| WMC | Chidamber-... | Weighted Methods Per Class | 34 | [0..12) |
| DIT | Chidamber-... | Depth Of Inheritance Tree | 2 | [0..3) |
| RFC | Chidamber-... | Response For A Class | 24 | [0..45) |
| LCOM | Chidamber-... | Lack Of Cohesion Of Methods | 7 | |
| NOC | Chidamber-... | Number Of Children | 0 | [0..2) |
| NOA | Lorenz-Kid... | Number Of Attributes | 5 | [0..4) |
| NOO | Lorenz-Kid... | Number Of Operations | 38 | |
| NOOM | Lorenz-Kid... | Number Of Overridden Methods | 3 | [0..3) |
| NOAM | Lorenz-Kid... | Number Of Added Methods | 10 | |
| SIZE2 | Li-Henry M... | Number Of Attributes And Methods | 43 | |
| NOM | Li-Henry M... | Number Of Methods | 17 | [0..7) |
| MPC | Li-Henry M... | Message Passing Coupling | 5 | |
| DAC | Li-Henry M... | Data Abstraction Coupling | 2 | |
| NCSS | Chr. Clemen... | Non-Commenting Source Statement.. | 7 | [0..1000) |
| CMI | Maintainabi... | Maintainability Index | 43.6031 | [0.0..19.0] |

***After:***

Class: SalaryScale

| Metric | Metrics Set | Description | Value | Regular Ra... |
|---|---|---|---|---|
| CHVL | Halstead M... | Halstead Volume | 284.3459 | |
| CHD | Halstead M... | Halstead Difficulty | 15.9231 | |
| CHL | Halstead M... | Halstead Length | 55 | |
| CHEF | Halstead M... | Halstead Effort | 4527.6612 | |
| CHVC | Halstead M... | Halstead Vocabulary | 36 | |
| CHER | Halstead M... | Halstead Errors | 0.0912 | |
| WMC | Chidamber-... | Weighted Methods Per Class | 14 | [0..12) |
| DIT | Chidamber-... | Depth Of Inheritance Tree | 2 | [0..3) |
| RFC | Chidamber-... | Response For A Class | 26 | [0..45) |
| LCOM | Chidamber-... | Lack Of Cohesion Of Methods | 5 | |
| NOC | Chidamber-... | Number Of Children | 0 | [0..2) |
| NOA | Lorenz-Kid... | Number Of Attributes | 4 | [0..4) |
| NOO | Lorenz-Kid... | Number Of Operations | 34 | |
| NOOM | Lorenz-Kid... | Number Of Overridden Methods | 3 | [0..3) |
| NOAM | Lorenz-Kid... | Number Of Added Methods | 7 | |
| SIZE2 | Li-Henry M... | Number Of Attributes And Methods | 38 | |
| NOM | Li-Henry M... | Number Of Methods | 13 | [0..7) |
| MPC | Li-Henry M... | Message Passing Coupling | 10 | |
| DAC | Li-Henry M... | Data Abstraction Coupling | 3 | |
| NCSS | Chr. Clemen... | Non-Commenting Source Statement.. | 8 | [0..1000) |
| CMI | Maintainabi... | Maintainability Index | 48.5177 | [0.0..19.0] |

***Additional methods:***

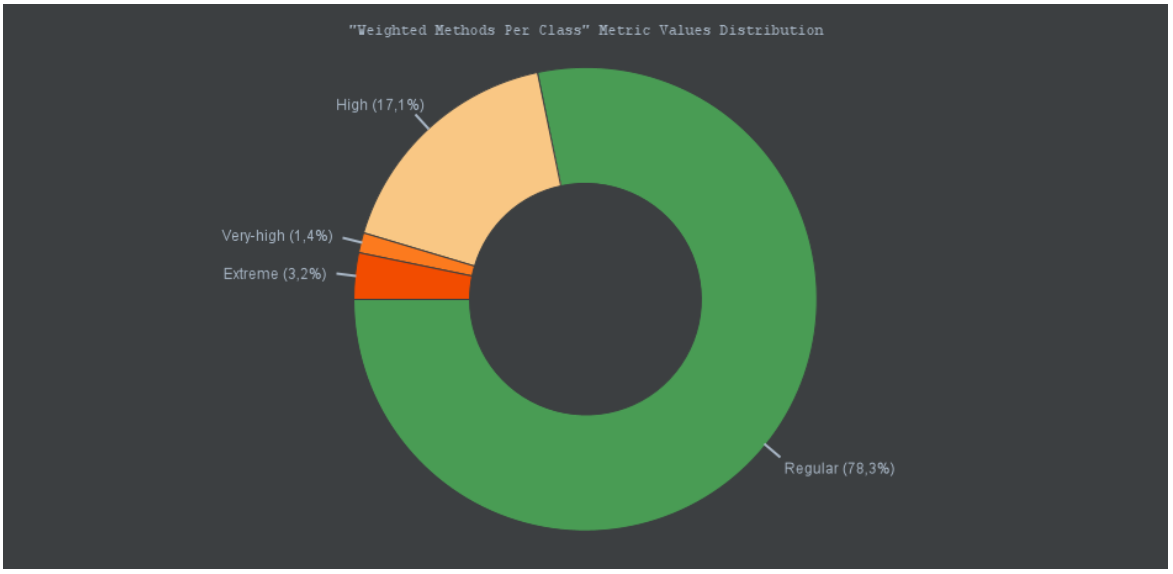
Class: Pay

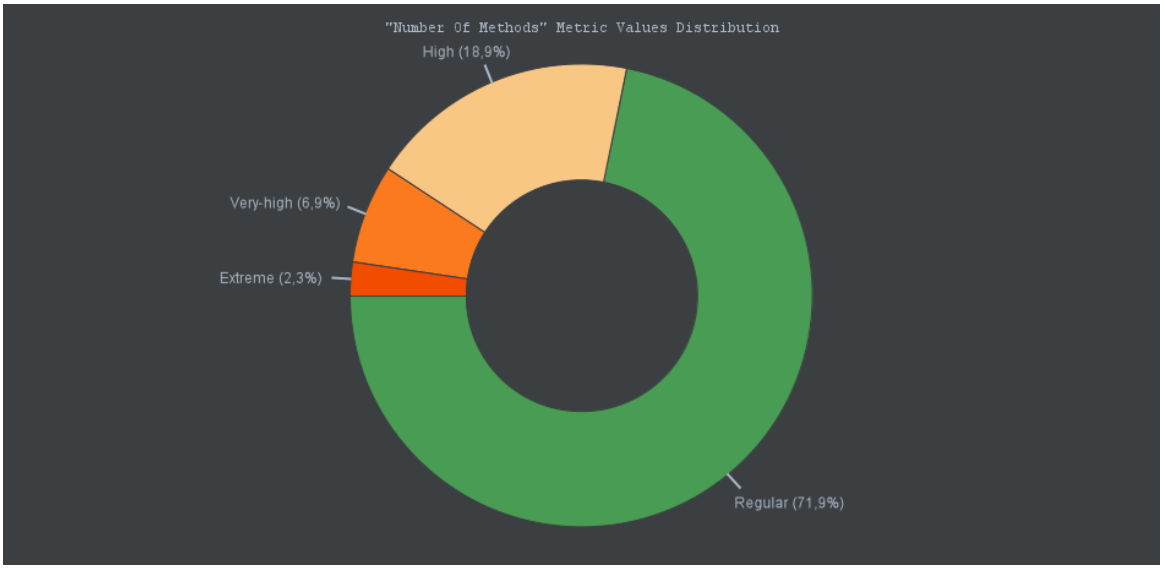
| Metric | Metrics Set | Description | Value | Regular Ra... |
|---|---|---|---|---|
| CHVL | Halstead M... | Halstead Volume | 46.507 | |
| CHD | Halstead M... | Halstead Difficulty | 2.0 | |
| CHL | Halstead M... | Halstead Length | 14 | |
| CHEF | Halstead M... | Halstead Effort | 93.014 | |
| CHVC | Halstead M... | Halstead Vocabulary | 10 | |
| CHER | Halstead M... | Halstead Errors | 0.0068 | |
| WMC | Chidamber-... | Weighted Methods Per Class | 7 | [0..12) |
| DIT | Chidamber-... | Depth Of Inheritance Tree | 1 | [0..3) |
| RFC | Chidamber-... | Response For A Class | 7 | [0..45) |
| LCOM | Chidamber-... | Lack Of Cohesion Of Methods | 3 | |
| NOC | Chidamber-... | Number Of Children | 0 | [0..2) |
| NOA | Lorenz-Kid... | Number Of Attributes | 3 | [0..4) |
| NOO | Lorenz-Kid... | Number Of Operations | 19 | |
| NOOM | Lorenz-Kid... | Number Of Overridden Methods | 0 | [0..3) |
| NOAM | Lorenz-Kid... | Number Of Added Methods | 6 | |
| SIZE2 | Li-Henry M... | Number Of Attributes And Methods | 22 | |
| NOM | Li-Henry M... | Number Of Methods | 7 | [0..7) |
| MPC | Li-Henry M... | Message Passing Coupling | 0 | |
| DAC | Li-Henry M... | Data Abstraction Coupling | 2 | |
| NCSS | Chr. Clemen... | Non-Commenting Source Statement.. | 2 | [0..1000) |
| CMI | Maintainabi... | Maintainability Index | 66.2817 | [0.0..19.0] |

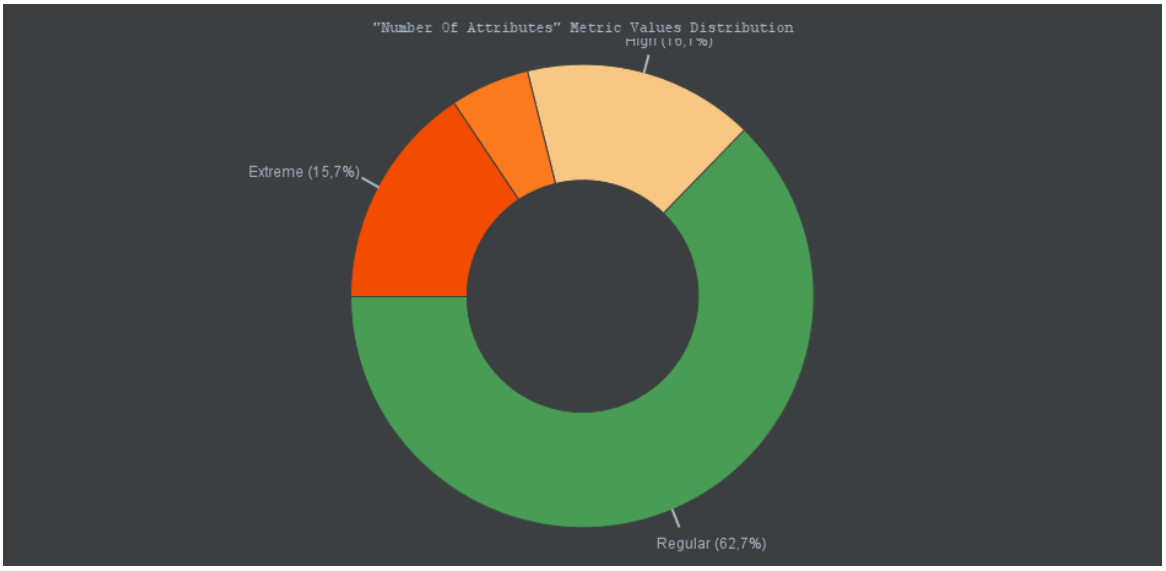
## Conclusion:

Looking at the metrics from after the refactoring, some significant changes were made. The goals set at the beginning of the assignment were met, looking at the overall project metrics, some of the classes no longer have extreme values for the metrics we set our focus on. More importantly, with the refactoring we took into account how adaptable our classes and methods are. By extracting classes, the codebase is more readable, and small changes can have a small impact. With method refactoring we reduced the coupling of some classes by moving methods. Overall the future adaptability of our project has been approved by a good margin.

# Appendix A:


"Weighted Methods Per Class" Metric Values Distribution
High (17,1%)
Very-high (1,4%)
Extreme (3,2%)
Regular (78,3%)


"Number Of Methods" Metric Values Distribution
High (18,9%)
Very-high (6,9%)
Extreme (2,3%)
Regular (71,9%)


"Number Of Attributes" Metric Values Distribution
Extreme (15,7%)
Regular (62,7%)