



**UNIVERSIDADE  
FEDERAL DO CEARÁ**  
CAMPUS DE QUIXADÁ

**CURSO DE ENGENHARIA DE SOFTWARE**

**RELATÓRIO – TRABALHO FINAL QUALIDADE DE SOFTWARE  
<Sistema Biblioteca UTFPR>**

**Equipe:**

**Ítalo Lima Dantas**

**Marcos Gênesis da Silva**

**Professora:**

**Carla Ilane Moreira Bezerra**

**QUIXADÁ**

**Março, 2021**

## SUMÁRIO

1	DESCRIÇÃO DO PROJETO	2
2	AValiação DO PROJETO	2
2.1	Medição 1 – Antes de refatorar o projeto	2
2.2	Detecção dos Code Smells	5
2.3	Medição 2 – Após Refatorar Code Smell God Class	7
2.4	Medição 3 – Após Refatorar Code Smell Intensive Coupling	8
2.5	Medição 4 – Após Refatorar Code Smell Shotgun Surgery	9
2.4	Medição 5– Após Refatorar Code Smell Feature Envy	10
2.5	Medição Z – Após a refatoração de todos os code smells do projeto	11
3	COMPARAÇÃO DOS RESULTADOS	14
	REFERÊNCIAS	16

## 1 DESCRIÇÃO DO PROJETO

O projeto Biblioteca UTFPR foi desenvolvido utilizando a linguagem JAVA, o mesmo tem como objetivo proposto a facilitação do controle e o gerenciamento das salas de estudo da biblioteca da UTFPR, no campus Mourão, é um sistema Desktop, no qual a linguagem de programação JAVA foi utilizada como tecnologia principal, além da conexão com banco de dados e interface gráfica, através do JavaFX e como esperado, segue o paradigma de Orientação a Objetos. Abaixo, está disposta uma captura de tela para exibir o repositório no *github*.

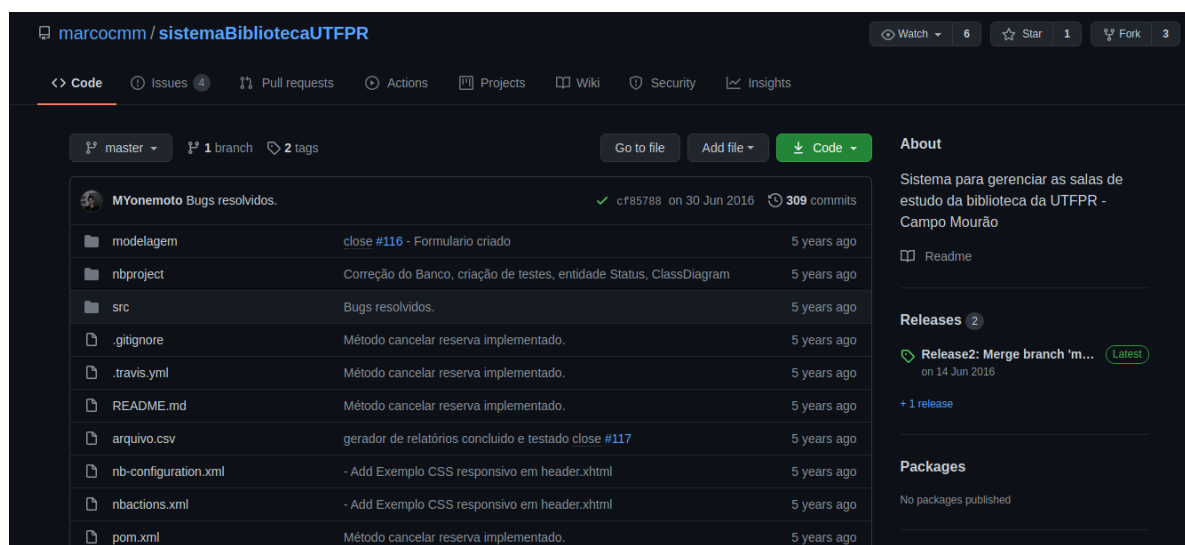


Figura 1 - Repositório do Projeto

[Link para o Repositório](#)

Abaixo, está disposta a tabela com as características descritivas do projeto.

Projeto	LOC	# de classes	# de releases
Biblioteca UTFPR	2798	38	2

Tabela 1 – Características do Projeto

## 2 AVALIAÇÃO DO PROJETO

### 2.1 2Medição 1 – Antes de refatorar o projeto

Nesta seção está incluída a Tabela com a medição das métricas de coesão, acoplamento, complexidade, herança e tamanho, antes do projeto ser refatorado. Para isso foi utilizada a

ferramenta *Understand*. Além disso, temos o link para a tabela com os resultados da primeira detecção e medição de *code smells*, no projeto.

Sistema	Coesão	Complexidade				Herança			Acoplamento	Tamanho			
	LCOM2	ACC	SCC	EVG	MaxNet	DIT	NOC	IFANIN	CBO	LOC	CLOC	NIM	CDL
S1 antes da refatoração	1248	40	462	0	41	34	4	41	92	2145	694	0	239
S1 após refat. CS X													
S1 após refat. CS X													

Atributos	Métricas	Descrição
Coesão	PercentLackOfCohesion	<p><i>Mede a coesão de uma classe</i></p> <p>↑ ↓</p> <p>Valor da métrica Coesão da Classe</p>
Acoplamento	CountClassCoupled	<p><i>Quantidade de classes que acoplam uma classe.</i></p> <p>↑ ↑</p> <p>Valor da métrica Complexidade</p>
Complexidade	Avg Cyclomatic	<p><i>Média da complexidade ciclomática de todos os métodos.</i></p> <p>↑ ↑</p> <p>Valor da métrica Complexidade</p>
	Sum Cyclomatic	<p><i>Somatório da complexidade ciclomática de todos os métodos.</i></p> <p>↑ ↑</p> <p>Valor da métrica Complexidade</p>
	Essential	<p><i>Nível máximo de aninhamento de construção de controle.</i></p> <p>↑ ↑</p> <p>Valor da métrica Complexidade</p>
























	Max Nesting	<i>Medição do grau na qual um módulo contém construtores não estruturados.</i>  Valor da métrica  Complexidade
Herança	Count ClassBase	<i>Quantidade de subclasses de uma classe.</i>  Valor da métrica  Herança
	CountClassDerived	<i>Quantidade de métodos e atributos que uma classe filho herda de uma classe.</i>  Valor da métrica  Herança
	MaxInheritanceTree	<i>Número imediato de classes base</i>  Valor da métrica  Herança
Tamanho	CountLineCode	<i>Número de linhas de código (sem comentários e espaços)</i>  Valor da métrica  Tamanho
	CountLineComment	<i>Número de linhas sem comentários</i>  Valor da métrica  Tamanho
	CountDeclClass	<i>Número de classes</i>  Valor da métrica  Tamanho
	CountDeclInstanceMethod	<i>Número de métodos de instância</i>  Valor da métrica  Tamanho

Tabela 2 – Medição dos atributos antes de refatorar o projeto.

✖ A relação entre os valores das métricas  e o significado desses valores  ou , representam a relação de proporcionalidade  e  indica uma relação proporcional, ou seja, quanto maior o valor da métrica, mais ela representará o significado do atributo de qualidade. A relação de proporcionalidade  e  indica uma relação inversamente proporcional, ou seja, quanto maior o valor da métrica, menos ela representará o significado do atributo de qualidade.

Legenda para a Tabela 2

Métricas 1ª Coleta de Code Smells													
1	2	ACOPLAMENTO		COESÃO		COMPLEXIDADE			HERANÇA			TAMANHO	
		CountClassCoupled	PercentLackOfCohesion	AvgCyclomatic	SumCyclomatic	Essential	MaxNesting	CountClassBase	CountClassDerived	MaxInheritanceTree	CountDeclClass	CountDeclInstanceMethod	CountLineCode
3	br.edu.utfpr.biblioteca.salas.controller	7	77	1	25	0	2	1	0	1	0	18	85
4	CalendarMB	8	78	1	14	0	1	1	0	1	0	10	54
5	ExibirReservaMB	1	0	1	5	0	1	2	0	1	0	3	23
6	LoginFilter	4	65	2	33	0	3	1	0	1	0	16	111
7	LoginMB	3	78	1	22	0	1	1	0	1	0	14	89
8	RelatorioMB	7	81	1	35	0	2	2	0	1	0	23	154
9	ReservaRápidaMB	1	88	1	12	0	1	1	0	1	0	8	37
10	SessionContext	2	42	1	8	0	1	1	0	1	0	7	28
11	br.edu.utfpr.biblioteca.salas.model	2	45	1	15	0	1	2	0	1	0	11	59
12	Dia	0	62	1	10	0	0	1	0	1	0	10	39
13	Hora	0	50	1	4	0	0	1	0	1	0	4	16
14	RelatorioReservas	5	0	4	4	0	2	1	0	1	0	1	21
15	ReservasHorario	2	50	3	12	0	2	1	0	1	0	4	80
16	br.edu.utfpr.biblioteca.salas.model.bo	13	60	2	23	0	2	1	0	1	0	0	104
17	AdministradorBO	12	37	2	18	0	2	1	0	1	0	0	82
18	ParserCnvBO	0	0	0	0	0	0	1	0	1	0	0	2
19	ReservaBO	5	11	1	15	0	1	1	0	1	0	0	59
20	SalasBO	0	28	1	9	0	1	1	4	1	0	7	41
21	StatusBO	5	0	1	13	0	2	2	0	2	0	9	76
22	UsuarioBO	3	0	1	13	0	2	1	0	2	0	8	76
23	br.edu.utfpr.biblioteca.salas.model.dao	1	0	1	2	0	0	1	0	2	0	2	9
24	GeneroDAO	4	0	1	10	0	2	1	0	2	0	6	59
25	ReservaDAO	5	0	1	9	0	1	2	0	1	0	25	145
26	SalasDAO	0	64	1	16	0	1	2	0	1	0	13	64
27	StatusDAO	0	55	1	12	0	1	2	0	1	0	9	49
28	UsuarioDAO	0	83	1	26	0	2	2	0	1	0	19	125
29	br.edu.utfpr.biblioteca.salas.model.entity	0	0	1	50	0	2	1	0	1	0	0	371
30	ReservaPO	0	84	1	31	0	1	2	0	1	0	25	145
31	SalasPO	0	64	1	16	0	1	2	0	1	0	13	64
32	StatusPO	0	55	1	12	0	1	2	0	1	0	9	49
33	UsuarioPO	0	83	1	26	0	2	2	0	1	0	19	125
34	br.edu.utfpr.biblioteca.salas.tools	0	0	1	50	0	2	1	0	1	0	0	371
35	CalendarFilter	0	0	1	50	0	2	1	0	1	0	0	371

[Clique aqui para acessar a tabela.](#)

## 2.2 Detecção dos Code Smells

Nessa seção estão indicados quais e quantos code smells foram detectados no projeto. A detecção foi feita de forma automatizada, utilizando a ferramenta JSPirit, em integração com a IDE Eclipse.

Nome do Code Smell	Quantidade
Feature Envy	42
Data Class	1
Intensive Coupling	2
Dispersed Coupling	9
God Class	1
Shotgun Surgery	2
<b>Total</b>	<b>58</b>

Tabela 3 – Code smells do projeto.

### Descrição dos code smells:

- ***Feature Envy:***
  - Quando uma classe utiliza mais métodos/atributos de uma outra classe do que da própria.
  
- ***Data Class:***
  - Refere a uma classe que contém apenas campos e métodos “brutos” para acessá-los (getters e setters). Esses são simplesmente contêineres de dados usados por outras classes. Essas classes não contêm nenhuma funcionalidade adicional e não podem operar independentemente nos dados que possuem.
  
- ***Intensive Coupling:***
  - Quando uma classe utiliza muitos métodos de uma única outra classe (ou de poucas).
  
- ***Dispersive Coupling:***
  - Quando uma classe utiliza muitos métodos de muitas outras classes.
  
- ***God Class***
  - Quando uma classe “sabe muito” ou “faz muita coisa”, geralmente é uma classe com um número maior de linhas de código e tem muitas responsabilidades.

- **Shotgun Surgery**

- A realização de uma mudança desencadeia uma sucessão de mudanças em outras partes do código.

### 2.3 Medição 2 – Após Refatorar God Class

Sistema	Coesão	Complexidade				Herança			Acoplamento	Tamanho			
	LCOM2	ACC	SCC	EVG	MaxNet	DIT	NOC	IFANIN	CBO	LOC	CLOC	NIM	CDL
S1 antes da refatoração	1248	40	462	0	41	34	4	41	92	2145	694	0	239
S1 após refat. God Class	1249	40	457	0	40	34	4	41	92	2082	666	0	241

**Técnica utilizada:** Como foi supracitado, o code smell “God class” diz respeito à uma classe que sabe muito ou tem muitas responsabilidades, para resolver esse tipo de code smell é necessário entender primeiramente qual dessas duas situações está ocorrendo, no nosso caso a classe x tinha uma quantidade elevada de métodos referentes a responsabilidades distintas, um indicativo de god class é o número de linhas. Para resolver esse smell, nos baseamos em dois princípios do SOLID (um conjunto de padrões de design que têm como objetivo facilitar o entendimento, desenvolvimento e a manutenção do código), são esses: Responsabilidade única e segregação de interface, ambos inferem que o código deve ser dividido em “pedaços”, de acordo com a sua responsabilidade, mantendo um baixo acoplamento. Durante a resolução do god class, tivemos também a resolução de vários code smells da categoria dispersed coupling, onde uma classe acessa vários métodos de várias outras classes, isso acontecia devido ao fato da classe ter muitas responsabilidades e realizar muitas ações, assim, tinha que saber muito também, acessando assim dados e métodos de outras classes.



**Análise:** No geral, a refatoração desse code smell não teve impacto significativo na medição das métricas em questão, o máximo observado, foram alterações óbvias na quantidade de linhas de código e comentários, além de uma leve melhoria nos dados referentes à complexidade, relacionado à herança, não houveram modificações na estrutura de árvore de herança, dentro das classes, dessa forma as métricas para o atributo se mantiveram inalteradas. Para a coesão houve uma leve piora.

## 2.4 Medição 3 – Após Refatorar Intensive Coupling

3

Sistema	Coesão	Complexidade				Herança			Acoplamento	Tamanho			
	LCOM2	ACC	SCC	EVG	MaxNet	DIT	NOC	IFANIN	CBO	LOC	CLOC	NIM	CDL
S1 antes da refatoração	1248	40	462	0	41	34	4	41	92	2145	694	0	239
S1 após refat. God Class	1249	40	457	0	40	34	4	41	92	2082	666	0	241
S1 após refat. Intensive Coupling	1249	40	444	0	40	34	4	41	91	2008	712	0	238

**Técnica utilizada:** Como supracitado, o code smell “intensive coupling” é identificado em métodos, quando estes utilizam / “chamam” muitos outros métodos de várias outras classes, nesse caso, utilizamos a seguinte linha de raciocínio: se um método precisa utilizar com frequência elevada métodos de outras classes, então, esse método deveria estar na própria classe, seja com o acesso através de um objeto da classe correspondente, ou seja o próprio método movido para essa classe que o invoca. Não houve o caso de mover métodos e deixar a classe quase vazia (o que poderia originar outro tipo de code smell, o “data class”).

**Análise:** No geral, a refatoração desse code smell não teve impacto significativo na medição das métricas em questão, o máximo observado, foram alterações na quantidade de

linhas de código e comentários, além de uma boa melhoria para uma das métricas de complexidade (SCC). Relacionado à herança, não houveram modificações na estrutura de árvore de herança, dentro das classes, dessa forma as métricas para o atributo se mantiveram inalteradas, a coesão se manteve estável.

### 3.1 Medição 4 - Após Refatorar Shotgun Surgery

4

Sistema	Coesão	Complexidade				Herança			Acoplamento	Tamanho			
	LCOM2	ACC	SCC	EVG	MaxNet	DIT	NOC	IFANIN	CBO	LOC	CLOC	NIM	CDL
S1 antes da refatoração	1248	40	462	0	41	34	4	41	92	2145	694	0	239
S1 após refat. God Class	1249	40	457	0	40	34	4	41	92	2082	666	0	241
S1 após refat. Intensive Coupling	1249	40	444	0	40	34	4	41	91	2008	712	0	238
S1 após refat. Shotgun Surgery	1249	39	446	0	41	34	4	41	91	2015	713	0	239

**Técnica utilizada:** Como supracitado, o code smell “shotgun surgery” representa um problema de controle de mudanças, quando, no caso da realização de uma mudança em uma classe, ela desencadeia uma sucessão de mudanças em outras, que possuem a chamada do método em questão. Para resolver esse problema, utilizamos as estratégias de mover os métodos correspondentes e mover também atributos dos objetos que sofrem essa modificação, essas são definidas como “move method” e “move field”,

**Análise:** No geral, a refatoração desse code smell não teve impacto significativo na medição das métricas em questão, o máximo observado, foram alterações óbvias na quantidade de linhas de código e comentários, neste caso, houve uma leve piora na medição referente à complexidade, com duas métricas impactadas.

#### 4.1 Medição 5 - Após Refatorar Feature Envy

5

Sistema	Coesão	Complexidade				Herança			Acoplamento	Tamanho			
	LCOM2	ACC	SCC	EVG	MaxNet	DIT	NOC	IFANIN	CBO	LOC	CLOC	NIM	CDL
S1 antes da refatoração	1248	40	462	0	41	34	4	41	92	2145	694	0	239
S1 após refat. God Class	1249	40	457	0	40	34	4	41	92	2082	666	0	241
S1 após refat. Intensive Coupling	1249	40	444	0	40	34	4	41	91	2008	712	0	238
S1 após refat. Shotgun Surgery	1249	39	446	0	41	34	4	41	91	2015	713	0	239
S1 após refat. Feature Envy	1477	38	459	0	41	33	3	41	90	2057	715	0	255

**Técnica utilizada:** Como supracitado, o code smell “feature envy” aparece quando alguns atributos são movidos para classes de dados e os métodos que os utilizam não foram, isso indica a necessidade de movê-los, ou, também identificar a ocorrência do smell em casos que um objeto e suas propriedades são mais utilizadas em uma outra classe, do que na classe que estão localizados. Para resolver esse impasse, utilizamos as estratégias de “move field”, “move method”, “extract method” e em alguns casos específicos, foram realizadas alterações na modelagem em questão, das classes, com uma reestruturação lógica de suas propriedades.

**Análise:** Esse code smell foi o que teve refatoração ainda de forma parcial e causou o maior impacto nas medições observadas, resultados mistos compõem a análise do mesmo, visto que no quesito coesão e complexidade, houve uma piora de significância maior que a registrada nas outras medições, no entanto, o atributo de herança teve os primeiros indícios de mudança, com melhora em duas das quatro métricas.

Nome do Code Smell	Quantidade
Feature Envy	20
Data Class	1
Intensive Coupling	0
Dispersed Coupling	3
God Class	0
Shotgun Surgery	0
<b>Total</b>	<b>24</b>

Tabela 4 – Code smells do projeto após a primeira entrega.

Métricas 1º Coleta de Code Smells											
Métricas 1º Coleta de Code Smells	COESÃO		COMPLEXIDADE				HERANÇA		ACOPLAMENTO		
	PercentLackOfCohesion	AvgCyclomatic	SumCyclomatic	Essential	MaxNesting	MaxInheritanceTree	CountClassDerived	CountClassBase	CountClassCoupled	CountLineCode	CountLine
br.edu.utfpr.biblioteca.salas.controller											
CalendarioMB	77	1	25	0	2	1	0	1	7	85	2
ExibirReservaMB	78	1	14	0	1	1	0	1	8	54	5
LoginFilter	0	1	5	0	1	1	0	2	1	23	1
LoginMB	65	2	33	0	3	1	0	1	4	131	4
RelatorioMB	78	1	22	0	1	1	0	1	3	89	0
ReservaRapidaMB	81	1	35	0	2	1	0	2	7	154	4
SessionContext	88	1	12	0	1	1	0	1	1	37	3
br.edu.utfpr.biblioteca.salas.model											
Dia	42	1	8	0	1	1	0	1	2	28	6
Hora	45	1	15	0	1	1	0	2	2	59	1
RelatorioReservas	62	1	10	0	0	1	0	1	0	39	4
ReservasHorario	50	1	4	0	0	1	0	1	0	16	4
br.edu.utfpr.biblioteca.salas.model.bo											
AdministradorBO	0	4	4	0	2	1	0	1	5	21	4
ParserCsvBO	50	3	12	0	2	1	0	1	2	80	1
ReservaBO	60	2	23	0	2	1	0	1	13	104	6
SalaBO	37	2	18	0	2	1	0	1	12	82	5
StatusBO	0	0	0	0	0	1	0	1	0	2	4
UsuarioBO	11	1	15	0	1	1	0	1	5	59	3
br.edu.utfpr.biblioteca.salas.model.dao											
GenericDAO	28	1	9	0	1	1	4	1	0	41	4
ReservaDAO	0	1	13	0	2	2	0	2	5	76	5
SalaDAO	0	1	13	0	2	2	0	1	3	76	5
StatusDAO	0	1	2	0	0	2	0	1	1	9	4

[Link para a planilha atualizada](#)

### 5.1 Medição Z – Após a refatoração de todos os code smells do projeto

Sistema	Coesão	Complexidade				Herança			Acoplamento	Tamanho			
	LCOM2	ACC	SCC	EVG	MaxNet	DIT	NOC	IFANIN	CBO	LOC	CLOC	NIM	CDL
S1 antes da refatoração	1248	40	462	0	41	34	4	41	92	2145	694	0	239
S1 após refat. God Class	1249	40	457	0	40	34	4	41	92	2082	666	0	241
S1 após refat. Intensive Coupling	1249	40	444	0	40	34	4	41	91	2008	712	0	238
S1 após refat. Shotgun Surgery	1249	39	446	0	41	34	4	41	91	2015	713	0	239
S1 após refat. Feature Envy	1477	38	459	0	41	33	3	41	90	2057	715	0	255
S1 após refat. 51 Code Smells	1649	37	462	0	41	33	3	41	90	2080	716	0	279

**Técnica utilizada:** Os code smells refatorados nessa release do trabalho foram principalmente os de “feature envy”, para estes em questão, o que havia no projeto era a utilização inadequada de muitos métodos estáticos dentro das classes, e então quando outras classes desejavam acessar aqueles métodos, era necessário utilizar uma instância da classe detentora do método, o que resulta no acesso indevido. Para resolver esse impasse, os métodos deixaram de ser estáticos e então utilizamos as estratégias supraexplicadas de move method e extract method, consequentemente, movemos também os atributos (dados) correspondentes que são utilizados dentro do escopo do método, assim, a classe que deseja utilizar um certo dado em sua operação, agora contém esse dado na sua estrutura interna.

**Análise:** Após as mudanças realizadas pela última etapa de refatoração, foi observado uma variação nas métricas de complexidade, visto que a SCC aumentou, a ACC diminuiu, e as outras se mantiveram estáveis. Para a herança, não houve, nessa release alterações referentes a árvore de dependências entre as classes, novas classes baseadas em outras também não foram adicionadas e isso manteve as métricas estáveis, para o atributo em questão. As métricas relacionadas ao tamanho sofreram alterações, LOC CLOC e CDL aumentaram, visto que tivemos que criar dados em classes e em vários casos criar instâncias que representam esses dados, utilizando assim o acesso através do objeto, ao invés de ser diretamente da classe.

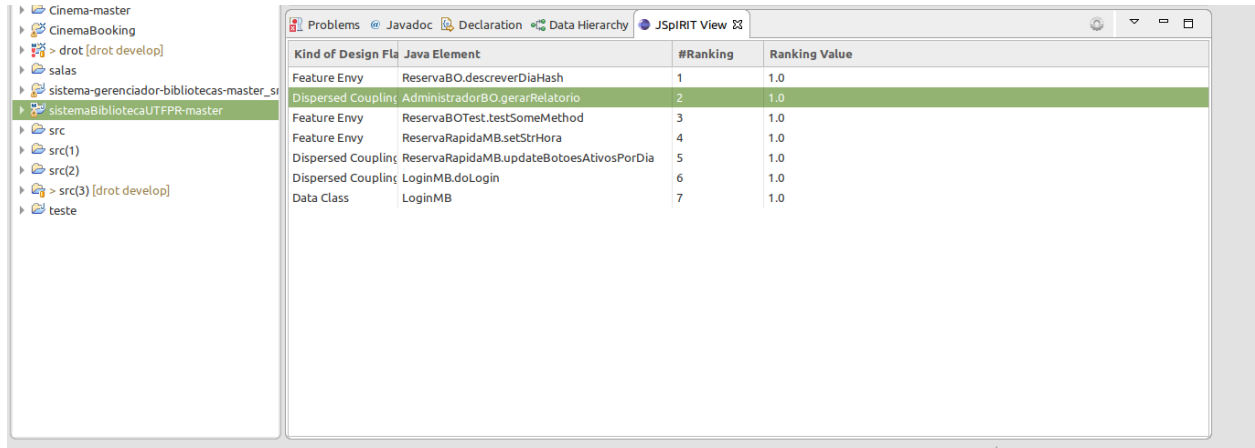
Nome do Code Smell	Quantidade
Feature Envy	3
Data Class	1
Intensive Coupling	0
Dispersed Coupling	3
God Class	0
Shotgun Surgery	0
<b>Total</b>	<b>7</b>

Tabela 5 – Code smells do projeto após a entrega final.

Métricas 1ª Coleta de Code Smells										
		COESÃO		COMPLEXIDADE			HERANÇA		ACOPLAMENTO	
		PercentLackOfCohesion	AvgCyclomatic	SumCyclomatic	Essential	MaxNesting	MaxInheritanceTree	CountClassDerived	CountClassBase	CountClassCoupled
1	Métricas 1ª Coleta de Code Smells									
2	br.edu.utfpr.biblioteca.salas.controller									
3	CalendarioMB	77	1	25	0	2	1	0	1	7
4	ExibirReservaMB	78	1	8	0	1	1	0	1	8
5	LoginFilter	50	1	6	0	1	1	0	2	1
6	LoginMB	65	2	33	0	3	1	0	1	4
7	RelatorioMB	80	1	22	0	1	1	0	1	3
8	ReservaRapidaMB	85	1	36	0	2	1	0	2	7
9	SessionContext	88	1	12	0	1	1	0	1	1
10	br.edu.utfpr.biblioteca.salas.model									
11	Dia	42	1	8	0	1	1	0	1	2
12	Hora	40	1	15	0	1	1	0	2	2
13	RelatorioReservas	59	1	11	0	0	1	0	1	0
14	ReservasHorario	50	1	4	0	0	1	0	1	0
15	br.edu.utfpr.biblioteca.salas.model.bo									
16	AdministradorBO	0	4	4	0	2	1	0	1	5
17	ParserCsvBO	61	2	14	0	2	1	0	1	2
18	ReservaBO	84	2	26	0	2	1	0	1	12
19	SalaBO	37	1	15	0	2	1	0	1	10
20	StatusBO	0	0	0	0	0	1	0	1	0
21	UsuarioBO	45	1	17	0	1	1	0	1	5
22	br.edu.utfpr.biblioteca.salas.model.dao									
23	GenericDAO	28	1	9	0	1	1	3	1	0
24	ReservaDAO	83	1	24	0	3	1	0	2	5
25	SalaDAO	25	1	13	0	2	2	0	1	3
26	StatusDAO	0	1	2	0	0	2	0	1	1
27										

[Link para a planilha atualizada](#)

## 6 COMPARAÇÃO DOS RESULTADOS



**Captura de tela que exibe os code smells restantes na aplicação.**

	Coesão	Complexidade	Herança	Acoplamento	Tamanho
God Class	↑	↓	-	-	↓
Shotgun Surgery	-	↑	-	-	↑
Feature Envy	↑	↓	↓	↓	↑
Intensive Coupling	-	↓	-	↓	↓
Dispersed Coupling	↑	-	-	-	↑

**Tabela 6 – Resumo da variação dos atributos de qualidade durante as iterações de refatoração.**

Para interpretação da tabela acima, temos que (↑) representa que durante a refatoração específica de determinado code smell, os valores para as métricas com essa label aumentaram, em contrapartida, (↓) representa que durante a refatoração específica de determinado code smell, os valores para as métricas com essa label diminuíram. Essa é a primeira representação dos resultados quantitativos, de forma superficial. Realizando uma análise estatística um pouco mais

aprofundada, temos que o atributo de coesão, a partir da métrica LCOM2 aumentou em aproximadamente 32%. Para a complexidade, houveram variações dentro de cada release de refatoração, o que pode significar desvios dentro do esperado, seguindo uma margem de erro estatística esperada, apenas para a métrica de ACC que foi registrada uma melhoria de 7.5%. Referente à herança, as métricas de DIT e NOC melhoraram em 25%, ambas. Com resultados menos significativos, temos o atributo de acoplamento, que obteve uma melhoria pouco superior a 2%.

Realizando uma análise das variações dos atributos, para cada release, podemos observar em quantos % cada um em questão variou, exemplo: como foram realizadas 5 medições, o atributo que aumentou em todas, aumentou em 100% das iterações. A tabela 6 registra justamente essa perspectiva, tendo como resultados: A coesão piorou em 60% das medições e se manteve estável nos outros 40%. A complexidade melhorou em 60% dos casos, já a herança se manteve estável em 80% deles. Para o acoplamento, a estabilidade foi predominante, estando identificada em 60% das releases. Para os atributos de tamanho, houveram variações já supra exemplificadas, mas que em resumo aumentaram em 60% das iterações.

Como conclusão, observa-se que a refatoração desempenhada nesse sistema não foi algo relativamente significativo, no caso de termos como base as métricas para os atributos de qualidade, verificadas pelas medições na ferramenta *Understand*. Entretanto, a remoção dos code smells e em suas instâncias se mostrou eficiente através da aplicação de técnicas previamente definidas por estudos, para cada tipo de problema, nessa etapa, estudamos o que cada code smell significava, como esse problema poderia se apresentar a nível de código (com entendimento genérico e abstrato, intensificado por exemplos) e então inferimos em como cada técnica poderia ser aplicada para a remoção do code smell, e então vem a aplicação de fato, através do código. Aproximadamente 88% das instâncias de code smells foram removidos, o detalhamento da porcentagem para cada tipo ficou assim: 99% dos *feature envy* foram resolvidos, 100% dos *intensive coupling*, 67% dos *dispersed coupling* e 100% dos *god class* e *shotgun surgery*, respectivamente.



## REFERÊNCIAS

AZEEM, Muhammad. Machine learning techniques for code smell detection: A systematic literature review and meta-analysis. *Information and Software Technology*, v. 108, p. 115-138, 2019.

SABIR, Fatima. A systematic literature review on the detection of smells and their evolution in object-oriented and service-oriented systems. *Software: Practice and Experience*, v. 49, n. 1, p. 3-39, 2019.